ulm university universität
uulm

Text Classification for Spoken Dialogue Systems

Dissertation zur Erlangung des Doktorgrades Dr.rer.nat.
der Fakultät für Ingenieurwissenschaften, Informatik und
Psychologie der Universität Ulm

von

Roman Sergienko

aus Krasnoyarsk

2016

Amtierender Dekan: Prof. Dr. Tina Seufert

Gutachter: Prof. Dr. Dr.-Ing. Wolfgang Minker

Gutachter: Juniorprof. Dr. Birte Glimm

Tag der Promotion: 03. Juni 2016

Abstract

The main objective of this thesis is the application and evaluation of text classification approaches for speech-based utterance classification problems in the field of advanced spoken dialogue system (SDS) design.

SDSs are speech-based human-machine interfaces that may be applied in various domains. A novel generation of SDSs should be multi-domain and user-adaptive. Designing multi-domain user-adaptive SDSs is related to some utterance classification problems: domain detection of user utterances and user state recognition including user verbal intelligence and emotion recognition. Text classification approaches may be applied for the considered problems. Text classification consists of the following stages: feature extraction, term weighting, dimensionality reduction, and machine learning.

The thesis has three aims:

1. To identify the best combinations of state-of-the-art text classification approaches for the considered utterance classification problems.
2. To improve utterance classification performance for SDSs.
3. To improve computational performance of utterance classification for SDSs.

For the first aim, different term weighting methods (IDF, CW, GR, TM2, RF, TRR, and NTW), different dimensionality reduction methods („stop-word“ filtering in combination with stemming, weight-based feature selection, feature transformation based on term clustering), and different machine learning algorithms (*k*-NN, SVM, the Rocchio classifier) have been validated on different datasets including two corpora for domain detection of user utterances, two corpora for verbal intelligence recognition, and a corpus for text-based user emotion recognition.

The best combinations of the text classification approaches were identified as follows:

- *For domain detection of user utterances*: *k*-NN + TRR or SVM + IDF with feature transformation based on term clustering.
- *For user verbal intelligence*: *k*-NN + CW. The CW method for verbal intelligence recognition provides a small number of useful terms that characterize only a class of higher verbal intelligence. It seems to be easier to recognize verbal intelligence in dialogues than in monologues

- *For text-based emotion recognition*: *k*-NN + NTW without dimensionality reduction. Emotion recognition based on linguistic information does not demonstrate high performance in comparison with audio-based and video-based emotion recognition.

Novel approaches were proposed and tested for achieving the second and the third aims of the thesis:

- *Collectives of different term weighting methods*. This approach allows to make use of the advantages of different term weighting methods. Collectives of term weighting methods based on the majority voting procedure may significantly improve the classification performance of utterance classification with the *k*-NN algorithm. These results may be again significantly improved by weighted voting with an optimization based on a self-adjusting genetic algorithm (GA).

- *Novel feature transformation based on terms belonging to classes*. It significantly reduces the dimensionality: it equals to the number of classes. The novel feature transformation significantly improves the computational performance of utterance classification in terms of computational time. The novel feature transformation method is especially effective in combination with the collectives of term weighting methods. The simultaneous use of two novel approaches may significantly improve both the classification results and the computational performance.

- *Novel approach to neural network structure optimization*. The novel approach has a simplified structure representation of artificial neural networks (ANN), requires less computational resources, and has fewer parameters for tuning than the baseline approach. Additionally, the results of the novel approach to ANN structure optimization may be improved with feature selection based on the wrapper. The wrapper may be performed by the self-adjusting GA. The novel approach to ANN design significantly improves the classification results and the computational performance of utterance classification with ANNs.

Therefore, the novel approaches lead to an improved classification performance of utterance classification (the second aim of the thesis) and the computational performance of utterance classification (the third aim of the thesis) as well.

Contents

List of Figures

List of Tables

1. Introduction

1.1. Motivation

Speech is a natural, quick and convenient way of communication between users and technical systems. Nowadays, an increasing number of different systems incorporate speech interaction (so-called spoken dialogue systems or SDSs). Not only personal assistants, such as Apple's Siri (http://www.apple.com/ios/siri/) or Samsung's SVoice (http://www.samsung.com/global/galaxys3/svoice.html) but also speech-based navigation systems (Geunter et al., 1998), telephone-based spoken dialogue systems (Mengistu and Wendemuth, 2007), or natural language call routing systems (Suhm et al., 2002) simplify our everyday live. Spoken dialogue systems are the third main generation of human-machine interfaces after batch processes and GUIs (graphical user interfaces).

While speech recognition and semantic interpretation have been the main fields of research in this area, further knowledge sources about interactions between human and machine have been analyzed. This research shall allow making spoken dialogue systems more intelligent and more adaptive in the future. Nowadays, there are some directions in the area of spoken dialogue system design that constitute a breakthrough in the field of human-machine interfaces:

- *Multi-domain spoken dialogue systems* (Lee et al., 2009; Komatani et al., 2009). Multi-domain SDSs should be able to process user queries in different domains. Such multi-domain SDSs are necessary for implementation of ubiquitous computing concepts, e.g. in intelligent houses (Sakamura, 2005). An intelligent house with a multi-domain SDS would provide a possibility for voice control of all systems within the intelligent house simultaneously.

- *User-adaptive spoken dialogue systems* (Schmitt and Minker, 2012). For future application areas for SDS such as social robotics or active listener, it is necessary not only to process a user query properly but also to adapt a human-machine dialogue to a user state. There are different modalities of a user state that may be useful for a user-adaptive spoken dialogue system: user emotions, user verbal intelligence, user gender, and age. Correct user state recognition will render an SDS more natural and user-friendly.

Designing effective and adaptive SDSs is related to complex analysis of different types of data: acoustic signals, video streams, utterance information after automatic

speech recognition (ASR) (Yu and Deng, 2012). The analysis of utterance transcripts is one of the most important parts of SDS design.

Semantic analysis of a user utterance based on natural language understanding is a typical module of a standard SDS (Dybkjær and Minker, 2008). Natural language understanding (Kamp and Reyle, 2013; Cambria and Bruce, 2014) is a very important part of such commercial systems as Microsoft LUIS (https://www.luis.ai/) and Nuance MIX (https://developer.nuance.com/mix).

The next generation of multi-domain user-adaptive SDSs should include some additional modules based on text processing applied on user utterances:

- *Domain detection*. Multi-domain SDSs should include different specific semantic models for each problem domain. Prior to a detailed semantic analysis the general topic domain of a user utterance needs to be determined. Such a topic detection is also based on text classification. Designing domain-related semantic models after a domain detection step may be more effective than designing a complex semantic model that applies to all domains. This is especially significant in the case of multi-language SDSs because semantic analyses require a specific model for each language.

- *User state recognition including verbal intelligence and emotional status recognition.* An effective user state recognition module provides a user-adaptive SDS. It may be possible to recognize a user verbal intelligence level using text processing of user utterances (Zablotskaya et al., 2012). Typically, acoustic and video information is used for emotion recognition (Sidorov and Minker, 2014c). However, linguistic information may be an additional useful modality for multi-modal emotion recognition.

The above described problems can be formulated as text classification tasks. Based on a set of predefined categories or classes the task is to automatically assign new text documents to one of these categories. In our case the text documents are user utterances or sets of user utterances in textual format after automatic speech recognition. In the following text classification applied to user utterance transcripts are denoted utterance classification. Categories for utterance classification can be formulated as follows:

- Domain detection of user utterance: different domains (e.g. technical support, account information, or service cancelling in an automatic call center as a SDS).

- User verbal intelligence recognition: different levels of user verbal intelligence (e.g. high or level).

- Emotion recognition: different kinds of user emotions (e.g. anger or happiness).

Therefore, **the main objective** of this thesis is the application and evaluation of text classification approaches for speech-based problems in the field of advanced spoken dialogue system design.

Generally, text classification consists of the following steps: textual feature extraction based on raw document processing, numerical feature extraction based on term weighting, dimensionality reduction, and classification with machine learning algorithms.

Textual feature extraction comprises tokenization, dictionary forming, and special procedures such as stop-words filtering (Fox, 1989) and stemming (Porter, 2001).

Numerical feature extraction based on term weighting is necessary for machine learning because the majority of machine learning algorithms process vectors of numerical features. There are different unsupervised and supervised term weighting methods: TF-IDF (Salton and Buckley, 1988), Gain Ratio (GR) (Debole and Sebastiani, 2004), Confident Weights (CW) (Soucy and Mineau, 2005), Term Second Moment (TM2) (Xu and Li, 2007), Relevance Frequency (RF) (Lan et al., 2009), Term Relevance Ratio (TRR) (Ko, 2012), and Novel Term Weighting (NTW) (Gasanova et al., 2014).

As a rule, the dimensionality for text classification problems is very high; it equals to the dictionary size of the text classification problem (thousands or even millions of terms). Due to the high dimensionality, the classification may be inappropriately time-consuming, especially for real-time spoken dialogue systems. Therefore, the next step of text classification is dimensionality reduction. Dimensionality reduction methods are divided into feature selection and feature transformation.

The last step is an application of machine learning algorithms for classification with the obtained numerical feature set. The *k* nearest neighbors algorithm (*k*-NN), support vector machines (SVM), and artificial neural networks (ANN) are examples of effective algorithms for text classification (Han et al., 2001; Kwon and Lee, 2003; Joachims, 2002; Baharudin et al., 2010; Morariu et al., 2005; Trappey et al., 2006).

A correct choice of techniques for term weighting, dimensionality reduction, and machine learning is very important for effective text classification. Classification results and computational effectiveness significantly vary depending on the approach and domain. There exists a variety of comparative studies in the area of text classification (Baharudin et al., 2010; Lan et al., 2005; Youn and McLeod, 2007; Deng et al., 2004). However, almost all of them describe results with written text or written documents. Speech-based text classification problems that are important for SDS design have some specific characteristics in comparison with written text:

- Speech utterances are usually short; they may contain only one simple sentence. As a rule, written documents have larger size.

- Dictionaries for speech-based utterance classification problems are usually smaller than dictionaries for written documents. It is a well-known fact from linguistics.

Due to these characteristics, those results obtained for written text classification may be not completely appropriate for utterance classification. Therefore, **the first aim** of this thesis is to identify the most effective combinations of term weighting methods, dimensionality reduction techniques, and classification algorithms for utterance classification. Such a comparative study of techniques for different utterance classification problems (utterance topic categorization, verbal intelligence recognition, and text-based emotion recognition) is novel.

Text and utterance classification results may be improved by a simultaneous use of different approaches. Meta-classification based on a collective use of different classification algorithms with different settings is a very popular direction in machine learning; it is also possible to improve text classification results with meta-classification (Schapire and Singer, 2000; Morariu et al., 2005). We propose an idea that not only a combination of different machine learning algorithms but also a combination of different term weighting methods may improve the classification results of text classification including utterance classification. The results obtained when applying the same classification algorithm but with different term weighting methods may differ from each other much more than the results for different classification algorithms using the same term weighting method. Therefore, we propose a novel approach for text classification based on a collective use of different term weighting methods. It may lead to achieve **the second aim** of this thesis: to improve utterance classification results.

The third aim of this thesis is to improve computational effectiveness of utterance classification with novel approaches of dimensionality reduction and machine learning. For real-time SDSs, computational effectiveness of utterance classification is also a very important criterion. It is possible to improve computational effectiveness of text classification with significant dimensionality reduction or with less time-consuming procedures of machine learning. Therefore, the development of effective dimensionality reduction techniques and less time-consuming procedures of machine learning are actual scientific and technical problems. We propose a novel feature transformation method based on terms belonging to classes, which is able to significantly reduce the dimensionality of the text classification problem, and a novel approach to neural network

design, which requires less computational resources than existing methods, to achieve the third aim of the thesis. Additionally, these approaches may also lead to improve classification results (the second aim of the thesis).

1.2. Organization of the Thesis

This thesis consists of six chapters.

Chapter 1 states the main objectives and motivation for the proposed research of text classification for SDS design.

Chapter 2 describes the background and the state-of-the-art on the general area of SDSs, speech-based text classification for SDS design, the structure of text classification steps and their detailed description including term weighting methods, dimensionality reduction techniques, and machine learning algorithms.

Chapter 3 describes the considered corpora for speech-based text classification: two corpora for domain detection of user utterances in English, two corpora for user verbal intelligence recognition in German, and one corpus for emotion recognition in German.

Chapter 4 presents a comparative study of state-of-the-art text classification approaches including term weighting methods, dimensionality reduction techniques, and classification algorithms applied for the considered utterance classification problems. This chapter is related to the first aim of the thesis.

Chapter 5 presents a description and an evaluation of the novel proposed approaches for text classification applied for user utterances: collectives of different term weighting methods for improvement of classification results, novel approaches of dimensionality reduction and machine learning for computational effectiveness improvement. This chapter is related to the second and third aims of the thesis.

Finally, Chapter 6, describes the main conclusions derived from the thesis work, provides a summary of the thesis main contributions, and outlines open issues for future work.

2. Background and State-of-the-Art

2.1. Spoken Dialogue Systems

Spoken Dialogue Systems (SDSs) are systems with human-machine interfaces based on speech communication (Jokinen and McTear, 2009).

Commercial applications of SDS historically:

- Telephone-based information services (since 2000)
- Natural language call routing (since 2005)
- Voice assistants in smart phones („Apple Siri“, „Microsoft Cortana“, “Google Now”) (since 2010).

Future application areas for SDS:

- Social robotics (Fong et al., 2003)
- Information extractor (active listener) (Meguro et al., 2010)
- Mediating interaction (Bos et al., 2003).

Figure 2.1 shows the architecture of a standard SDS (Dybkjær and Minker, 2008).

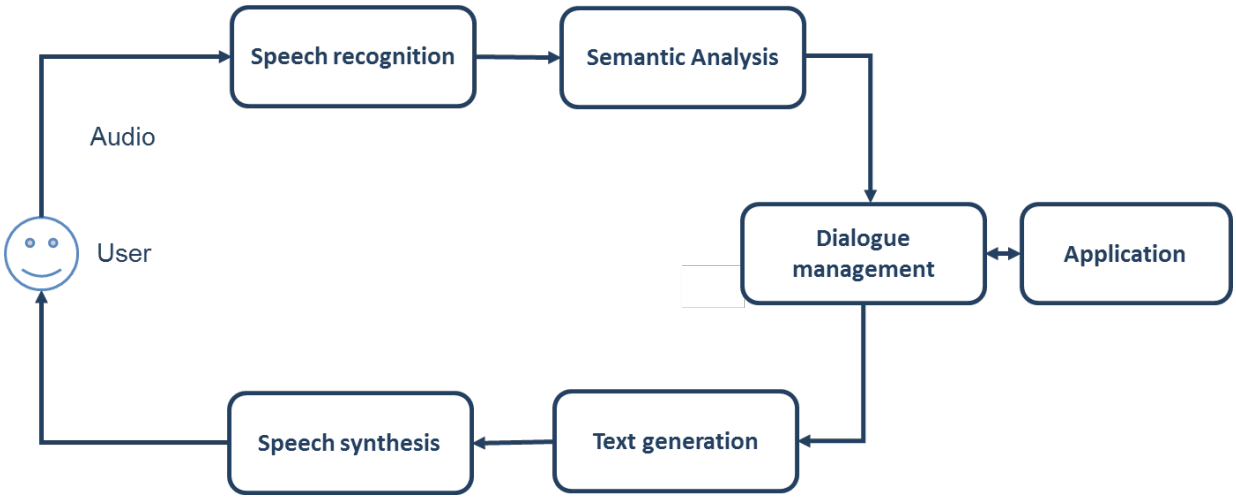

Figure 2.1 – The standard architecture of a SDS

First, input acoustic vectors generated from the speech signal are processed by an Automatic Speech Recognizer (ASR) (Potamianos et al. 2004), which obtains a raw text transcription of the input user utterance. Next, the transcribed text comes to a Natural Language Understanding block (semantic analysis) which extracts the meaning of the call in form of an appropriate semantic structure. Then this semantic representation is

processed by the dialogue manager which also communicates directly with an external application, namely a database interface. The dialogue manager controls the overall progress of interaction according to the task completion. During this process, the user may be asked for confirmations, disambiguation, additional information, etc. Finally, the result of the interaction is given to the user in form of speech by text-to-speech synthesis or pre-recorded prompts.

It is possible to improve a standard SDS with the addition of the following elements:

- User state recognition: emotion recognition (Sidorov et al., 2014a), verbal intelligence (VI) recognition (Nothdurft et al., 2014), speaker identification (Sidorov et al., 2013), gender and age identification (Harb and Chen, 2003; Schmitt et al., 2010), and other characteristics such as voice alcohol intoxication detection (Ultes et al., 2011). This information is useful for effective and user-adaptive dialog management.

- Topic identification of user utterances; it provides multi-domain spoken dialogue systems.

These elements could bring advantages and improve the human-machine interaction quality. These are ones of the techniques that add knowledge to an SDS and that ultimately will render an SDS more natural and user-friendly in the future (Schmitt and Minker, 2012). The architecture of such an extended multi-domain and user-adaptive SDS is illustrated in Figure 2.2.

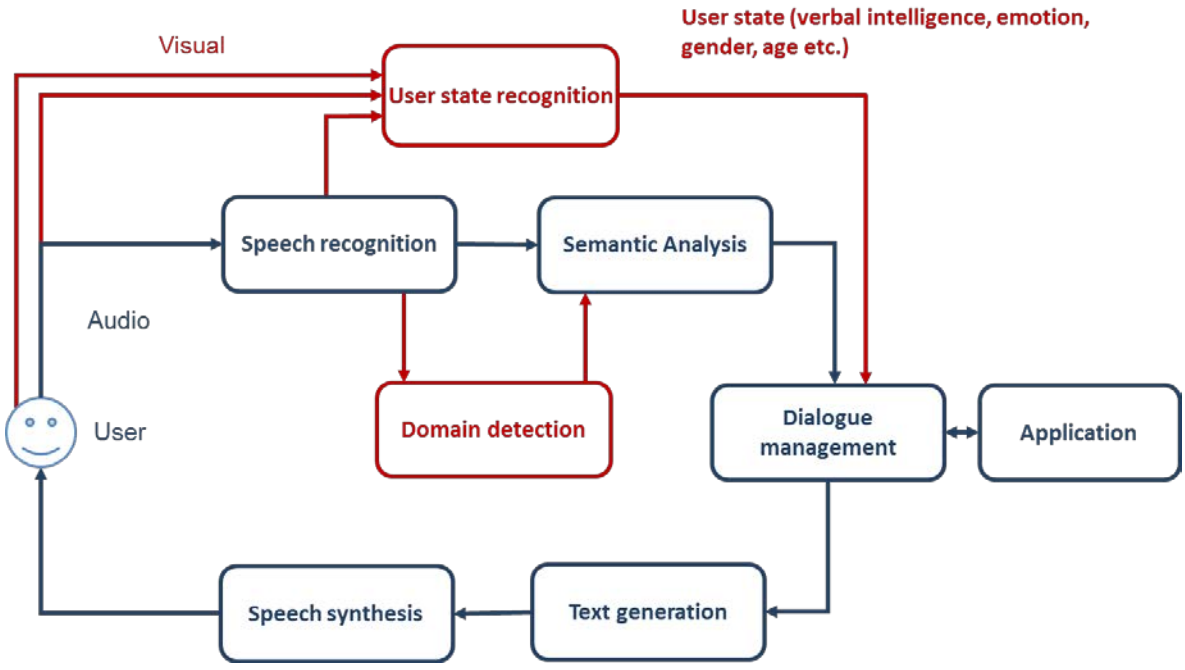

Figure 2.2 – The architecture of a multi-domain user-adaptive SDS

The details of the User State Recognition block are presented in Figure 2.3.

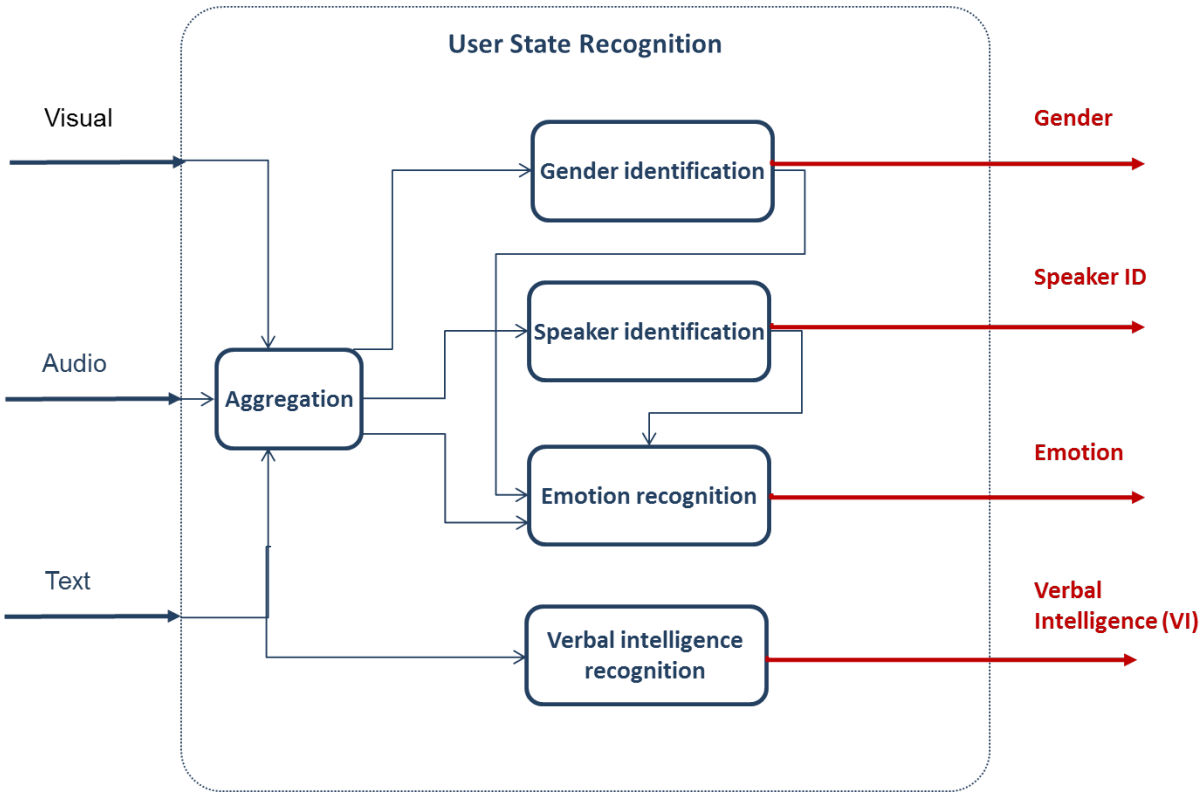

Figure 2.3 – The scheme of the User State Recognition block

2.2. Text Classification Problems Applied for Spoken Dialogue Systems

The development of extended multi-domain user-adaptive spoken dialogue systems (Figure 2.2) is related to complex analyses of different types of data: acoustic signals, video streams, text information after automatic speech recognition (ASR). Text analysis problems may be categorized into two different directions: semantic analysis based on ontologies, grammatical processing etc. (Fensel, 2001; Noy, 2004; Goddard, 2011) and text classification. The research presented in this thesis is about text classification for SDS design.

The following three text classification problems are important for extended multi-domain user-adaptive SDS design:

- *Topic Identification of User Utterances***.** This problem belongs to the domain detection block for multi-domain SDS. Nowadays, multi-domain spoken dialogue

systems are a new step in the technologies of human-machine interfaces (Lee et al., 2009; Komatani et al., 2009). Multi-domain SDS includes different specific semantic models for each problem domain. Therefore, it is necessary to detect a domain of a user utterance before detailed semantic analysis. Domain detection is based on two inputs (Komatani et al., 2009): the previous domain history and the domain in which the speech recognition results of the current user utterance can be accepted with the highest recognition score. The last stage can be formulated as a text classification problem based on the ”bag-of-words” model (Zhang et al., 2010). It can be more effective to design domain-related semantic models after domain detection based on text classification than to design a complicated semantic model for all domains (Figure 2.4).

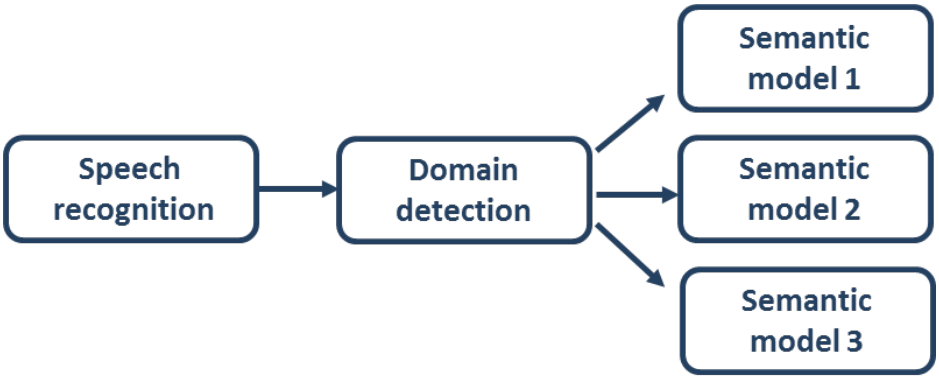

Figure 2.4 – Topic identification of user utterances for multi-domain SDSs

- *User Verbal Intelligence Recognition***.** This problem belongs to the user state recognition block**.** In general, verbal intelligence is defined as the ability to use language for accomplishing certain goals (Goethals et al., 2004; Cianciolo and Sternberg, 2004). This can be interpreted in a way that verbal intelligence is the ability to analyze information and to solve problems using language-based reasoning. Analyzing verbal intelligence is related to checking the readability of written text, which is for example done by Burstein et al. (2003) or Elliot (2003). These systems automatically assess the linguistic accessibility, denoting parts of text, which may cause difficulties while reading. However, due to the fact that spoken language often does not contain grammatically correct sentences (e.g. slang, dialectical expressions, or ordinary language) these methods are not applicable in SDSs.

For spoken language it should be possible to find out which language peculiarities may reflect the verbal intelligence of speakers. The automatic estimation of users’ verbal

intelligence may help spoken language dialogue systems to more effectively control the flow of the dialogues, engage the users in an interaction, be more attentive to human needs and preferences and, as a result, be more helpful and user-friendly. For training machine learning algorithms, we need to know a maximum number of language features that reflect users' verbal intelligence. It is possible to investigate to which extent the vocabulary of users reflect their levels of verbal intelligence when they all describe the same event and explain their thoughts and feelings about it (Zablotskaya et al., 2012; Fernández-Martínez et al., 2012). Therefore, verbal intelligence recognition can be formulated as a text classification problem.

- *Text-based user emotion recognition***.** This problem belongs to the user state recognition block**.**

Automatic recognition of human emotions based on the speech signal is in the focus of research groups all over the world. One of the pilot experiments which deals with speech-based emotion recognition has been presented by Kwon et al. (2003). The authors compare the emotion recognition performance of various classifiers: support vector machines, linear discriminant analysis, quadratic discriminant analysis and hidden Markov models on SUSAS and AIBO databases of emotional speech. The following set of speech signal features has been used in the study: pitch, log energy, formant, mel-band energies, and mel frequency cepstral coefficients (MFCCs). The authors have managed to achieve the highest value of accuracy (70.1% and 42.3% on the databases, correspondingly) using Gaussian support vector machines (Schölkopf et al., 1997).

Gharavian et al. (2012) highlight the importance of feature selection for emotion recognition using the fast correlation-based filter feature selection method. A fuzzy neural network (Carpenter et al., 1992) was used as an algorithm for emotion modeling. The authors achieve an accuracy of over 87.52% for emotion recognition on the FARSDAT speech corpus (Bijankhan et al., 1994).

One of the most significant and important emotions for effective and user-adaptive human-machine dialog management is user anger. Therefore, Polzehl et al. (2011) only focused on the emotion anger. The authors analyzed two different anger corpora of German and American English to determine the optimal feature set for anger recognition. The German database contains 21 hours of records from a German Interactive Voice Response (IVR) portal offering assistance troubleshooting. For each utterance, three annotators assigned one of the following labels: not angry, not sure, slightly angry, clear anger, clear rage, and garbage. Garbage-marked utterances are non-applicable, e.g.,

contain silence or critical noise. The English corpus originates from an US-American IVR portal which is capable of fixing Internet-related problems. Three labelers divided the corpus into angry, annoyed, and non-angry turns. A total of 1,450 acoustic features and their statistical description (e.g., means, moments of first to fourth order, and the standard deviation) have been extracted from the speech signal. The features are divided into seven general groups: pitch, loudness, MFCC, spectral, formants, intensity, and other (e.g., harmonics-to-noise). Analyzing each feature group separately, the authors achieved in a baseline approach without further feature selection a maximal *F*-score of 68.6 with 612 MFCC-based features of the German corpus and a maximal *F*-score of 73.5 with 171 intensity-based features of the English corpus.

Basic approaches for automatic emotion recognition regard the problem independently of the speaker. However, while the basic emotions are shared between all people and cultures (Scherer, 1997), humans have a fine-tuned emotional model of people they know allowing for recognizing their emotions more accurately. Moreover, speaker-specific models have shown to improve speech recognition as well (Leggetter and Woodland, 1995). Therefore, it is possible to add speaker-specific information to the emotion recognition process (Sidorov et al., 2014a; Sidorov et al., 2014b).

Dialogues may not only consist of speech, but also of a visual representation. Hence, an analysis of captured speaker images or even video recordings may also improve the performance of individual tasks and the whole SDS in general. In this case we deal with multi-modal emotion recognition (Sidorov and Minker, 2014c; Sidorov and Minker, 2014d; Sidorov et al., 2014e).

It may be possible to extract additional useful information for emotion recognition based on text contents of user utterances. We assume that some words or phrases may definitely help to recognize a correct emotion. Therefore, in our work we propose text-based emotion recognition as an additional modality for multi-modal emotion recognition; it means a model-level fusion of emotion classifiers based on acoustic, video, and text information. Text-based emotion recognition can be also formulated as text classification problem.

Therefore, three different text classification problems were formulated that are important in the field of advanced spoken dialogue system design: topic identification of user utterances, user verbal intelligence recognition, and text-based user emotion recognition.

2.3. Text Classification Stages

After speech has been converted into text form, text classification can be performed. Text classification can be considered to be a main part of natural language understanding, where there is a set of predefined categories and the task is to automatically assign new documents to one of these categories.

In general, documents are not represented as sequences of symbols. The most popular model for text classification is the vector space model (Sebastiani, 2002). In this case text categorization may be considered as a machine learning problem. Complexity of text categorization using a vector space model is compounded by the need to extract the numerical data from text information before applying machine learning methods. The method of text preprocessing and text representation influences the results that are obtained even with the same classification algorithms. Therefore text categorization consists of two main parts: text preprocessing and classification using numerical data.

Text preprocessing comprises the following stages that are also presented in Figure 2.5:

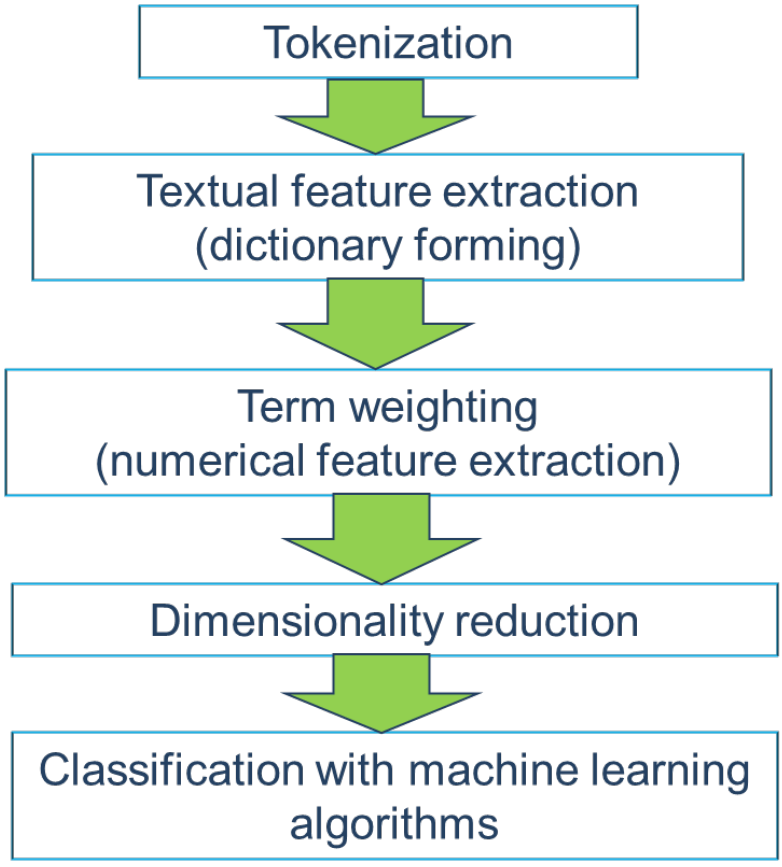

Figure 2.5 – Text classification: learning stages

- *Tokenization*. Tokenizing is the process of converting text sentences into separate words. It means cutting down the sentence into individual parts called tokens. It gives out tokens or terms and deletes language based characters e.g. punctuations, special characters, transforming capital letters to lowercase etc. As a result, we consider each document as an individual object for classification. All words in the document are presented in one single string.

- *Textual feature extraction (dictionary forming).* Textual feature extraction is based on raw preprocessing of the documents. It means that each document for classification should be characterized by a vector of terms (textual features). The total set of unique terms from the training data form the dictionary. This process can include additional procedures such as stop-words filtering (Fox, 1989) and stemming (Porter, 2001). A stop-word list contains pronouns, prepositions, articles and other words that usually have no importance for the classification. Using stemming it is possible to join different forms of the same word into one textual feature.

- *Term weighting (numerical feature extraction).* The next stage is the numerical feature extraction based on term weighting. In order to perform the classification task terms (textual features) need to be converted into numbers since the majority of classifiers work with numerical features. The process of converting text features into meaningful numbers is known as term weighting. In our study, we use a ”bag-of-words” model for term weighting, in which the word order is ignored. There exist different unsupervised and supervised term weighting methods. The most well-known unsupervised term weighting method is TF-IDF (Salton and Buckley, 1988). The following supervised term weighting methods are also considered in our research: Gain Ratio (GR) (Debole and Sebastiani, 2004), Confident Weights (CW) (Soucy and Mineau, 2005), Term Second Moment (TM2) (Xu and Li, 2007), Relevance Frequency (RF) (Lan et al., 2009), Term Relevance Ratio (TRR) (Ko, 2012), and Novel Term Weighting (NTW) (Gasanova et al., 2014a); these methods involve information about the classes of the documents.

- *Dimensionality reduction.* As a rule, the dimensionality for text classification problems is high even after stop-words filtering and stemming. Therfore, the classification may be inappropriately time-consuming, especially for real-time systems such as spoken dialogue systems. Consequently, the next stage of preprocessing is the dimensionality reduction based on numerical features performed by feature selection or feature transformation. The feature selection process involves the use of those features or terms in each document which are useful for text classification later on. The main aim in

this step is to remove useless features from the dataset. There are different ways to extract useful features from a raw text in order to establish a dictionary of words or in other words a feature set which will be useful for the text classification task. In many cases this step improves the classification effectiveness since terms with no meaning in the context of the classification task have been removed. Feature transformation results in a transformed and reduced feature set.

It should be noticed that a dimensionality reduction can be performed on the stage of textual feature extraction. Stemming and stop-words filtering are examples of linguistic dimensionality reduction methods.

After text preprocessing, classification is performed with standard machine learning algorithms. The *k*-NN algorithm and the SVM-based algorithms are popular techniques for text classification.

Term weighting methods are described in Section 2.4, dimensionality reduction approaches for text classification are presented in Section 2.5, and classification algorithms with the focus on the application to the text classification are described in Section 2.6.

2.4. Term weighting methods

2.4.1. Introduction

Term weighting is a part of text preprocessing, where the importance values for the extracted features are measured.

The simplest text representation is a binary vector or Boolean model. The Boolean model is the simplest model in which each document's term is checked with the dictionary set. A term which does not exist in a particular document but which exists in the vocabulary set is associated with the number '0' and a term which exists in a particular document as well as in the dictionary set is assigned the number '1'. The example of such a model is described below.

Suppose we have three documents in a dataset and after feature extraction they contain the following words:

doc. 1 (d1) = {training, football, exercise, enjoy, training}
doc. 2 (d2) = {football, sport, player, sport}
doc. 3 (d3) = {enjoy, player, exercise, football}

Dataset (D) = [d1, d2, d3]

Dictionary set = {enjoy, player, exercise, football, sport, training}

The Boolean model representation for the above mentioned example is presented in Table 2.1. The dimensionality of the feature space is the number of unique words appearing in the training set (dictionary size).

Table 2.1

Boolean model example

	Dictionary Set					
	enjoy	player	exercise	football	sport	training
d1	0	0	1	1	0	1
d2	0	1	0	1	1	0
d3	1	1	1	1	0	0

This model is very simple and does not require significant computational resources, however it has some drawbacks. The most important drawback is that every word is given the same importance by associating it with number '1' if it appears in the document. This problem can be especially significant in large dataset having large documents and a large or redundant dictionary set. Another problem is the fact of ignoring repetition numbers of terms in one document; the number of occurrences of terms may be very important for effective classification.

It can be seen clearly that the Boolean model is insufficient for large datasets and we need to use a different model to assign weights to words. These values should not be constant values (1 or 0). It seems to be necessary to involve frequencies of terms into the term weighting process in the current document and in the whole corpus. Therefore, term weighting methods that involve such information should be more appropriate techniques for text representation.

There are many weighting schemas through which particular weights may be assigned to words in each document. The term weighting methods can be roughly divided into two groups:

- *Unsupervised term weighting*. These methods do not use the information about class labels of the documents introduced in the training set. The weights are calculated on the basis of only statistics on the term occurrence in documents from the whole collection.

- *Supervised term weighting*. The term weights are calculated using statistical information about the class labels.

In general, term weighting consists of two parts: the one that corresponds to statistics of term occurrences in the given document (binary, term frequency (TF), random walk (Hassan et al., 2007) etc.) and the part that estimates statistics of term occurrences in the whole collection (IDF modifications). As a rule, term weighting is a multiplication of these two parts.

In our study the first part of term weighting (term frequency, TF) is fixed for all considered term weighting methods and is calculated as following:

$$TF_{ij} = \log(tf_{ij} + 1);\ tf_{ij} = \frac{n_{ij}}{N_i}, \qquad (2.1)$$

where n_{ij} is the number of times the i^{th} word occurs in the j^{th} document, N_j is the document size (number of words in the document).

The second part of the term weighting is calculated once for each word from the dictionary and does not depend on an utterance for classification. We consider seven different methods for the calculation of the second part of term weighting: IDF, GR, CW, TM2, RF, TRR, and NTW.

The completely different term weighting concepts such as random walk (Hassan et al., 2007) or ConceprFreq (Song et al., 2004) are also described in this section.

2.4.2. Inverse Document Frequency

Inverse Document Frequency (IDF) is a well-known unsupervised term weighting method which was proposed in (Salton and Buckley, 1988). There are some modifications of IDF:

1) IDF 1:

$$idf_i = \log\frac{|D|}{n_i}, \qquad (2.2)$$

where $|D|$ is the number of documents in the training set and n_i is the number of documents that have the i^{th} word.

2) IDF 2: The formula is given by equation (2.2) except n_i is calculated as the number of times the i^{th} word appears in all documents from the training set.

3) IDF 3:

$$idf_i = \left(\frac{|D|}{n_i} \right)^{\alpha}, \alpha \in (0;1), \tag{2.3}$$

where n_i is calculated as in IDF 1 and α is the parameter.

4) IDF 4: The formula is given by equation (2.3) is calculated except n_i.

In our research we use the most popular modification IDF 1.

2.4.3. Gain Ratio

Gain Ratio (GR) is mainly used in term selection (Yang and Pedersen, 1997). However, in (Debole and Sebastiani, 2004) it was shown that it could also be used for weighting terms, since its value reflects the importance of a term. The definition of GR is as follows:

$$GR(t_i, c_j) = \frac{\sum_{c \in \{c_j, \overline{c_j}\}} \sum_{t \in \{t_j, \overline{t_j}\}} P(t,c) \cdot \log \frac{P(t,c)}{P(t) \cdot P(c)}}{-\sum_{c \in \{c_j, \overline{c_j}\}} P(c) \cdot \log P(c)}, \tag{2.4}$$

where $P(t, c)$ is the estimation of the probability that a document contains the term t and belongs to the category c; $P(t)$ is the estimation of the probability that a document contains the term t, and $P(c)$ is the estimation of the probability that a document belongs to category c.

Then, the weight of the term t_i is the maximal value between all categories as follows:

$$GR(t_i) = \max_{c_j \in C} GR(t_i, c_j), \tag{2.5}$$

where C is a set of all classes.

2.4.4. Confident Weights

The Confident Weights method (CW) is a supervised term weighting that involves information about classes which correspond to documents. This approach has been proposed by Soucy and Mineau (2005). The main idea of the method is that the term t has a non-zero weight in the class c only if the frequency of the term t in documents of the class c is greater than the frequency of the term t in all other classes.

Firstly, the proportion of documents containing term t is defined as the Wilson proportion estimate p (x; n) (Wilson, 1927) by the following equation:

$$p(x;n) = \frac{x + 0.5z_{\alpha/2}^2}{n + z_{\alpha/2}^2}, \tag{2.6}$$

where x is the number of documents containing the term t in the given corpus, n is the number of documents in the corpus and $\Phi(z_{\alpha/2}) = \alpha/2$, where Φ is the t-distribution (Student's law) when $n < 30$ and the normal distribution when $n \geq 30$.

In this work $\alpha = 0.95$ and $0.5z_{\alpha/2}^2 = 1.96$ (as recommended by the authors of CW). For each term t and each class c two functions $p+$ (x; n) and $p-$ (x; n) are calculated. For $p+$ (x; n) x is the number of vectors which belong to the class c and have term t; n is the number of documents which belong to the class c. For $p-$ (x; n) x is number of vectors which have the term t but do not belong to the class c; n is the number of documents which do not belong to the class c.

The confidence interval $\left(\underline{p}; \overline{p}\right)$ at 95% is calculated using equations (2.7):

$$\underline{p} = p - 0.5z_{\alpha/2}^2\sqrt{\frac{p(1-p)}{n + z_{\alpha/2}^2}}, \quad \overline{p} = p + 0.5z_{\alpha/2}^2\sqrt{\frac{p(1-p)}{n + z_{\alpha/2}^2}}. \tag{2.7}$$

The strength of the term t in the category c is defined as equation (2.8):

$$str(t,c) = \begin{cases} \log_2\left(\dfrac{2\underline{p+}}{\underline{p+} + \overline{p-}}\right), if\left(\underline{p+} > \overline{p-}\right), \\ 0, otherwise \end{cases} \tag{2.8}$$

The maximum strength (*Maxstr*) of the term t is calculated as follows:

$$Maxstr(t) = \left(\max_{c \in C} str(t,c)\right)^2, \tag{2.9}$$

where C is the set of all classes.

The CW method uses *Maxstr* as an analogue of IDF:

$$ConfWeight_{ij} = \log(tf_{ij} + 1) \cdot Maxstr(i), \tag{2.10}$$

where i is an index of a term, j is an index of a document.

The numerical experiments conducted in (Soucy and Mineau, 2005) have shown that the CW method outperforms the Gain Ratio (GR) and TF-IDF with SVM and k-NN as classification methods on three benchmark corpora. However, this method is more computationally demanding than the TF-IDF method.

2.4.5. Term Second Moment

Term Second Moment (TM2) was proposed by Xu and Li (2007) as a supervised term weighting method.

Let $P(c_j \mid t)$ be the empiric estimation of the probability that a document belongs to the category c_j with the condition that the document contains the term t and belongs to the category c; $P(c_j)$ is the empiric estimation of the probability that a document belongs to the category c without any conditions. The idea is the following: the more $P(c_j \mid t)$ is different from $P(c_j)$ the more important the term t_i is. Therefore, we can calculate the term weight as follows:

$$TM(t_i) = \sum_{j=1}^{|C|} \left(P(c_j \mid t) - P(c_j)\right)^2 . \quad (2.11)$$

2.4.6. Relevance Frequency

Another supervised term weighting method has been proposed by Lan et al. (2009). The idea of the Relevance Frequency (RF) method is that the higher the difference between the concentration of the term with high frequency in the positive category and its concentration in the negative category, the bigger contribution it makes towards the detection of this category.

The RF value is calculated as the following:

$$rf(t_i, c_j) = \log_2\left(2 + \frac{a_j}{\max\{1, \overline{a_j}\}}\right), \quad (2.12)$$

$$rf(t_i) = \max_{c_j \in C} rf(t_i, c_j), \quad (2.13)$$

where a_j is the number of documents of the category c_j which contain the term t_i and $\overline{a_j}$ is the number of documents of all the other categories which also contain this term.

The experimental results using linear SVM and *k*-NN algorithms which have been conducted in (Lan et al., 2009) have shown that TF.RF outperforms other term weighting schemes in all experiments.

2.4.7. Term Relevance Ratio

The Term Relevance Ratio (TRR) method (Ko, 2012) uses *tf* weights (see equation 2.1) and it is calculated as the following:

$$TRR(t_i, c_j) = \log_2\left(2 + \frac{P(t_i \mid c_j)}{P(t_i \mid \overline{c_j})}\right), \tag{2.14}$$

$$P(t_i \mid c) = \frac{\sum_{k=1}^{|T_c|} tf_{ik}}{\sum_{l=1}^{|V|}\sum_{k=1}^{|T_c|} tf_{lk}}, \tag{2.15}$$

$$TRR(t_i) = \max_{c_j \in C} TRR(t_i, c_j), \tag{2.16}$$

where c_j is a class of the document, $\overline{c_j}$ is all of other classes of c_j, V is the vocabulary of the training data and T_c is the document set of the class c.

Experimental results (Ko, 2012) show good performance of the introduced scheme in comparison with other term weighting techniques using *k*-NN and SVM on the two benchmark databases.

2.4.8. Novel Term Weighting

The Novel Term Weighting (NTW) method was proposed in (Gasanova et al., 2013; Gasanova et al., 2014a). The term weight is calculated using a modified formula of fuzzy rule relevance estimation for fuzzy classifiers (Ishibuchi et al., 1999). The membership function has been replaced by word frequency in the current class. The details of the procedure are the following:

Let L be the number of classes; n_i is the number of documents which belong to the i^{th} class; N_{ij} is the number of occurrences of the j^{th} word in all articles from the i^{th} class. $T_{ij} = N_{ij} / n_i$ is the relative frequency of occurrences of the j^{th} word in the i^{th} class; $R_j = \max_i T_{ij}$; $S_j = \arg(\max_i T_{ij})$ is the class which we assign to the j^{th} word. The term relevance C_j is calculated by the following:

$$C_j = \frac{1}{\sum_{i=1}^{L} T_{ij}}\left(R_j - \frac{1}{L-1}\sum_{\substack{i=1 \\ i \neq S_j}}^{L} T_{ij}\right). \tag{2.17}$$

NTW is calculated for each word in the document and it is clear that the NTW values will be higher for a term occurrence frequently in one classes than if it appears in all other classes. The maximum value of NTW equals 1; in this situation the term occurs only in documents of one class. The minimum value of NTW equals 0; in this situation the frequencies of term occurring in all classes are exactly equal.

2.4.9. Random Walk

A completely different concept for term weighting is the so-called Random Walk (Hassan et al., 2007).

This approach links terms based on their occurrences in the document within a specific window size. Term weighting is based on graph theory, in which nodes are considered as terms and edges connecting different nodes link terms based on their occurrences in the document. Each connection casts a vote for the destination node. The more votes gained by the destination node, the more importance or in other words the higher weight it obtains for that term. The importance of a node vote depends on the source node casting the vote. There are several random walk algorithms which determine the score for each node. The PageRank algorithm (Hassan et al., 2007) proves to be the one which performs very well in text classification problems. This method has been implemented by Hassan et al. (2007) in their paper for improved text classification using the Random Walk Term Weighting algorithm.

Suppose we have a document for which we want to find a score for a term node ‘bill’ via a random walk. Figure 2.7 shows the graph for this example.

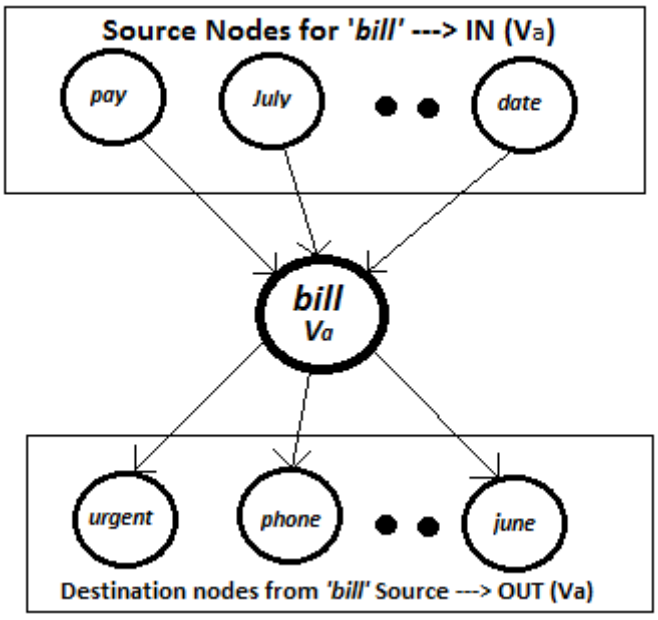

Figure 2.7 – An example graph for term weighting based on random walk

(Taken from: Gasanova, Tatiana (2015): Novel methods for text preprocessing and classification. Open Access Repositorium der Universität Ulm. Dissertation; Text; http://dx.doi.org/10.18725/OPARU-3242)

The random walk term weighting algorithm is the following. Let $G = (V; E)$ be a graph, where V is a set of all vertices and E is a set of all edges; In(V_a) is a subset of all vertices that point to the vertex V_a (predecessors), and Out(V_a) is a subset of all vertices that the vertex V_a points to (successors). The random walk value $S(V_a)$ associated with the vertex V_a of the term a is calculated using a following recursive function:

$$S(V_a) = (1-d) + d \cdot \sum_{V_b \in In(V_a)} \frac{S(V_b)}{|Out(V_b)|}, \tag{2.18}$$

where d is a parameter which can be set between 0 and 1 (usually $d = 0.85$).

2.4.10. ConceptFreq

This term weighting method has been proposed by Song et al. (2004). It is based on the lexical chain and the sentence extraction algorithm that uses it. ConceptFreq is calculated as follows:

$$ChainFreq(w_{ij}, c_j) = WordConnectivity(w_{ij}) \times ChainConnectivity(c_j), \tag{2.19}$$

where *WordConnectivity* (w_{ij}) is given by:

$$WordConnectivity(w_{ij}) = Frequency(w_{ij}) \cdot \frac{r(w_{ij}, c_j) + \alpha}{\sum_{j=1}^{J} \sum_{k=1}^{N} r(w_{kj}, c_j)}, \tag{2.20}$$

and *ChainConnectivity* (c_j) is given by:

$$ChainConnectivity(c_j) = \frac{\sum_{k=1}^{N} r(w_{kj}, c_j)}{|c_j|}, \tag{2.21}$$

where $r(w_{ij}$; $c_j)$ is the number of relations that have the term w_{ij} in the chain c_j , $|\ c_j\ |$ is the number of unique words in the chain c_j and α is a constant.

Authors in (Song et al., 2004) have shown that the proposed scheme outperformed the standard TF-IDF, however the problem of semantic relations between terms remains a complex task.

2.5. Dimensionality Reduction Methods

2.5.1. Introduction

Text collections which contain millions of unique terms are quite common. Thus, in order to more efficiently apply existing classification algorithms and to achieve better

performance, a dimensionality reduction of the feature space is widely used when applying machine learning methods to text categorization. Since most classification algorithms cannot handle such high dimensionality of text data, techniques which reduce the term set size are particularly useful for text classification for both efficiency and efficacy.

Dimensionality reduction methods for text classification can be classified in two different ways.

At first, all dimensionality reduction methods can be divided into feature selection and feature transformation methods.

- *Feature selection (term selection)*: *Ts* (a selected feature set) is a subset of *T* (a predefined feature set). These methods usually consider the selection of only the terms that occur in the highest number of documents, or the selection of terms depending on the observation of information-theoretic functions.

- *Feature transformation:* Feature transformation (term extraction): the terms in *Tp* (a transformed feature set) are not of the same type as the terms in *T* (a predefined feature set); the features in *Tp* are obtained by combinations or transformations of the original features. These methods generate from the original a set of synthetic features.

The difference between feature selection and feature transformation is illustrated in Figures 2.7 and 2.8.

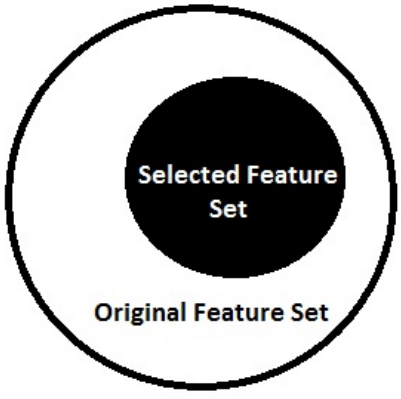

Figure 2.7. – Feature selection

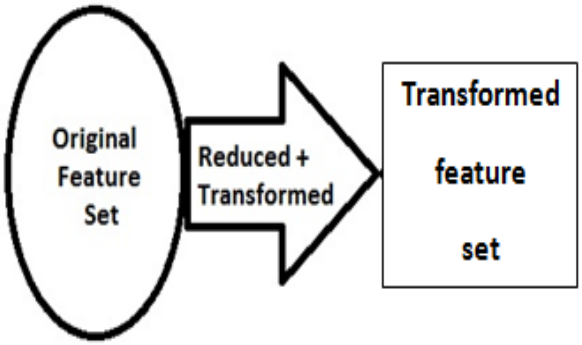

Figure 2.8. – Feature transformation

Additionally, dimensionality reduction methods for text classification can be divided into linguistic and numerical methods.

- *Linguistic dimensionality reduction methods* use specific linguistic information for feature selection or feature transformation. As a rule, linguistic dimensionality reduction methods are language-dependent; it means that a special procedure must be constrained

for each language. Linguistic dimensionality reduction methods are applied in the stage of textual feature extraction.

- *Numerical dimensionality reduction methods* for text classification are based on feature selection or feature transformation after term weighting (numerical feature extraction). These methods are language-independent.

A classification of dimensionality reduction methods for text classification is presented in Table 2.2.

Table 2.2

Dimensionality reduction methods for text classification

	Feature selection	**Feature transformation**
Linguistic methods	„Stop-words“ filtering	Stemming Term clustering based on dictionaries of synonyms
Numerical methods	Weight-based term filtering Wrappers (e.g. GA-based wrappers)	Principal Component Analysis Latent Semantic Indexing Weight-based term clustering

2.5.2. Linguistic Dimensionality Reduction Methods

- *“Stop-words” filtering.* “Stop-words” filtering is based on the manually created lists of non-informative words (for example, the SMART stop list by Blanchard (2007) or the stop list by Dragut et al. (2009)) that are deleted from the original feature set (a dictionary). Stop words are irrelevant for information extraction, clustering or classification, since they do not contain any semantic information related to the document category. Pronouns and prepositions are examples of such stop words. For instance, many frequently used words in English are basically useless for the text classification task e.g. „the”, “of”, “to” etc. an estimate count for such words is around 400 to 500 in English language. These types of words are called stop words which need to be removed from a dataset as, in principle, they carry no information useful for the classification task. It is important to get rid of these words e.g. prepositions, conjunctions etc. as commonly they take 20-30% of the total datasets for English. Therefore, less time will be required for further processing since fewer words need to be taken care of for the classification task.

For natural language documents (usually documents which have been transcribed from speech automatically by an ASR) the sounds which are common for a spontaneous speech (eh, ehm, uh etc.) are also considered as stop-words.

Stop-words lists are obviously different for every language.

- *Stemming.* Another way to decrease the dictionary size of the text classification problem is to find the common root or stem of many forms of the same word and replace them with their root. The goal of stemming is to transform different forms of the word to its canonical form: stem or lemma. It eliminates the morphological diversity which is provided by the suffixes and prefixes as discussed in (Porter, 1980; Schmid, 1994). There are several approaches to stemming for example lookup algorithms, suffix-stripping algorithms, lemmatization algorithms, *n*-gram analysis and a hybrid approach. An example of stemming is presented in Table 2.3.

Table 2.3

An example of stemming

User	Thinking
Using	Thought
Used	Thoughtful
Users	Thinks
Use	**Think**

There are two error measures in stemming algorithms: overstemming and understemming. In the case of overstemming, the morphologically related words, which are used in different context with different meaning, are considered to be the same stem. In this situation the relevance of the obtained feature set will be worse. Understemming is the case where two words should be stemmed to the same base but the stemming algorithm has not stemmed them to one root.

- *Term clustering based on dictionaries of synonyms.* Term clustering aims to combine terms with synonymous or near synonymous meaning into groups (clusters). This set of clusters (or centroids or a prototype) is used as a term set (feature space). The idea of term clustering was proposed by Brown et al. (1992) and applied in many different fields including speech recognition, named entity tagging, machine translation, query expansion, text categorization and word sense disambiguation. The idea of using a class-based language model by applying term clustering, proposed by Momtazi and Klakow (2009), has been found to be effective in overcoming the problems of data sparsity, exact matching using small text segments.

2.5.3. Numerical Feature Selection Methods

Feature selection approaches attempt to find such a subset of the terms initially extracted from the train sample that will lead to the better document representation and higher performance of the classification algorithm used to solve the task.

Yang and Pedersen (1997) have shown that term selection can increase the algorithm's performance by up to 5% depending on how many terms have been excluded, the selection criterion that has been applied and the classifier.

Moulinier and Ganascia (1996) have proposed a method of term selection which tunes to the classifier that is used. An obvious advantage of this approach is that the document representation is tuned with the classifier and in the reduced term set there may be different numbers of terms for detecting each category (some categories are more separable than others). The problem is that the method is time consuming because the term set size is usually very large.

Feature selection algorithms can be divided by:

- *Filters.* These methods evaluate each term independently from the classifier according to the chosen weighting technique, rank the features after evaluation and take the subset with the highest weights.

- *Wrappers.* In contrast to the previous approach these algorithms depend on the selected classifier, they evaluate subsets of the initial term set with the chosen classifier. The subset which provides the best performance is selected. Wrappers are more reliable than filters because the classification algorithm influences the selection of the term subset. Since it is a time consuming process heuristic algorithms are often used to find the optimal subset for the selected classifier: genetic algorithm, greedy stepwise, best first or random search.

The filtering approach, in accordance with results in (John et al., 1994), is usually a computationally easier alternative than the wrapper selectors; filters remove terms with low measure of importance. There are some ways to measure this importance of the term. The most natural approach for the term selection is a use of the term weight as a term importance measure. In this case we can perform sorting terms by their weights. After sorting, a particular percentage of terms or features with the highest weights are selected e.g. 50%, 70% etc. This way of selecting is unsupervised in which no information regarding the labels or classes has been taken into account. However, a certain percentage of term selection can be performed for each class independently if the assignment of

terms to classes is performed; in this case we have supervised feature selection. Another simple technique for filtering is to set constraint values for term weights and only select those terms which have weights greater than a constraint value. In this technique many constraints values can be set and classification results can be observed and compared for different constraints results.

Yang and Pedersen (1997) have shown a filtering term reduction where the words which appear more often are kept in the reduced term set. Term set reduced ten times will produce the same classification performance as the initial set, and hundred times reduction will lead only to a small loss of effectiveness. This result seems to show that the words with the highest frequency are the most useful for text classification. However, there is somehow a contradictive result by Salton and Buckley (1988) that shows that the words with lowest frequency carry the most information about document category. It may be explained by the fact that results depend on the document sizes and the dictionary size. As a rule, feature selection with deleting a lot of frequent terms is very effective for large documents with a large or redundant dictionary (Gabrilovich and Markovitch, 2004) and can be not effective for small documents such as user utterances in speech-based text classification problems (Sergienko et al., 2016).

This contradiction may be solved with using different techniques for term importance measure in the filtering stage and for term weighting in the classification stage. There are specific methods to weight the term importance in the corpus for filtering-based feature selection. A common approach is to use an information-theoretic function, which has been used to weight the terms. Many information-theoretic functions have been used in the literature. A detailed list is provided in Sebastiani (2002), which is repeated in Table 2.3. In this table t_k is the current term for importance measurement, $\bar{t}_k$ is the set of all other terms from the dictionary, c_i is the positive category, $\bar{c}_i$ is the negative category, $|T|$ is the size of the dictionary, $P(*)$ is the empirical estimation of the probability to have a document with the conditions (*). These functions attempt to capture the intuition that the most important terms for the classification performance on the category are the ones distributed most differently in the sets of positive and negative examples for the category.

Karabulut et al. (2012) have shown how the chosen feature selection technique affects the classification performance of different algorithms: Naive Bayes, Multilayer

Perceptron, and J48 decision tree are compared. The most sensitive classifier was the multilayer perceptron.

Table 2.3

Information-theoretic functions for feature selection based on filtering

Name	Authors	Formula
DIA assosciation factor	Fuhr et al. (1991)	$P(c_i \mid t_k)$
Information gain (gain ratio)	Dumais et al. (1998) Joachims (1998) Yang and Pedersen (1997) Yang and Liu (1999)	$\sum_{c\in\{c_i,\bar{c}_i\}}\sum_{t\in\{t_k,\bar{t}_k\}} P(t,c)\cdot\log\frac{P(t,c)}{P(t)\cdot P(c)}$
Mutual information	Yang and Pedersen (1997)	$\log\frac{P(t_k,c_i)}{P(t_k)\cdot P(c_i)}$
Chi-square	Yang and Pedersen (1997) Yang and Liu (1999)	$\frac{\lvert T\rvert\cdot\left[P(t_k \mid c_i)P(\bar{t}_k,\bar{c}_i)-P(t_k \mid \bar{c}_i)P(\bar{t}_k,c_i)\right]^2}{P(t_k)P(c_i)P(\bar{t}_k)P(\bar{c}_i)}$
NGL coefficient	Ng et al. (1997) Ruiz and Srinivasan (1999)	$\frac{\sqrt{\lvert T\rvert}\cdot\left[P(t_k \mid c_i)P(\bar{t}_k,\bar{c}_i)-P(t_k \mid \bar{c}_i)P(\bar{t}_k,c_i)\right]}{\sqrt{P(t_k)P(c_i)P(\bar{t}_k)P(\bar{c}_i)}}$
Relevance score	Wiener et al. (1995)	$\log\frac{P(t_k \mid c_i)+d}{P(t_k \mid \bar{c}_i)+d}$
Odds ratio	Ruiz and Srinivasan (1999)	$\frac{P(t_k,c_i)(1-P(t_k,\bar{c}_i))}{P(t_k,\bar{c}_i)(1-P(t_k,c_i))}$
GSS coefficient	Galavotti et al. (2000)	$P(t_k \mid c_i)P(\bar{t}_k,\bar{c}_i)-P(t_k \mid \bar{c}_i)P(\bar{t}_k,c_i)$

Cannas et al. (2012) proposed a method to reduce feature space dimension based on the combination on filter and wrapper approaches. In this method the term relevance is weighted by a filter. Then, the top-ranked terms are grouped in several sets of relatively small size. The performance of the proposed approach has been evaluated on the *Reuters* corpus and shows competitive results among existing models.

Vieira et al. (2012) introduce the idea of a fuzzy criterion in feature selection. It allows defining the goal in feature selection in a more flexible way and does not have a problem of weighting different goals. An ant colony optimization algorithm has been applied for the optimization problem.

2.5.4. Wrappers Based on Genetic Algorithms

The popular kind of wrappers is feature selection based on genetic algorithms (GA) (Goldberg and Holland, 1988). GAs are used as optimization algorithms for finding the

optimal or the sub-optimal subset of the original feature set. Classification accuracy with the predefined classification algorithm is used as a fitness function (Yang and Honavar, 1998). GA-based wrappers may be effective tools for text classification as well.

2.5.4.1. Wrappers with Standard Genetic Algorithms

GAs are well-known and widely used methods of global optimization since they can work with algorithmically defined criteria on the binary, real, nominal, mixed etc. data types; they search in parallel in different regions of the feature space. GAs were proposed firstly by Holland and Reitman (1977). These algorithms of stochastic optimization are inspired by the evolution process. This evolutionary principle forms the basis for GAs. The environment is modeled by the optimization criterion (objective function), while the individuals are represented by their chromosomes. GAs are based on the classic view of a chromosome as a string of genes. Each chromosome represents a feature vector as a binary string.

The basic scheme of the standard GA is shown in Figure 2.9. The standard GA is described in Algorithm 2.1.

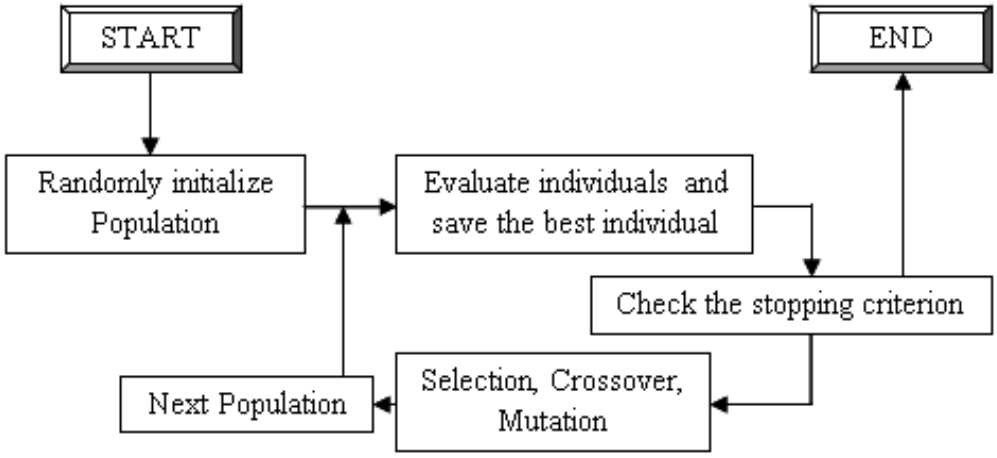

Figure 2.9 – Scheme of standard genetic algorithm

(Taken from: Gasanova, Tatiana (2015): Novel methods for text preprocessing and classification. Open Access Repositorium der Universität Ulm. Dissertation; Text; http://dx.doi.org/10.18725/OPARU-3242)

Algorithm 2.1. Standard genetic algorithm

1. Randomly initialize the population (usually the candidate solutions, individuals, are encoded as binary sequences). Set generation counter $g = 0$.
2. Calculate the fitness functions (chosen optimization criterion) for all individuals.

3. Save the individual with the best value of fitness function

4. Check if the generation counter is greater than some predefined value G (maximal number of generations): if $g > G$ than END; otherwise go to step 5.

5. Using GA operators: selection, crossover and mutation, form the next population.

6. Increment the generation counter $g = g + 1$. Go to step 2.

Selection, crossover and mutation operators are explained in detail in the following paragraphs.

Selection. Selection is the stage of a genetic algorithm in which individual genomes are chosen from a population for later breeding (recombination or crossover). This stage of GA should satisfy two contradictory factors:

1. Selection should choose individuals with high value of the fitness function.

2. Individuals with low fitness function values should also have a chance to be selected; otherwise the diversity of the population will become poor and the algorithm will go to the stagnation (the stage when the algorithm is unable to find better solutions and all populations consist of similar individuals).

There are many variants of the selection operator. In the following, the most popular selection types are described.

- *Proportional selection*. Proportional selection is also known as roulette wheel selection. The probability to be chosen for reproduction is calculated by the following formula:

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_j}, \tag{2.22}$$

where f_i is a fitness function of the i^{th} individual, N is a population size.

- *Linear Ranking selection.* Ranking selection has been proposed by Baker (1985) in order to overcome the drawbacks of the proportionate selection. For such a ranking selection the population is sorted according to the fitness functions of the individuals and then the rank N (equals to a population size) is assigned to the best individual and the rank 1 to the worst individual. It means that the rank of the individual equals to a position of this individual in the population sorted by fitness function. The probability of being selected is calculated as:

$$p_i = \frac{r_i}{\sum_{j=1}^{N} r_j}, \tag{2.23}$$

where r_i is a rank of the i^{th} individual, N is a population size.

- *Tournament selection*. Tournament selection involves running a "tournament" among a few individuals chosen at random from the population. The winner of each tournament (the one with the best fitness) is selected for crossover. The selection pressure is easily adjusted by changing the tournament size. If the tournament size is larger, weak individuals have a smaller chance to be selected.

Crossover. Crossover is a stage of genetic algorithms when selected individuals share their genetic material to produce a next population presumably with better fitness functions of individuals. Several well-known types of recombination operators are shown below.

- *One-point crossover*. In this type of crossover, a point inside the individual's chromosome is randomly selected. The genes of the two parents are divided by this point and are swapped to form the children. Figure 2.10 illustrates one-point crossover.

- *Two-point crossover*. Two points inside the individual's chromosome are randomly selected and their genes are divided into parts and swapped between each two consecutive points. Figure 2.11 illustrates one-point crossover.

- *k point crossover*. One-point and two-point crossovers can be generalized into *k* point crossover, where *k* points inside the chromosome are selected and the genetic material is swapped between each two consecutive points.

- *Uniform crossover*. The uniform crossover uses a fixed mixing ratio between two parents. Unlike one- and two-point crossovers, the uniform crossover enables the parent chromosomes to contribute the gene level rather than the segment level. If the mixing ratio is 0.5, the offspring has approximately half of the genes from the first parent and the other half from second parent. Figure 2.12 illustrates uniform crossover.

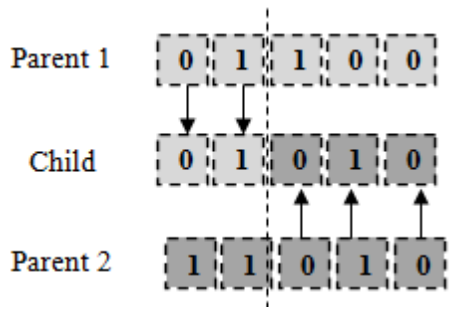

Figure 2.10 – One-point crossover

(Taken from: Gasanova, Tatiana (2015): Novel methods for text preprocessing and classification. Open Access Repositorium der Universität Ulm. Dissertation; Text; http://dx.doi.org/10.18725/OPARU-3242)

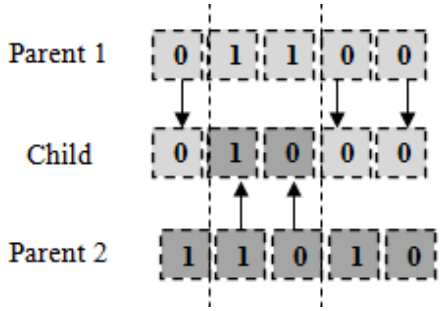

Figure 2.11 – Two-point crossover

(Taken from: Gasanova, Tatiana (2015): Novel methods for text preprocessing and classification. Open Access Repositorium der Universität Ulm. Dissertation; Text; http://dx.doi.org/10.18725/OPARU-3242)

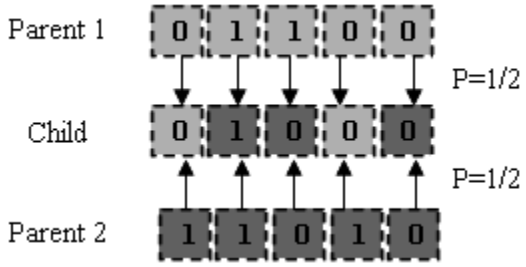

Figure 2.12 – Uniform crossover

(Taken from: Gasanova, Tatiana (2015): Novel methods for text preprocessing and classification. Open Access Repositorium der Universität Ulm. Dissertation; Text; http://dx.doi.org/10.18725/OPARU-3242)

Mutation. Mutation is a genetic operator used to maintain genetic diversity from one generation of a population of genetic algorithm chromosomes to the next. It is analogous to biological mutation. Mutation alters one or more gene values in a chromosome from its initial state. In mutation, the solution may change entirely from the previous solution. Hence GA can come to a better solution by using mutation. Mutation occurs during evolution according to a user-definable mutation probability. This probability should be set low. If it is set too high, the search will turn into a primitive random search.

The purpose of mutation in GAs is preserving and introducing diversity. Mutation should allow the algorithm to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing or even stopping evolution. This reasoning also explains the fact that most GA systems avoid only taking the fittest of the population in generating the next but rather a random (or semi-random) selection with a weighting toward those that are fitter.

GA can be applied as an optimization algorithm for the wrapper. In this case the chromosome length equals to the original dimensionality (feature set size); each bit in the binary string corresponds to one feature (one term from the dictionary in the case of text classification). “1” in the binary string means including the corresponding feature, “0” means excluding the corresponding feature. GA-based feature selection is presented in Algorithm 2.2.

Algorithm 2.2. GA-based wrapper

1. Randomly initialize the population of the binary strings with the length equals to dimensionality of the considered classification problem. Set generation counter $g = 0$.

2. Apply the classification algorithm on the validation set for all individuals form the population (with including only features with „1“ values in the chromosomes)

3. Set the classification effectiveness measure (i.e. *F1*-score) on the validation set as a fitness function value for all individuals.

4. Save the individual with the best value of the fitness function

5. Check if the generation counter is greater than some predefined value *G* (maximal number of generations): if $g > G$ than go to the step 8; otherwise go to step 6.

6. Using GA operators: selection, crossover and mutation, form the next population.

7. Increment the generation counter $g = g + 1$. Go to step 2.

8. Put the best individual as a solution of the feature selection procedure; apply the chosen feature subset for the test set.

2.5.4.2. Self-adjusting Genetic Algorithm

One of the most complicated problems with GA applications is setting the algorithm parameters. A conventional genetic algorithm has at least three selection methods (proportional, tournament, and rank), three recombination methods (one-point, two-point, and uniform) and mutation probability requires tuning as well. Exhaustive search of combinations requires significant computational resources, especially for time-consuming problems such as a GA-based wrapper for text classification. The parameter combination selection performed randomly may also be inappropriate since the algorithm efficiency on the same problem may differ very much with various parameters settings (Sergienko and Semenkin, 2010).

For solving this problem a self-adjusting GA proposed in (Semenkin and Semenkina, 2012) may be applied. The scheme of the self-adjusting GA is presented in Figure 2.13.

In the self-adjusting GA, different types of selection, recombination, and different levels of mutation are executed simultaneously. At the beginning of the algorithm, all types of GA operators have the same probability to be used for a new off-spring generation. After that, the dynamic adaptation of probabilities is performed according to the ”usefulness” of GA operator types in terms of fitness function. The details of the self-adjusting GA are described in Algorithm 2.3:

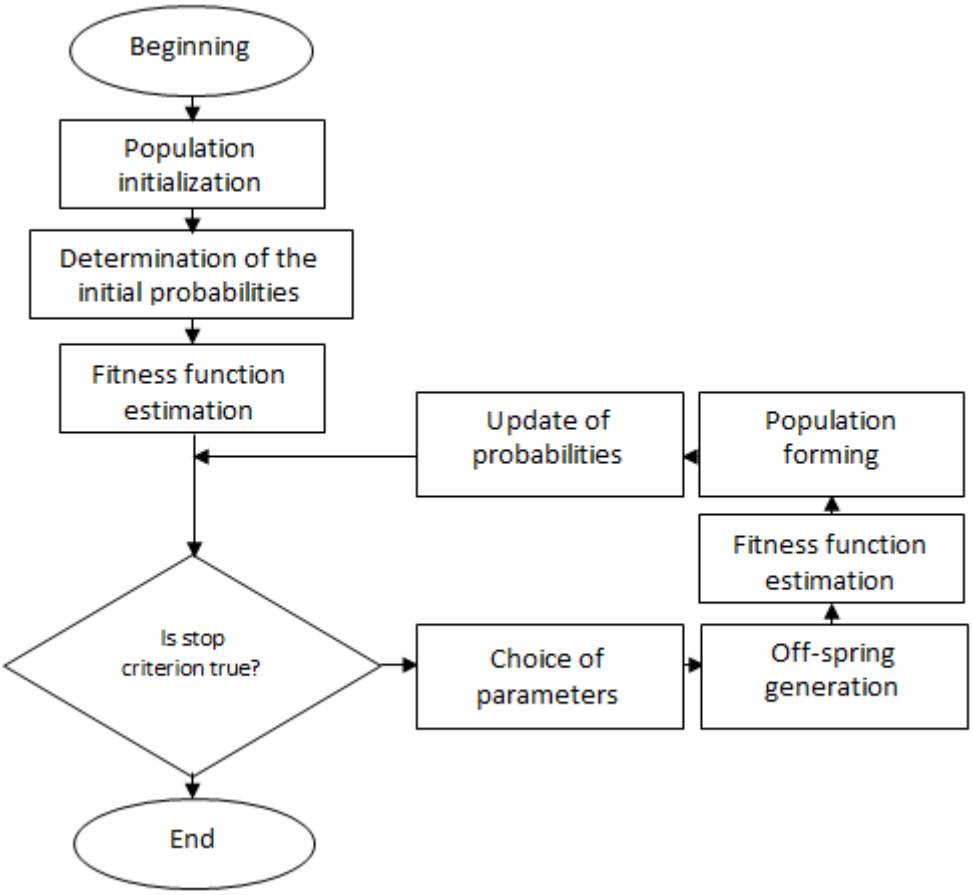

Figure 2.13 – Self-adjusting genetic algorithm

Algorithm 2.3. Self-adjusting genetic algorithm

1. Put the probabilities of all used types of GA operators: $p_{ij} = 1/N_i$, where $j = 1 \ldots N_i$, N_i is the number of types of the i^{th} operator, $i = 1 \ldots N$, N is the number of GA operators. In our case $N = 3$: selection, recombination, and mutation.

2. Set threshold probabilities for all used types of GA operators: $P_{ij} = 3/(10*N_i)$.

3. Generate new population. For each off-spring we randomly choose types of selection, recombination, and mutation according probabilities p_{ij}.

4. Recalculate the probabilities with the following:

4.1. For each i^{th} operator do:

4.1.1. Set $S_i = 0$.

4.1.2. For each j^{th} type of the i^{th} operator do:

4.1.2.1. If $p_{ij} < P_{ij} + 1 / (T*N_i)$ AND $p_{ij} > P_{ij,}$ where T is the number of generations, then: $S_i := S_i + (p_{ij} - P_{ij})$; $p_{ij} := P_{ij}$.

4.1.2.2. If $p_{ij} > P_{ij} + 1 / (T*N_i)$ then: $S_i = S_i + 1 / (T*N_i)$; $p_{ij} = p_{ij} + 1 / (T*N_i)$.

4.1.2.3. Calculate the average fitness function *Fij* of all off-springs of the current generation that were generated with the j^{th} type of the i^{th} operator.

4.1.3. Find the best type *d* of the i^{th} operator with the maximal fitness function on the current generation and recalculate its probability: $p_{id} := p_{id} + S_i$.

5. Check stop criterion. If TRUE then: END; else: go to the step 3.

2.5.5. Principal Component Analysis

Principal Component Analysis (PCA) (Jolliffe, 2002) is a feature transformation method that is mathematically defined as an orthogonal linear transformation that transforms the original feature set to a new coordinate system such that the greatest variance by some projection of the training data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on. The description of PCA in accordance with https://en.wikipedia.org/wiki/Principal_component_analysis:

Consider a data matrix *X* with column-wise zero empirical mean (the sample mean of each column has been shifted to zero), where each of the *n* rows represents a different element of the training data, and each of the *p* columns gives a particular kind of feature. Mathematically, the transformation is defined by a set of *p*-dimensional vectors of weights or loadings $w_{(k)} = (w_1,...,w_p)_{(k)}$ that map each row vector $X_{(i)}$ of *X* to a new vector of principal component scores $t_{(i)} = (t_1,...,t_k)_{(i)}$, given by $t_{k(i)} = X_{(i)} \cdot w_{(k)}$ in such a way that the individual variables of *t* considered over the data set successively inherit the maximum possible variance from *X*, with each loading vector *w* constrained to be a unit vector.

The first principal component is calculated by the following formula in the matrix form:

$$w_{(1)} = \arg\max_{\|w\|=1} \left\{ \|Xw\|^2 \right\} = \arg\max_{\|w\|=1} \left\{ w^T X^T X w \right\}, \tag{2.24}$$

The k^{th} component can be found by subtracting the first $k-1$ principal components from *X*:

$$\hat{X}_k = X - \sum_{s=1}^{k-1} X w_{(s)} w_{(s)}^T . \tag{2.25}$$

The PSA method can be effective in very different applications but the main drawback of the method is its computational complexity that can be inappropriate for

high-dimensional data analysis problems with large training data such as a lot of text classification problems.

2.5.6. Latent Semantic Indexing

Latent Semantic Indexing (LSI) proposed by Deerwester et al. (1990) or Latent Semantic Analysis (LSA) proposed by Landauer et al. (1998) are methods that attempt to overcome the problem of using the synonymous, polysemous words as dimensions of the feature space. This approach reduces the feature space by mapping the initial document vectors of high dimensional space into a lower dimensional space, where the new dimensions are obtained by combining initial terms according to their co-occurrence. According to Deerwester et al. (1990), this feature transformation can also help to remove the noise induced by words in the term document matrix due to certain lexical properties (synonymy and polysemy).

2.5.7. Weight-based Term Clustering

It is possible to organize feature transformation for text classification based on weight-based term clustering.

Lewis (1992) has first proposed the use of weight-based term clustering in text classification. The name of his method is reciprocal nearest neighbor clustering. Lewis has divided the term set by combining pairs of the most similar terms (add the similarity measure). The algorithm performance on this reduced term set was worse than the results on the initial one probably due to the clustering method that was used.

In contrast with those unsupervised term clustering approaches, Baker and McCallum (1998) have proposed a supervised learning model applied to term clustering. The idea was that the terms which point to the same category or the group of categories should be combined in one cluster. As a classification algorithm they have used a Naïve Bayes and its performance on the obtained reduced term set has shown some improvement. The work of Slonim and Tishby (2001) has also shown the advantages of the supervised term clustering.

The method in (Baker and McCallum, 1998) uses relative frequencies of terms in different categories for determination of the most appropriate classes for terms and as metric for clustering. Some novel supervised term weighting methods (GR, CW, RF, TRR, and NTW) include the determination of the most appropriate classes for terms

automatically (see Section 2.4). Therefore, we can use these term weights as a metric for clustering. The term clustering approach based on novel term weighting methods was proposed in (Gasanova et al., 2014b). This method uses hierarchical agglomerative clustering (Ward, 1963). Hierarchical agglomerative clustering starts with small clusters, each of them contains only one element. Then the clusters are united consequentially into larger clusters, until there is only one big cluster, which includes all objects. In every iteration, two clusters are combined, which have the shortest distance between them. All hierarchical algorithms are sensitive to the chosen distance metric. The definition of the shortest distance between two clusters is what differentiates between the different agglomerative clustering methods.

The weight-based term clustering proposed in (Gasanova et al., 2014b) is illustrated in Algorithm 2.4.

Algorithm 2.4. Weight-based hierarchical agglomerative term clustering

---

1. Assign each term from the dictionary of the text classification problem to the most appropriate class:

1.1. If the term weighting method includes the determination of the most appropriate class for terms itself, this assignment is used. Go to step 2.

1.2. Otherwise we assign one class for each term using the relative frequency of the term in classes:

$$S_j = \arg\max_{c \in C} \frac{n_{jc}}{N_c}, \tag{2.26}$$

where Sj is the most appropriate class for the j^{th} term, c is an index of a class, C is a set of all classes, n_{jc} is number of documents of the c^{th} class which contain the j^{th} term, N_c is the number of all documents of the c^{th} class. Go to step 2.

2. For each class perform:

2.1. Set the maximal number of clusters N

2.2. Start with disjointed terms (each term a_i of the current class forms its own cluster c_i).

2.3. Calculate all distances between pairs of clusters. In our case distance d_{ij} between i^{th} and j^{th} clusters equals to: $d_{ij} = | T_i - T_j |$, where T_i and T_j are weights of corresponding clusters.

2.4. Find two closest clusters c_i and c_j ($i < j$).

2.5. Add cluster c_j to cluster c_i. Calculate the new weight of the joined cluster as arithmetical weight mean of all terms that belong to the new cluster. Increment i, $i = i + 1$.

2.6. Recalculate the distances between the new cluster and other clusters.

2.7. If the number of clusters is less than N, go to step 2.4. Otherwise END

2.6. Classification Algorithms

2.6.1. Introduction

Classification or supervised machine learning (predicting the class for given data points) is the last stage of text categorization after text preprocessing (feature extraction, term weighting, and dimensionality reduction). Almost all supervised machine learning algorithms can be applied to the text classification task. Supervised machine learning requires training labeled data, test data for algorithm effectiveness estimation, and additionally validating data (part of the training set) for the optimization of classification algorithm parameters.

Text classification based on the „bag-of-word“ model yields some specific characteristics compared to classical classification problems:

- High dimensionality (a text classification task may contain millions of different terms).

- Large number of zero values in feature vectors (a document or an utterance in case of speech-based text classification cannot contain all terms contained in the dictionary of text classification task);

- Efficiency of a classifier strongly depends on text preprocessing techniques. This implies that classification results, when using the same machine learning algorithm but different term weighting methods, differ much more between each other compared to results obtained with different classifiers but the same term weighting method.

Therefore, comparative studies of text classification algorithms lead to completely different results across different applications domains.

The most common and performed classification algorithms for text classification are Naïve Bayes Classifier (NB) (Zhang and Li, 2007; Kim et al., 2006), *k*-nearest neighbors algorithm (*k*-NN) (Han et al., 2001; Zhou et al., 2009), Rocchio classifier (nearest centroid algorithm) (Joachims, 1996), Support Vector Machine (SVM)

(Joachims, 1998; Tong and Koller, 2002) and artificial neural networks (ANN) (Ruiz and Srinivasan, 1999; Zhang and Zhou, 2006).

In accordance with the overview of machine learning algorithms for text classification in (Baharudin et al., 2010), the following algorithms are considered to be the most appropriate ones in the field of text classification: SVM, *k*-NN, and NB.

This section describes the state-of-the-art in supervised learning models with the focus on the application to text classification.

2.6.2. Naïve Bayes Classifier

Bayesian or probabilistic classifiers have been widely used for text categorization (Lewis, 1998). They apply the joint probabilities of words and classes to estimate the probabilities of each class for an input document.

Let D be a set of n document vectors $D = \{d_1, \ldots, d_n\}$ which are classified into a set C of m classes, $C = \{c_1, \ldots, c_m\}$. Bayesian classifiers estimate the probabilities of each class c_i for an input document d_j as:

$$P(c_i \mid d_j) = \frac{P(c_i)P(d_j \mid c_i)}{P(d_j)}, \tag{2.27}$$

where $P(d_j)$ is the probability that a randomly taken document will be equal to d_j in the feature space; $P(c_i)$ is the probability that a randomly taken document will belong to the class c_i. Since the number of possible documents d_j is very high, it is difficult to estimate $P(d_j \mid c_i)$.

To simplify the estimation of $P(d_j \mid c_i)$, the Naïve Bayes Classifier (NB) assumes that the probability of a given word or term is independent of other terms that appear in the same document. This may seem as an over simplification, but in fact Naïve Bayes presents results that are very competitive with those obtained by more elaborate methods. Moreover, because only words and not combinations of words are used as predictors, this naive simplification allows model computation from data associated with this method to be far more efficient than with non-naive Bayesian approaches. Using this simplification, it is possible to determine $P(d_j \mid c_i)$ as the product of the probabilities of each term that appears in the document. So, $P(d_j \mid c_i)$, where $d_j = (w_{1j}, \ldots, w_{|T|j})$ is a vector of term weights of the document d_j with the dictionary size $|T|$, may be estimated as:

$$P(d_j \mid c_i) = \prod_{k=1}^{|T|} P(w_{kj} \mid c_i). \tag{2.28}$$

The document belongs to the class which has the higher probability of $P(c_i \mid d_j)$:

$$class(d_j) = \arg\max_i P(c_i \mid d_j). \quad (2.29)$$

In accordance with Baharudin et al. (2010), the NB algorithm is effective for such text classification tasks as spam filtering and email categorization and requires a small amount of training data to estimate the parameters necessary for classification. Naïve Bayes works well on numeric and textual data and is easy to implement compared to other algorithms. However, the conditional independence assumption is violated by real-world data and the algorithm performs poorly when features are highly correlated and do not consider frequency of word occurrences.

Ganiz et al. (2011) proposed a new supervised classification algorithm. It is called Higher Order Naive Bayes (HONB) and leverages higher order relations between features across different instances. Experimental results on several benchmark corpora show that the introduced approach achieves significantly better classification accuracy over the baseline methods.

2.6.3. *K*-nearest Neighbor Algorithm (*k*-NN)

K-nearest neighbor (*k*-NN) classification is one of the most fundamental and simple classification methods. It is usually applied should only little or no prior knowledge about the distribution of the data exists. An unpublished US Air Force School of Aviation Medicine report by Fix and Hodges (1951) introduced a non-parametric method for pattern classification that has since become known the *k*-nearest neighbor rule. For text classification the *k*-nearest-neighbor algorithm has been proposed by Masand et al. (1992). The idea of this method is to determine the category of an input document on the basis of not only the document which is the nearest to it in the feature space, but also on the *k* closest documents.

There exists a modification of the *k*-NN algorithm with weight distances when *k* elements from the training set have weights in accordance with their positions: the nearest element yields the maximal weight; the farthest one has the minimal weight.

Another important parameter for *k*-NN is the metric used to compute distances between different examples. The following metrics between elements $x = \{x_1, \ldots x_T\}$ and $y = \{y_1, \ldots y_T\}$can be applied:

- taxicab metric:

$$taxicab(x, y) = \frac{1}{T} \cdot \sum_{i=1}^{T} |x_i - y_i| . \quad (2.30)$$

- Euclidean metric:

$$Euclidean(x, y) = \frac{1}{T} \sqrt{\sum_{i=1}^{T} (x_i - y_i)^2} . \quad (2.31)$$

- cosine similarity:

$$cosine\_similarity(x, y) = 1 - \cos(x, y) = 1 - \frac{\sum_{i=1}^{T} x_{Ti} \cdot y_i}{\sqrt{\sum_{i=1}^{T} x_i^2 \sum_{i=1}^{T} y_i^2}} . \quad (2.32)$$

In accordance with (Baharudin et al., 2010), if a suitable text preprocessing is used with *k*-NN, then this algorithm continues to achieve very good results of text classification and scales up well with the number of documents, which is not the case for SVM (Yuan et al., 2008; Colas and Brazdil, 2006). As for naive Bayes, *k*-NN also achieves good performance with suitable preprocessing. The algorithm performes well as more local characteristics of documents are considered, however the classification time is long and it is difficult to find an optimal value of *k*.

There are some examples of effective *k*-NN applications for text classification tasks presented in (Han et al., 2001; Zhou et al., 2009).

2.6.4. Nearest Centroid Classifier (Rocchio Classifier)

Nearest centroid classifier or Rocchio Classifer is one of the simplest classification methods. It has been proposed by Rocchio (1971). For each category *c* the weighted centroid is calculated using the following equation:

$$g_c = \frac{1}{|v_c|} \sum_{d \in v_c} d - \gamma \frac{1}{|\overline{v_{c,k}}|} \sum_{d \in \overline{v_{c,k}}} d, \quad (2.33)$$

where v_c is set of documents that belong to the category *c*, $\overline{v_{c,k}}$ is a set of documents that do not belong to the category *c* and that are close to the centroid $\frac{1}{|v_c|} \sum_{d \in v_c} d$, and γ is the parameter of the relative importance of negative precedents consideration.

The application of this method to text classification was proposed by Hull (1994) and it is used in the case where the role of negative precedents is deemphasized (usually γ= 0.25).

There exist several other approaches to calculate the centroids:

- The average formula where each centroid g_c is represented as an average of all vectors which belong to the class *c*:

$$g_c = \frac{1}{|v_c|} \sum_{d \in v_c} d \,. \tag{2.34}$$

- The sum formula where each centroid g_c is represented as a sum of all vectors which belong to the class *c*:

$$g_c = \sum_{d \in v_c} d \,. \tag{2.35}$$

- The normalized sum formula where each centroid g_c is represented as a sum of all vectors which belong to the class c, normalized so that it has unitary length:

$$g_c = \frac{1}{\left| \sum_{d \in v_c} d \right|} \sum_{d \in v_c} d \,. \tag{2.36}$$

In accordance with (Baharudin et al., 2010), the Rocchio classifier is easy to implement, efficient in computation, fast learner and has a relevance feedback mechanism but low classification accuracy for text classification. Linear combination is too simple for classification and the constant γ is empirical. The Rocchio classifier is a widely used relevance feedback algorithm that operates in the vector space model (Ittner et al., 1995). The researchers have used a variation of Rocchio's algorithm in a machine learning context, i.e., for learning a user profile from unstructured text (Balabanovic and Shoham, 1997; Pazzani and Billsus, 1997), the goal in these applications is to automatically induce a text classifier that can distinguish between classes of documents.

2.6.5. Support Vector Machines

Support Vector Machines (SVMs) have been proposed in (Vapnik and Chervonenkis, 1974; Cortes and Vapnik, 1995) and have been first applied to text classification by Joachims (1998).

The SVM is a method for training linear classifiers. It is based on statistical learning algorithms: it maps the documents into the feature space and attempts to find a hyper-plane that separates the classes with largest margins. A SVM can be interpreted as an extension of the perceptron. It simultaneously minimizes the empirical classification error and maximizes the geometric margin.

In the most common formulation, SVMs (Vapnik and Chervonenkis, 1974) are classification mechanisms, which, given a training set

$$X^l = \{(x_1, y_1), ..., (x_l, y_l)\}, x_i \in R^m, y_i \in \{-1;1\}, \quad (2.37)$$

assuming l examples (documents) with m real attributes (term weights in the case of text classification), y_i is a class label which is equal to -1 if the i^{th} document belongs to the first class and is equal to 1 if it belongs to the second one, learn a hyper-plane:

$$< w, x > + b = 0, \quad (2.38)$$

where $w \in R^m, b \in R, < \_, \_ >$ is a dot product, which separates examples labeled as -1 from the ones labeled as +1. Therefore, using this hyper-plane, a new instance x is classified using the following classifier:

$$f(x) = \begin{cases} 1, (< w, x > + b) \geq 1 \\ -1, (< w, x > + b) \leq -1 \end{cases}. \quad (2.39)$$

A SVM is based on maximization of the distance between the discriminating hyper-plane and the closest examples (support vectors, see Figure 2.14). This maximization reduces the so-called structural risk, which is related to the quality of the decision function. The most discriminating hyper-plane can be computed by solving the following constrained optimization problem:

$$\|w\|^2 \rightarrow \min, \quad (2.40)$$

$$y_i (< w, x_i > + b) \geq 1, i = \overline{1, l}. \quad (2.41)$$

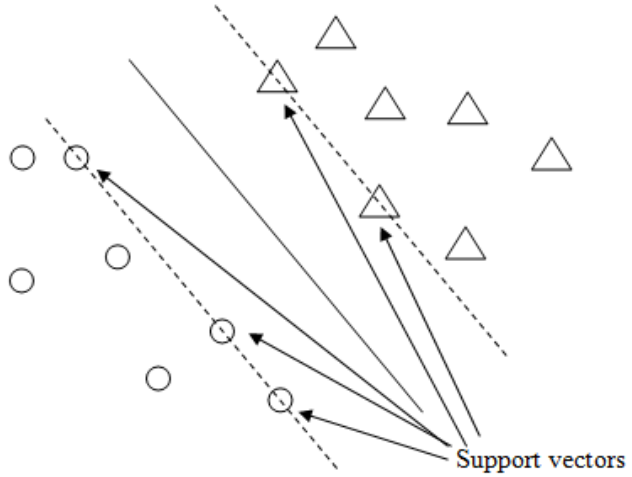

Figure 2.14 – Support Vector Machine

(Taken from: Gasanova, Tatiana (2015): Novel methods for text preprocessing and classification. Open Access Repositorium der Universität Ulm. Dissertation; Text; http://dx.doi.org/10.18725/OPARU-3242)

However, the given data set is not always linearly separable, and in this case the SVM (as a linear classifier) does not provide satisfying classification results. One way to solve this problem is to map the data onto a higher dimension space and then to use a linear classifier in that space. The general idea is to map the original feature space to some higher-dimensional feature space where the training set is linearly separable. SVMs provide an easy and efficient way of doing this mapping to a higher dimension space, which is referred to as the kernel trick (Boser et al., 1992). For instance, the polynomial kernel can be used.

So, let $K(x, x') = (\alpha < x, x' > + \beta)^d$, where α, β, d are parameters of the kernel function *K*. Then the classifier is:

$$f(x) = \begin{cases} 1, ((K(w, x) + \beta)^d + b) \geq 1 \\ -1, ((K(w, x) + \beta)^d + b) \leq -1 \end{cases}. \quad (2.42)$$

It means that the following constrained optimization problem should be solved:

$$\|w\|^2 \rightarrow \min, \quad (2.43)$$

$$y_i((\alpha < w, x_i > + \beta)^d + b) \geq 1, i = \overline{1, l}. \quad (2.44)$$

Thus for solving a classification problem, the kernel function's parameters α, β, d, a vector *w* and a shift factor *b* should be determined, i.e. this constrained optimization problem with continuous variables has to be solved.

The above described two-class SVM algorithm can be easily transformed into a multi-class classifier using the „one class against all" approach.

There are different modifications of SVM; one of them is the Fast Large Margin (SVM-FML) (Fan et al., 2008) which can be effective for high-dimensional classification problems with large training data such as text classification tasks. The Fast Large Margin algorithm applies a fast margin learner based on the linear support vector learning scheme. Although the result is similar to those delivered by classical SVM or logistic regression implementations, this linear classifier is very efficient for training large-scale problems and able to work on data sets with millions of examples and attributes. For example, it takes only several seconds to train a text classification problem from the Reuters Corpus Volume 1 (Rose et al., 2002) that has more than 600,000 examples (documents). For the same task, a general SVM solver would take several hours. Moreover, SVM-FML is competitive with or even faster than the state-of-the-art linear

classifiers such as Pegasos (Shalev-Shwartz et al., 2007) and SVMperf (Joachims, 2006). SVM-FML was applied for the numerical experiments presented in this thesis.

In accordance with (Baharudin et al., 2010), the SVM classifier has been recognized as one of the most effective text classification method in comparison of supervised machine learning algorithms. Some SVM modifications have the ability to learn independent of the dimensionality of the feature space and global minima vs. local minima; however, for SVM some difficulties have been found related to parameter tuning and kernel selection.

2.6.6. Artificial Neural Networks

An artificial neural network, usually called neural network (NN), is a mathematical model or computational model that is inspired by the structure and functional aspects of biological neural networks. A neural network consists of an interconnected group of artificial neurons, and it processes information using a connectionist approach to computation (the central connectionist principle is that mental phenomena can be described by interconnected networks of simple and often uniform units). In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase. Modern neural networks are usually used to model complex relationships between inputs and outputs or to find patterns in data.

A feed-forward neural network is an artificial neural network where connections between the units do not form a directed cycle. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) to the output nodes. There are no cycles or loops in the network.

Different types of ANNs have been implemented to document classification tasks. Some use the single-layer perceptron which contains only an input layer and an output layer due to its simplicity of implementation (Ng et al., 1997). Inputs are fed directly to the outputs via a series of weights. In this way it can be considered the simplest type of feed-forward network. The multi-layer perceptron which is more sophisticated consists of an input layer, one or more hidden layers, and an output layer in its structure. It is also widely implemented for text classification tasks (Ruiz and Srinivasan, 1999).

The most popular method of ANN learning is the error backpropagation algorithm (Hecht-Nielsen, 1989), which allows training an ANN within a reasonable period of time with an appropriate result.

The backpropagation learning algorithm can be divided into two phases: propagation and weight update. These phases are described in Algorithm 2.5.

Algorithm 2.5. Error backpropagation learning for ANN

---

1. Propagation

Each propagation involves the following steps:

1.1. Forward propagation of a training pattern's input through the neural network in order to generate the propagation's output activations.

1.2. Backward propagation of the propagation's output activations through the neural network using the training pattern target in order to generate the deltas (the difference between the targeted and actual output values) of all output and hidden neurons.

2. Weight update

For each weight-synapse follow the following steps:

2.1. Multiply its output delta and input activation to get the gradient of the weight.

2.2. Subtract a ratio (percentage) of the gradient from the weight.

This ratio (percentage) influences the speed and quality of learning; it is called the learning rate. The greater the ratio, the faster the neuron trains; the lower the ratio, the more accurate the training is. The sign of the gradient of a weight indicates where the error is increasing, this is why the weight must be updated in the opposite direction.

3. Repeat phase 1 and 2 until the performance of the network is satisfactory.

However, the error backpropagation algorithm does not take the structure of an ANN (the number, location and activation functions of neurons) into account. All these features yield a significant influence on the training quality and training speed; it would be effective to control them in order to improve the result of the training process. There are some approaches to ANN structure optimization based on evolutionary algorithms, such as a genetic algorithm (Bukhtoyarov and Semenkina, 2010; Bukhtoyarov et al., 2011). It is also possible to optimize neuron weights with GA (Whitley et al., 1990). However, GA-based approaches require a significant computational time and

computational resources due to the high dimensionality of the optimization problem and the complexity of the fitness function calculation.

In recent years, artificial neural networks have been applied for text classification problems to improve efficiency. Text categorization models using back-propagation neural networks (BPNNs) and modified back-propagation neural networks (MBPNNs) are proposed in (Yu et al., 2008) for document classification (in the introduced algorithm the feature selection method is applied to reduce the feature space dimensionality and also to improve the classification quality).

A novel neural network for text classification method has been presented in (Trappey et al., 2006), which is helpful for companies to manage patent documents more efficiently.

In the work by Zhang and Zhou (2006) a novel type of an ANN algorithm which is called Back propagation for Multi-Label Learning (BP - MLL) has been introduced. It is based on the usual Back propagation algorithm with a novel error function capturing the characteristics of multi-label learning (the class labels which belong to the given object are ranked higher than those class labels which do not belong). The numerical experiments conducted on two applications: functional genomics and text categorization have shown that the BP-MLL performs better in comparison to some well-established multi-label learning algorithms.

The main drawback of ANNs is their high computing cost on the learning stage which consumes high CPU and physical memory usage. Especially, it can be significant for high-dimensional problems such as text classification. However, neural networks for document classification produce good results in complex domains (Baharudin et al., 2010).

The novel direction of ANN applications for text classification is deep belief networks (so-called „deep learning“ when more than one hidden layer is used) that demonstrates very encouraging results (Sarikaya et al., 2014).

2.7. Summary

Based on the analysis of the state-of-the-art, we can conclude that utterance classification may significantly contribute to SDS research: topic detection of user utterances and user state recognition including text-based user verbal intelligence recognition and text-based user emotion recognition. Solving all these problems is related

to designing a new generation of SDSs: multi-domain user-adaptive SDSs. Topic detection of user utterances is important for a domain detection module of such a SDS. User verbal intelligence recognition and user emotion recognition are tasks of a user state recognition module of a multi-domain user-adaptive SDS. The considered utterance classification problems can be considered as text classification tasks and the existing approaches for text classification may be applied.

Text classification consists of the following important stages: textual feature extraction, term weighting, dimensionality reduction, and classification with machine learning algorithms. For the considered utterance classification problems this is necessary to find the most effective combinations of approaches for all text classification stages (the first aim of the thesis).

There are various unsupervised and supervised term weighting methods for text classification; different term weighting methods are effective for different applications. In our study the following methods are considered: TF-IDF, GR, CW, TM2, RF, TRR, and NTW. These approaches demonstrate different efficiency for different text classification tasks. An approach, which can combine advantages of different term weighting methods, would be promising. It may be possible to increase the classification accuracy with collectives of different term weighting methods. Therefore, it may help to achieve the second aim of the thesis: to improve utterance classification results.

Dimensionality reduction is also a significant stage in text classification because it influences to classification and computational effectiveness simultaneously. Different approaches exist including feature selection or feature transformation, linguistic or numerical approaches. They all perform differently depending on the domain of use. For real-time speech-based text classification systems that can be incorporated into SDSs, it seems to be especially important to provide significant dimensionality reduction with appropriate classification effectiveness. In this thesis the following existing dimensionality reduction methods are applied for utterance classification: „stop-words“ filtering in combination with stemming, weight-based term filtering, GA-based wrapper, and weight-based supervised hierarchical agglomerative term clustering. A novel approach of dimensionality reduction for text classification, which would be able to reduce the dimensionality significantly, would lead to achieve the third aim of the thesis: to improve computational effectiveness of utterance classification.

For the last stage of text classification – machine learning, the following algorithms seem to be the most appropriate in terms of classification and computational

effectiveness: the *k*-NN algorithm and SVM-based algorithms. Therefore, these algorithms are considered and applied in our study. Additionally, ANNs may also produce effective classification results. However, ANNs require significant computational resources. Therefore, it is important to develop less time-consuming machine learning approaches with ANNs. It also leads to improve computational effectiveness of utterance classification (the third aim of the thesis).

3. Corpora Description

3.1. Introduction

For achieving the different aims of the thesis it is necessary to make use of corpora related to the considered utterance classification problems (topic detection of user utterances and user state user recognition including verbal intelligence identification and text-based user emotion recognition). This is not an obligation that these corpora must contain direct conversations between a user and a SDS. However, they should contain conversations with a similar behavior as in a human-machine speech interaction. This means that the conversations should be task-oriented: speakers should solve a problem or obtain problem-related information during a conversation or a discussion.

One of the most appropriate types of such conversations is speech interaction in call centers. In this case, there exist a user which needs to solve a problem or obtain information and an employee of a call center which must help the user. Another appropriate type of conversations is discussions on an actual topic. These conversations may be collected from TV-shows, interviews etc.

For domain detection of user utterances two corpora are considered. The first one (the „Speech Cycle“ corpus, section 3.2.1) contains user utterances from caller interactions with automated agents. Utterances in this corpus are direct interactions with a SDS and they contain only one phrase as a query for further routing (so-called natural language call routing). Therefore, this corpus is ideally appropriate for topic detection as in a multi-domain SDS. The second corpus (the “Rafaeli” corpus, section 3.2.2) contains answers of call center employees with different customer orientation behaviors. Therefore, these two corpora contain different types of utterances: the “Speech Cycle” corpus contains task-oriented queries and the “Rafaeli” corpus contains task-oriented informative answers. Both types of user utterances are also used by a human in a human-machine speech interaction with a SDS. This implies that both corpora are appropriate for our research as databases of task-oriented user utterances of different types. Furthermore, the “Speech Cycle” corpus has excessive data and the “Rafaeli” corpus has scarce data for machine learning. It allows testing text classification approaches for topic identification of user utterances in different conditions.

For user state recognition two corpora of verbal intelligence recognition (section 3.3.1) and a corpus for emotion recognition (section 3.3.2) are considered. The first corpus for verbal intelligence recognition consists of monologues that are descriptions of films and the second one consists of dialogues that are discussions on an education system in Germany. Both corpora contain task-oriented utterances. This would be interesting to demonstrate differences of verbal intelligence recognition from monologues and dialogues. For user emotion recognition the VAM corpus which consists of utterances collected from a TV-show is considered.

In the following, the characteristics of the employed corpora are described.

3.2. Topic Identification of User Utterances

3.2.1. “Speech Cycle” Corpus

The “Speech Cycle” corpus consists of 292,156 user utterances recorded in English from caller interactions with automated agents. Utterances are short and contain only one phrase for further routing. They are task-oriented user queries. The database contains utterance transcripts. They have been manually labeled by experts and divided into 20 classes (such as appointments, operator, bill, internet, phone and technical support). One of them is a special class TE-NOMATCH which includes utterances that cannot be attributes to another class or can be attributed to more than one class.

The database contains 45 unclassified calls which have been removed. It also contains 23,561 empty calls. These were automatically placed in the class TE-NOMATCH and then removed. The remaining calls contained in the database are mostly short. Many of them contain only one or two words. The average length of an utterance is 4.66 words, the maximal length is 19 words. Many identical utterances exist in the database. The corpus therefore contains only 25,079 unique non-empty classified calls. From the classes and their distribution (Table 3.1) we conclude that the corpus is rather unbalanced.

Due to the very high frequency of a small number of utterances in the corpus, correct recognition of these utterances is most important for the final classification score. This implies that a classifier, which would be able to effectively recognize different non-recurring utterances, may obtain a lower value of the final classification score than another classifier which would be able to effectively recognize a small number of frequent utterances. Additionally, frequent utterances may occur in the training and the

test sets simultaneously. In this case a classifier with overffiting may demonstrate high performance on the considered corpus. It implies that the overfitting problem may be hidden. Furthermore, in case of utterances with very high frequencies the results of different classifiers may vary slightly. Therefore, it may be difficult to perform a significant comparative study of different approaches. Due to these characteristics, three different data configurations of the corpus were formulated that have been admitted in this thesis:

Table 3.1

The distribution and examples of the classes for the „Speech Cycle“ corpus

Class	Percentage in the whole corpus	Percentage without repetitions (unique utterances)	Examples
TE-NOMATCH	13.95	27.16	“elephants ordering pizza”
serviceVague	1.06	2.04	“I'd like service”
appointments	3.60	2.76	“to confirm an appointment”
none	2.05	2.20	“I said not any of those”
cancelService	0.40	1.42	“need to cancel subscription”
idk (I don’t know)	0.20	0.62	“I really don't know”
orders	6.32	3.68	“pay per view orders”
UpForDiscussion_Complaint	0.04	0.20	“I wanna make a complaint”
operator	8.15	16.10	“I need to talk to somebody”
techSupport	24.87	7.46	“just try tech support”
bill	27.05	10.88	“you mean my payment”
internet	1.91	2.07	“about my internet”
phone	1.00	2.04	“it’s about my telephone”
techSupport_internet	0.24	1.56	“problem with the modem”
techSupport_phone	0.24	1.93	“telephone is not working”
techSupport_video	0.76	6.01	“uh my TV is real fuzzy”
video	1.96	4.17	“correct cable service”
changeService	3.79	3.41	“change to my account”
UpForDiscussion_no_audio	2.32	3.88	“I have no dial tone”
UpForDiscussion_AskedToCall	0.09	0.41	“I am returning your call”

Data configuration 1. The entire database with 268,550 classified non-empty calls is used for establishing training and test sets. The number of utterance repetitions in sets is used as a weight of the utterance for the classification. This is closest to a real situation. However, frequently repeated utterances decrease the difference between preprocessing and classification methods. Additionally, there are some identical utterances in the training and test sets. In this case the overfitting problem of classification may be hidden.

Therefore, this data configuration is not very appropriate to be admitted in a comparative study of different text classification approaches.

Data configuration 2. The scheme of training and test sample generation is the same as for data configuration 1. However, the numbers of utterance repetitions are ignored. In this case there are intersections between training and test samples.

Data configuration 3. Before establishing training and test samples, all utterance duplicates were removed from the database. This implies that there is no intersection between training and test sets and the frequency of utterances is ignored.

Therefore, data configuration 1 should be suitable for the quality estimation of a real natural language call routing system. In turn, data configuration 3 seems to be the most appropriate for the comparative study of different preprocessing and classification methods. Data configuration 2 ranges between configurations 1 and 3.

For statistical analyses we randomly performed 20 different divisions of the database into training and test sets. This procedure was performed for data configuration 3 separately. The train samples contain 90% of the calls and the test samples contain 10% of the calls. For each training sample we have designed a dictionary of unique words which appear in the training sample after deleting punctuation and transforming capital letters to lowercase. The size of the dictionary varies from 3,275 to 3,329 words for data configurations 1 and 2 (average dictionary size equals to 3,304) and from 3,277 to 3,311 for data configuration 3 (average dictionary size equals to 3,295).

3.2.2. “Rafaeli” Corpus

This corpus is provided by Rafaeli et al. (2008). The main goal of the corpus creation is to identify specific customer orientation behaviors (COBs) of call center employees and to show that these behaviors relate to customer evaluations of service quality.

Using qualitative, inductive analyses of 166 telephone service interactions in a retail bank call center, the author of the corpus identified five types of COBs associated with helping customers. The COBs are (a) anticipating customer requests, (b) offering explanations/justifications, (c) educating customers, (d) providing emotional support, and (e) offering personalized information.

Therefore, we can formulate the problem of topic identification of employee utterances as a text classification problem. These utterances may be interpreted as task-

oriented informative answers of a user. Similar utterances may be pronounced by a user during a human-machine interaction with a SDS.

Totally, there are 337 utterances in English for classification in the corpus. The distribution of the classes and the examples of the utterances are presented in Table 3.2.

Table 3.2

The distribution and examples of the classes for the "Rafaeli" corpus

Class	Percentage (number)	Example
anticipating customer requests	23.15 (78)	"if you want I can give you that mailing address"
offering explanations / justifications	28.45 (96)	"okay and it's only going to access his primary checking account"
educating customers	17.21 (58)	"and put your account number on the back of the check and put for deposit only"
providing emotional support	21.96 (74)	"I'm glad I could help you"
offering personalized information	9.20 (31)	"And what about yeah okay now there's also one in the village too you know that one"

From Table 3.2 we observe that the corpus is partly unbalanced. The proportion between the largest class and the smallest one equals 3.1 : 1.

Due to the small size of the second corpus, we have performed the Leave-One-Out (LOO) cross-validation for feature extraction, dimensionality reduction, and classification with the "Rafaeli" corpus. The average dictionary size equals 842 (min = 834, max = 843).

Therefore, we have one corpus with excess data ("Speech Cycle" corpus) and another corpus ("Rafaeli" corpus) with scarce data for machine learning.

3.3. User State Recognition

Effective user state recognition is helpful for providing a user-adaptive SDS. User state recognition includes the following tasks: user emotion recognition, user verbal intelligence recognition, user age and gender recognition. In our study we consider user emotion and verbal intelligence recognition as text classification tasks. This implies that only linguistic information is used for recognition. Therefore, corpora for user state recognition should contain utterance transcripts.

3.3.1. Corpora for Verbal Intelligence Recognition

Corpora for verbal intelligence identification have been described by Zablotskaya et al., 2010. Their criterion was motivated by the development of algorithms for automatic estimation of a person's verbal intelligence - in other words the level of verbal cognitive processes - based on the analysis of transcribed spoken utterances. There are many psychological research efforts that have shown that different words of everyday speech may reflect intellectual states of a speaker.

The first corpus consists of German monologues by native speakers and the second one consists of German dialogues collected at the University of Ulm, Germany.

The monologues are descriptions of two short films from the TV-program ”Galileo” (http://www.prosieben.de/tv/galileo/). After the first film had been shown, the participants were asked to imagine that they met their good friends and wanted to tell them about one interesting film. They were asked to describe the story to their friends. Then they were shown the second film and their speech was also recorded.

The first film was about the craziest hotels in the world: a ”capsule hotel” in Tokyo, where each room consists of a horizontal plastic box about 6 feet long, 2 feet wide and 2 feet high, complete with television and radio; a temporary ice hotel made up entirely of snow and sculpted blocks of ice; a Berlin theme hotel, where 40 themed rooms have a couple of real standouts, like the Flying Bed and Grandma's; different tree-houses, which are perched 8 to 10 meters above the ground, accommodate four to six people, and can be rented for the night; the Hotel Everland – the first mobile hotel and a contemporary artwork, which was installed on the roof top of the ”Palais de Tokyo” in Paris in 2007.

The second film was about an experiment on how long people could stay awake. Two men and one woman were asked to stay in the same house and to fight against sleep. When they were in a bathroom, they had to sing a song or to whistle. The participants also had to take different tests to control their concentration, memory, attention, condition and general well-being. The winner of the experiment, a woman, could be without sleep for 58 hours. At the end of the film it was told that sleep was very necessary and experiments with animals showed that being without sleep can be dangerous to life.

When the participants were asked to discuss about these films, the authors of the corpora were not aiming at official or scientific speech. They needed normal speech, which these persons used every day when they talked to their family, children and friends, and when they talked about general things. It was not important how many details of

these films they could remember. The main idea was to give them the same topic to talk about. An example of the monologues is illustrated below (translated from German):

"Well, the film was about hotels, mad hotels, not usual hotels. The first one was in Tokyo, it was a hotel where you could see only small boxes for sleeping, small beds, and there were no rooms there. It had communal showers and so on. There was also an ice hotel in the Alps. Igloos were built in the same way as Eskimos did. People slept in sleeping bags on coats. There was a hotel in Berlin, where the rooms were unusually furnished. For example, a mine tunnel, a prison cell, a castle. Exactly as everybody likes. There was a hotel which had only one room, quite small and mobile. It stands at the moment in Paris on a museum. It could stand in other places, where you would like. And there was another hotel. Yes, tree-house hotel, which stayed somewhere in a forest. Five tree-houses were built, put very high, and set in the forest. It was for people who would like to spend the nights in the nature."

„Hi. Yesterday I watched a film. It was Galileo. And it was about a project, about sleep, how long people could stay awake. When they were not concentrated, they had to participate in different experiments. For example, to park a car. And it was really interesting. There were three candidates and they had to perform these tasks. When they were overtired, they were getting cold. But it was 23 degrees in the room. It was interesting to know that some experiments were conducted with mice. The mice were not allowed to sleep. They died after two weeks because they were too overtired. They also said that if you are awake for so many hours, you will feel as you have drunk a glass of wine. So, it is also dangerous."

The second corpus contains dialogues. The topic of the dialogue was about the education and the school system in Germany. The participants had to express their opinions, to determine advantages and disadvantages of the school system, to talk about teachers, lectures, marks and etc. For the experiment the participants could come alone or with a dialogue partner. If they intended to come alone, the author of the corpus asked another participant or some of the colleagues to join them to discuss this problem. If the candidates had not met each other before and had difficulties in making a dialogue, they were asked to dispute and to prove a certain position about the school system. For example, they were asked to imagine that they had different points of view about German education. The first participant was asked to prove that the school system in Germany is

very good, that the children get a very good education and it is no use making changes to it. The second participant was asked to describe bad features of German education, make different examples and to offer some innovations. Sometimes it was easier for the participants to dispute, because they could analyze the position of the dialogue partner and to react in some way. But sometimes it was more difficult, because the participants could not find (for example) good features of the education if their private opinion was different. An example of the dialogues is illustrated below (translated from German, *P1* – participant 1, *P2* – participant 2):

P1/ I think that teachers work very hard. They have their lessons, but they have to prepare something for them. And after the lessons they have to check something. It takes much time. I think, they are paid for these hours.

P2/ Hmm.

P1/ So, the children have to go home and to learn their lessons with their parents.

P2/ Yes, it is very often.

P1/ But, it doesn't work!

P2/ Yes, it is not possible.

P1/ Because their parents are at work!

P2/ Yes.

P1/ Because they have to earn a living, their children need money.

P2/ Yes, the parents are overbusy.

P1/ Do you have children?

P2/ Yes, I have a son. When he went to school I saw that he wasn't overbusy, he didn't have much homework.

P1/ They have to do much homework! It is better than to play computer games. And the lessons have to be more interesting.

P2/ And they have to learn for themselves.

P1/ Yes, you are right.

The first corpus contains 100 monologues of 100 different speakers. The second corpus contains 53 dialogues of 91 different speakers. 52 dialogues are conversations between two persons and one dialogue is a conversation between three persons. The set of the dialogue speakers is a subset of the set of the monologue speakers. The speakers are from different social groups with different education levels.

The corpora contain audio files and also text documents. The data corpora contain the verbal intelligence quotient of each speaker, which were measured with the Hamburg Wechsler Intelligence Test for Adults (Wechsler, 1982), the German version of the American original. Its scale is based on a projection of the subject's measured rank on the Gaussian bell curve with a center value (average IQ) of 100 and a standard deviation of 15. The test is organized for adults ranging in age from 16 to 74 years and consists of 6 verbal and 5 performance tests. Education, experience and lifestyle also contribute to scoring better on this test. For the research the authors of corpora used only the verbal section:

- *Information.* With this sub-test the general knowledge is measured; 25 questions address cultured issues. For example, ”What is the capital of Russia?”

- *Comprehension*. This sub-test measures social awareness and common-sense. It focuses on the social sense and the conception of cultural values. For example, ”What would you do if you lost your way in a forest?”

- *Digit Span*. The auditory short memory, concentration and attention are measured with this sub-test. A participant is asked to repeat strings of digits forward and then backward.

- *Arithmetic*. Arithmetic problems are offered in a story-telling way to identify mental alertness. It focuses upon attention and concentration while manipulating mental mathematical problems. For example, ”Seven envelopes cost twenty five cents. How many envelopes can you buy if you have one dollar?”

- *Similarities in Dissimilar Objects*. A test taker is asked to find abstract similarities among different objects, for example among ”a dog” and ”a lion”. With this test, abstract reasoning and power of conceptualization are measured.

- *Vocabulary*. A participant is asked to explain the meaning of different words, for example ”to crawl” or ”a needle”. The sub-test measures the comprehension of meanings and relations between the expressive words. For example, ”What does the word zebra mean?”

The speakers were clustered into two classes with the *K*-means algorithm (Zablotskaya et al., 2012). The first class means lower verbal intelligence and the second one means higher verbal intelligence. The corpus of the monologues contains 40 speakers with lower verbal intelligence (40.0%) and 60 speakers with higher verbal intelligence (60.0%). The corpus of the dialogues contains 37 speakers with lower verbal intelligence (40.7%) and 54 speakers with higher verbal intelligence (59.3%).

For each speaker from the corpus of the dialogues, all phrases were extracted from the dialogues and placed into the same file because a speaker can be involved in more than one dialogue. Therefore, 53 dialogues were transformed to 91 documents, each of them corresponds to one speaker.

The distribution of classes for the corpora of verbal intelligence recognition are presented in Table 3.3 (examples are illustrated above).

Table 3.3

The distributions of the classes for the corpora of verbal intelligence recognition

Class	Percentage (number)	
	Monologue corpus	**Dialogue corpus**
Higher verbal intelligence	60.0 (60)	59.3 (54)
Lower verbal intelligence	40.0 (40)	40.7 (37)

Due to the small size of the corpora we used Leave-One-Out (LOO) cross validation for feature extraction, feature selection, and classification. The average dictionary size for monologue corpus equals 3,138 terms; for dialogue corpus the average dictionary size equals 6,987 terms.

3.3.2. VAM Corpus for Emotion Recognition

For text-based emotion recognition the corpus VAM ("Vera am Mittag") was used which was introduced in (Grimm et al., 2008). The corpus was created by Karlsruhe Institute of Technology. It is a collection of spontaneous emotional speech extracted from a German talk-show "Vera am Mittag". The emotions are described based on a three-dimensional, continuous-valued emotion primitives (valence, activation, and dominance). The database is composed of two parts, the first part contains 19 speakers with 499 utterances and the second, includes 519 additional sentences from 28 speakers. Therefore, overall there exists 1018 utterances from 47 speakers. From the 947 labeled utterances, their class distributions are as follows: 48% "sad-bored", 45% "angry-anxious", 5% "relaxed-serene" and 2% "happy-exciting".

For statistical analyses we performed 20 random separations of the corpus into a training and a test sets with proportion 9:1. The average dictionary size equals 1550 terms (min = 1510, max = 1583).

The distribution of the classes and example utterances translated from German are illustrated in Table 3.4.

Table 3.4

The distribution and examples of the classes for the VAM corpus

Class	Percentage (number)	Example
sad-bored	47.73 (452)	„ Yes, but that's not what I really want to do. “
angry-anxious	44.77 (424)	“I know the fact that I fucked up!”
relaxed-serene	5.28 (50)	“Uh! That cannot be true.”
happy-exciting	2.16 (21)	“We can reach a compromise: you no longer seize me and I drive no longer drunk.”

From Table 3.4 we may summarize that the corpus is highly unbalanced. Due to the small size of the corpus at whole (947 labeled utterances), it would be difficult to effectively recognize the smallest classes: relaxed-serene and happy-exciting. Nevertheless, the corpus is appropriate for a representation of text-based emotion recognition results in comparison with the existing audio-based and video-based emotion recognition results obtained on this corpus (Sidorov et al., 2014a; Sidorov et al, 2014b).

3.4. Summary

Totally, we consider five different speech-based corpora of utterance classification for evaluation: two corpora for topic categorization of user utterances and three corpora for user state recognition (two for verbal intelligence recognition and one for emotion recognition). A general comparison of these corpora is illustrated in Table 3.5.

Table 3.5

Comparison of corpora

Name	Task	Language	Classes	Number of utterances	Dictionary size
“Speech Cycle”	Topic categorization of user utterances	English	20	268,550	3,304
“Rafaeli”	Topic categorization of user utterances	English	5	337	842
Monologue corpus	User verbal intelligence recognition	German	2	100	3,138
Dialogue corpus	User verbal intelligence recognition	German	2	91	6,987
VAM	Text-based user emotion recognition	German	4	947	1,550

From Table 3.5 we observe that only the “Speech Cycle” has excessive data for machine learning. This implies that different validation techniques may be easily applied on this corpus. The other corpora have only small number of utterances or yield some specific classes with a very small number of utterances. Therefore, these corpora have some constraints for complex validation applications. Nevertheless, almost all of the state-of-the-art term weighting methods, dimensionality reduction techniques, and classification algorithms may be applied on all five considered corpora.

The numerical experiments on the corpora of two different languages (English and German) may demonstrate language-independence of the text classification approaches.

4. Comparative Study of Existing Utterance Classification Approaches

4.1. Introduction

The first aim of the thesis is to find the most effective combinations of the existing term weighting methods, dimensionality reduction techniques, and classification algorithms for the considered utterance classification problems. This implies to perform a comparative study of the techniques described in Chapter 2.

In a first stage of the evaluation, the training and test data sets have been prepared for numerical experiments that should be available for correct statistical analyses. Due to the different size of the considered corpora described in Chapter 3, we need to prepare the data:

- *„Speech Cycle“ corpus for topic categorization of user utterances*: 20 random separations of the whole corpus with the proportion 9:1 (90% of the corpus as a training set and 10% as a test set) for three different data configurations (see Section 3.2.1). The excessive data in the „Speech Cycle“ corpus provides a possibility to organize representative sets with such divisions.

- *„Rafaeli“ corpus for topic categorization of user utterances* (Section 3.2.2): Leave-One-Out (LOO) cross-validation. This corpus consists of the small number of the utterances (337). Therefore, LOO cross-validation is necessary: in this case the maximal amount of the data is used for training. This is especially important for the smallest classes because a standard division into a training and a test sets would not be representative for such small classes.

- *Corpora for verbal intelligence recognition* (Section 3.3.1): LOO cross-validation. The reason of a LOO cross-validation choice is the same as for the “Rafaeli” corpus: the corpora for verbal intelligence recognition contains a small number of utterances (100 for the monologue corpus and 91 for the dialogue corpus)

- *VAM corpus for emotion recognition* (Section 3.3.2): 20 random separations of the whole corpus with the proportion 9:1. This corpus contains enough amounts of the data (947 utterances) for such a methodology.

The next step is term weighting implementation. For each corpus, term weighting methods were implemented as a multiplication of term frequency (TF, equation 2.1,

section 2.4.1) and seven different corpus-based weighting techniques (unsupervised one – IDF and six supervised methods):

- *Inverse Document Frequency (IDF)* (Salton and Buckley, 1988, the details are described in Section 2.4.2).
- *Gain Ratio (GR)* (Debole and Sebastiani, 2004, see Section 2.4.3).
- *Confident Weights (CW)* (Soucy and Mineau, 2005, see Section 2.4.4).
- *Term Second Moment (TM2)* (Xu and Li, 2007, see Section 2.4.5).
- *Relevance Frequency (RF)* (Lan et al., 2009, see Section 2.4.6).
- *Term Relevance Ratio (TRR)* (Ko, 2012, see Section 2.4.7).
- *Novel Term Weighting (NTW)* (Gasanova et al., 2014a; Akhmedova et al., 2014, see Section 2.4.8).

The following techniques of dimensionality reduction were applied for the comparative study:

- *„Stop-words“ filtering in combination with stemming* (see Section 2.5.2). For these techniques, the special libraries in the program language *R* (”tm”, ”SnowballC”) were applied for English and German (https://www.r-project.org/). This procedure was performed before term weighting.
- *Weight-based term filtering* based on ignoring the predefined percentage of the terms with the lowest weights (see Section 2.5.3).
- *Weight-based feature transformation based on term clustering* with hierarchical agglomerative algorithm as an unsupervised learning (see Section 2.5.7, Algorithm 2.3).

Therefore, the comparative study includes a linguistic dimensionality reduction method („stop-words“ filtering in combination with stemming) and two numerical techniques: one based on feature selection (weight-based term filtering) and another one based on feature transformation (weight-based term clustering).

The following classification algorithms were chosen for applications based on the review of machine learning algorithms in Section 2.6:

- *The k-NN algorithm* (see Section 2.6.3) with weight distance (validation of *k* from 1 to 15 for „Speech Cycle“ and “Rafaeli” corpora, form 1 to 30 for other corpora).
- *The SVM-based algorithm Fast Large Margin (SVM-FLM)* (Fan et al., 2008, see Section 2.6.5).
- *The Rocchio classifier* (the version which is described by equation 2.20, see Section 2.6.4). The Rocchio classifier was tested using three different metrics: the „taxi cabine“ metric (equation 2.26), the Euclidean metric (equation 2.27), and the cosine

similarity (equation 2.28). This algorithm showed low results for the “Speech Cycle” corpus; therefore, the Rocchio classifier was not applied for other corpora.

As a main classification quality criterion, we used the macro *F1*-score (Goutte and Gaussier, 2005) which is appropriate for classification problems with unbalanced class distribution. The *F1*-score is calculated as a harmonic mean of precision and recall. The macro *F1*-score means that we use average values of the precision and the recall of classes:

$$F1 = 2 \cdot \frac{\overline{precision} \cdot \overline{recall}}{\overline{precision} + \overline{recall}}, \tag{5.1}$$

$$\overline{precision} = \frac{\sum_{i=1}^{n} precision_i}{n} = \frac{\sum_{i=1}^{n} \frac{TP_i}{TP_i + FP_i}}{n}, \tag{5.2}$$

$$\overline{recall} = \frac{\sum_{i=1}^{n} recall_i}{n} = \frac{\sum_{i=1}^{n} \frac{TP_i}{TP_i + FN_i}}{n}, \tag{5.3}$$

where n is a number of classes for a classification problem, TP_i are „true positive“ elements of the i^{th} class (elements of the i^{th} class that were correctly classified), FP_i are „false positive“ elements of the i^{th} class (elements that were classified as elements of the i^{th} class but they do not belong to the i^{th} class), FN_i are „false negative“ elements of the i^{th} class (elements of i^{th} class that were not correctly classified).

We also used a standard accuracy (percentage of the correctly classified elements form the test set) as an additional classification criterion.

The following procedure requires a validation: tuning of k for the k-NN algorithm. The validation sets were obtained from the training sets in the following ways:

- *The „Speech Cycle“, the „Rafaeli“, and the “VAM” corpora*: the trainings sets were divided into two parts with the proportion 4:1. The larger part was used as a training set for the validation; the smaller one was used as a test set for the validation. The first stage of validation always requires more data than the second one. This explains the chosen proportion of dividing.

- *The corpora for verbal intelligence recognition*: the LOO cross-validation was organized for each training set for the validation (“the LOO cross-validation inside the LOO cross-validation”). These corpora contain a very small number of utterances (100 and 91 correspondingly). In this case, the LOO cross-validation inside the LOO cross-validation is the only possible way for the validation.

Due to the excessive data in the “Speech Cycle” corpus, most of the experiments have been performed on this corpus. Some results and conclusions obtained on the “Speech Cycle” have been generalized for the other corpora, i.e. the Rocchio classifier which obtained low classification results on the “Speech Cycle” corpus was not applied for the other corpora.

Statistical analyses of the numerical results are performed with *t*-test with the confidence probability equals 0.95.

4.2. Implementation of Algorithms

The term weighting methods and the dimensionality reduction techniques were implemented with *C#*, *Python*, and *R* scripts.

The free software *RapidMiner* version 5.0 (https://rapidminer.com) with a standard setting was used for *k*-NN and SVM-FLM applications (Shafait et al, 2010). The Rocchio classifier was implemented with the programming language *Python* (https://www.python.org/).

The standard settings of the *k*-NN and the SVM-FLM algorithms in *RapidMiner* 5.0 are presented in Tables 4.1 and 4.2.

Table 4.1

The standard settings of the *k*-NN algorithm in *RapidMiner* 5.0

Parameter	Value
k	*Validation procedure*
weighted vote	True
measure types	MixedMeasures
mixed measure	MixedEuclideanDistance

Table 4.2

The standard settings of the SVM-FLM algorithm in *RapidMiner* 5.0

Parameter	Value
Solver	L2 SVM Dual
C	1.0
Epsilon	0.01
class weights	False
use bias	True

4.3. Topic Identification of User Utterances

4.3.1. Results on the „Speech Cycle“ Corpus

4.3.1.1. Full Dimensionality

At the first stage, different term weighting methods (IDF, GR, CW, TM2, RF, TRR, and NTW) and classification algorithms (*k*-NN, SVM-FML, and the Rocchio classifier) were tested for three data configurations of the „Speech Cycle“ corpus (see Section 3.2.1) without dimensionality reduction methods. This implies to use the total set of unique terms obtained from the training set or, in other words, the entire dictionary of the text classification problem.

Figures 4.1-4.3 demonstrate numerical results with the *k*-NN algorithm and SVM-FML for three different data configurations in terms of *F1*-score (average values calculated by twenty different divisions into training and test sets).

From the results illustrated in Figures 4.1-4.3 we can observe that the classification effectiveness significantly varies for different term weighting methods. *k*-NN and SVM-FML also obtained different results as the classification algorithms. However, the best results with *k*-NN and SVM-FML have no statistically significant difference between each other for all three data configurations. It is interesting to notice that IDF is the best term weighting method with SVM-FML and the worst one with *k*-NN.

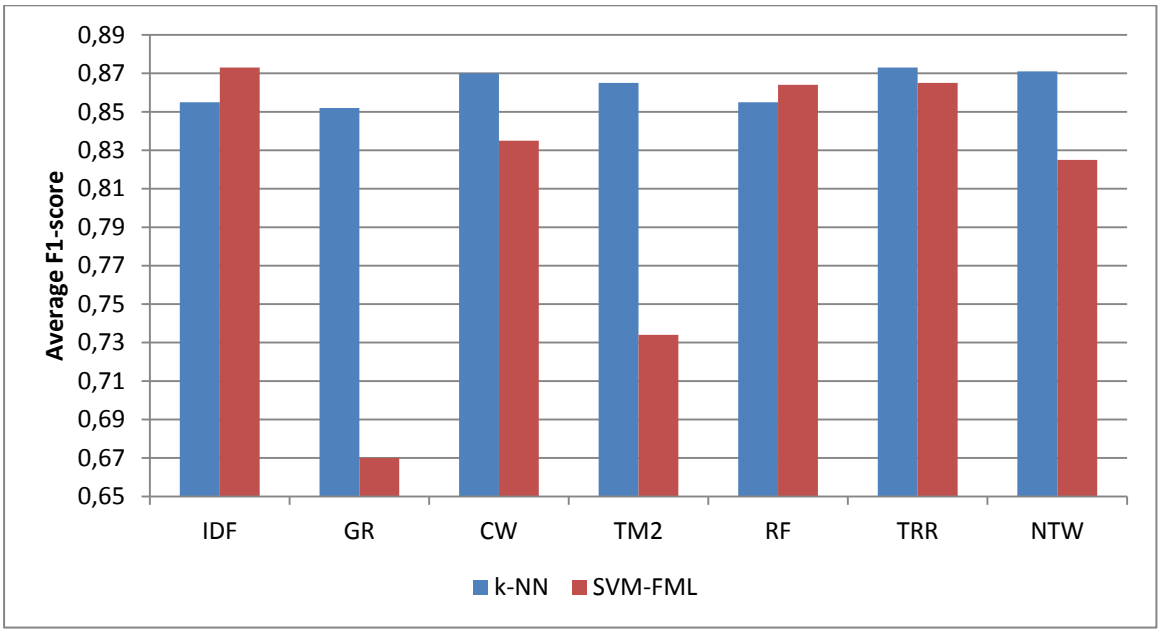

Figure 4.1 – Numerical results with all terms for data configuration 1

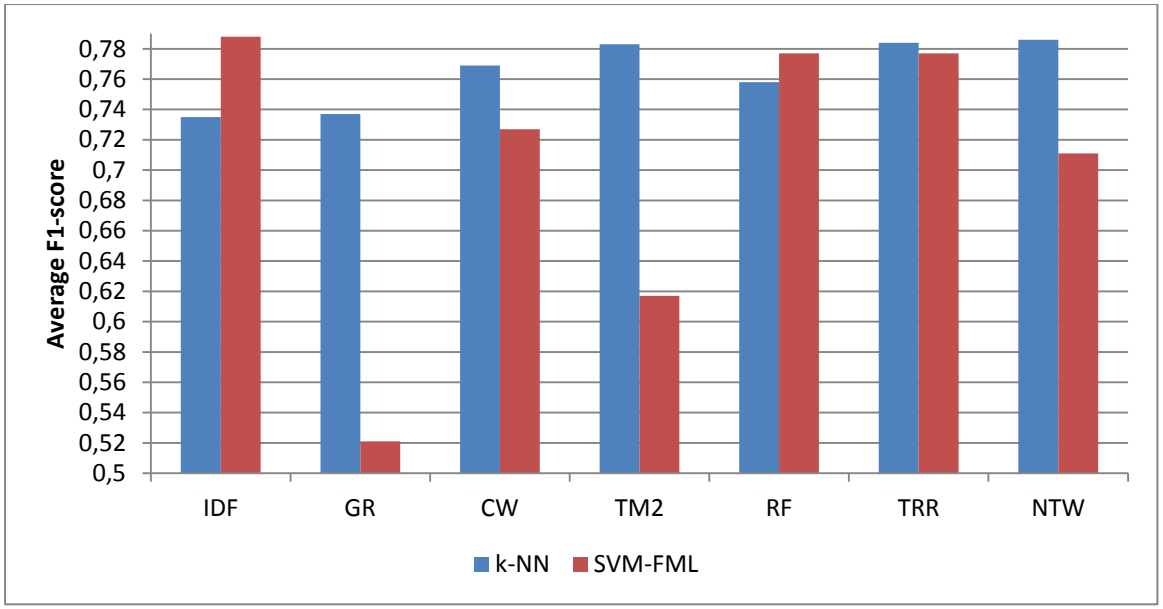

Figure 4.2 – Numerical results with all terms for data configuration 2

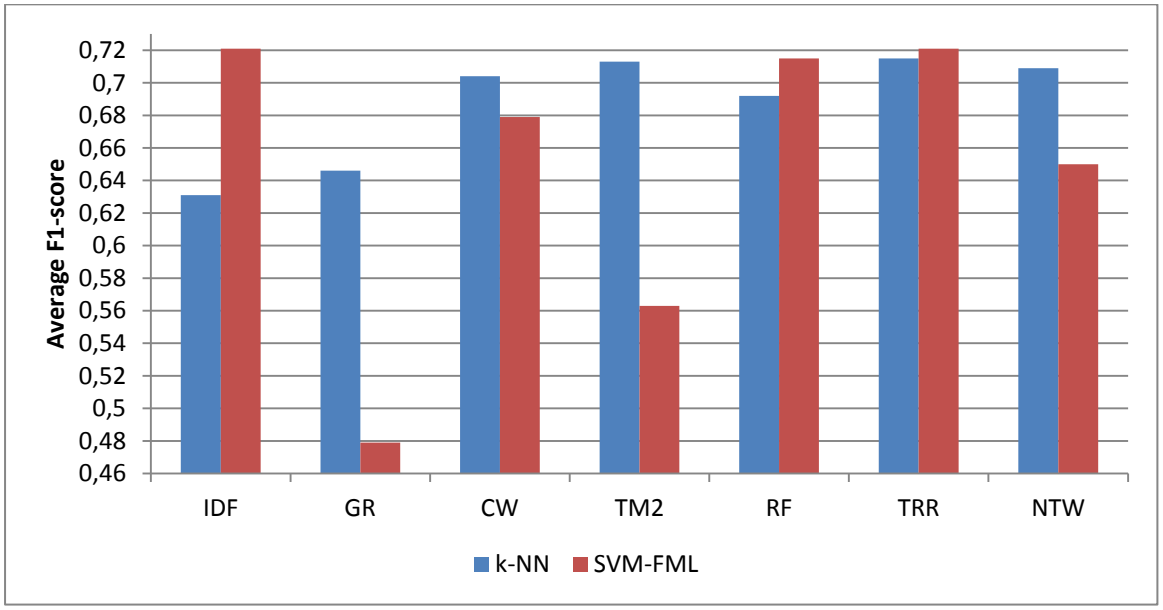

Figure 4.3 – Numerical results with all terms for data configuration 3

The Rocchio classifier was tested with three different metrics: „taxi cabine“, Euclidean, and cosine similarity metrics. The numerical experiments have shown that the Euclidean metric performs best with all term weighting methods. Figure 4.4 presents the results of the the Rocchio classifier with the Euclidean metric.

The numerical results presented in Figure 4.4 show a low performance of the Rocchio classifier in comparison with *k*-NN and SVM-FML. This is consistent with the review of classification algorithms in Section 2.6. Therefore, the Rocchio classifier was

not applied for the furthering experiments on the „Speech Cycle“ corpus and was not applied at all for the other corpora.

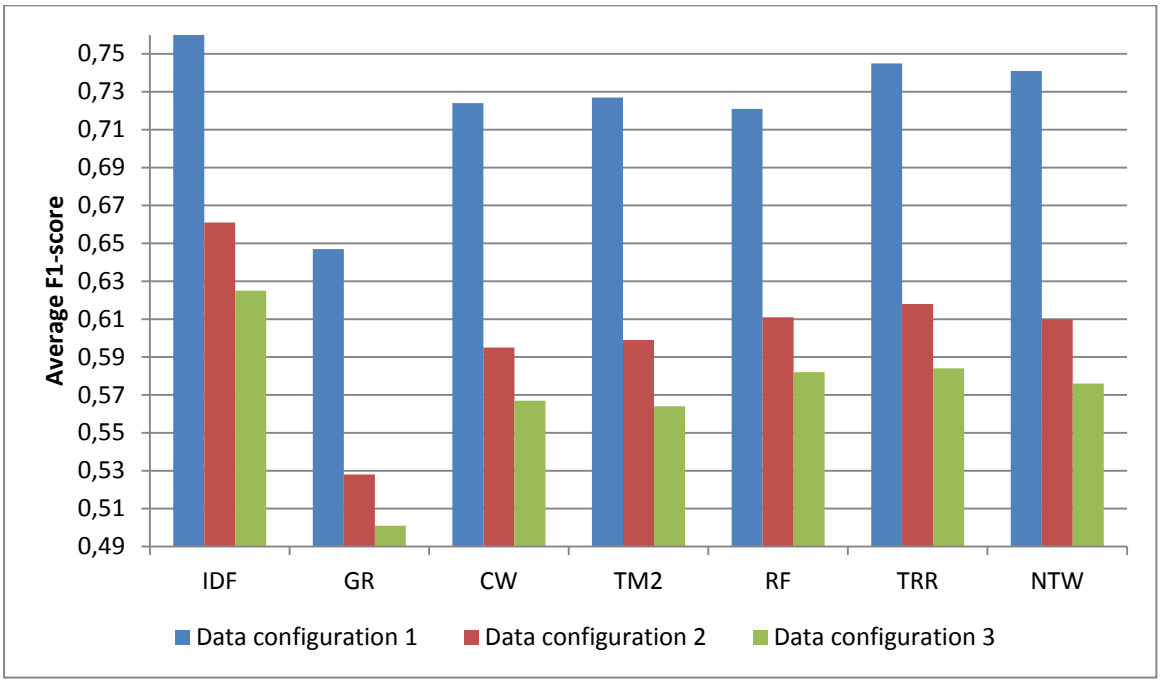

Figure 4.4 – Numerical results with all terms for the Rocchio classifier

F1-scores vary significantly more for data configurations 2 and 3 than for data configuration 1. This confirms our assumption in Section 3.2.1 that data configurations 2 and 3 are more appropriate for the comparative study of different text classification approaches than data configuration 1. Nevertheless, data configuration 1 describes the situation that is closest to real natural language call routing.

The GR method may be recognized as one of the worst term weighting with all considered classification algorithms.

The best combinations of term weighting methods and classification algorithms that have no statistically significant differences between each other based on *t*-test are presented in Table 4.3. These best combinations were separately found for different data configurations and for different classification criteria (*F1*-score and accuracy). A name of a combination begins with a classification algorithm (*k*-NN or SVM-FML). Then a term weighting method is presented. For example, the first row of Table 4.3 means that for data configuration 1 the following combinations of classification algorithms and term weighting methods are the best ones in terms of *F1*-score: *k*-NN with CW, *k*-NN with

TRR, *k*-NN with NTW, and SVM-FML with IDF. These combinations have no statistically significant differences between each other.

Table 4.3

The best combinations of classification algorithms and term weighting methods

Data configuration 1, *F1*-score	*k*-NN+CW, *k*-NN+TRR, *k*-NN+NTW, SVM-FLM+IDF
Data configuration 1, accuracy	*k*-NN+TM2, *k*-NN+TRR, *k*-NN+NTW, SVM-FLM+IDF
Data configuration 2, *F1*-score	*k*-NN+TM2, *k*-NN+TRR, *k*-NN+NTW, SVM-FLM+IDF
Data configuration 2, accuracy	*k*-NN+TM2
Data configuration 3, *F1*-score	*k*-NN+TM2, *k*-NN+TRR, SVM-FLM+IDF, SVM-FLM+TRR
Data configuration 3, accuracy	*k*-NN+TM2, SVM-FLM+IDF, SVM-FLM+RF, SVM-FLM+TRR

The analysis of Table 4.3 allows to conclude that for all three data configurations the following combinations are the best in terms of *F1*-score: *k*-NN+TRR, *k*-NN+TM2, and SVM-FLM+IDF.

4.3.1.2. Weight-based Term Filtering

Term weighting methods provide a natural feature selection method. It is possible to ignore terms with the lowest weights because the term weight represent the importance of the corresponding term (see Section 2.5.3).

Since the distribution of terms and weights can vary significantly for different classes, it is more appropriate to perform class-based term filtering, i.e. deleting the corresponding number of terms with the lowest weights for each class independently. Therefore, it is necessary to assign each term from the dictionary to one corresponding class. During supervised term weighting methods CW, GR, RF, NTW, and TRR such an assignment is performed automatically. With IDF and TM2 we can also assign one class for each term using the relative frequency of the word in classes (equation 2.26).

For RF, TM2, and TRR methods we decreased the dictionary size from 100% to 10% with the interval equals 10. This means deleting the corresponding number of the terms with the lowest weights.

IDF and NTW yield a significant number of terms with the equal highest value. For IDF the highest weight means that the term occurs only in one document from the training sample, for NTW it means that the term occurs only in documents of one class. Therefore, for these two methods we used different constraints for the value of weights; the predefined percentage of the dictionary size is not appropriate for NTW and IDF.

CW and GR yield a significant number of terms with zero weights. It means that these two methods provide feature selection automatically. For our problem we have 43.5% (1,436 terms) of the dictionary as terms with non-zero weights for GR and 20.4% (673 terms) for CW on the average. We also decreased the size of the dictionary for CW and GR with the class-based approach.

The results with weight-based term filtering for data configuration 1 are presented in Figures 4.5 and 4.6 for *k*-NN and SVM-FLM respectively. A similar behavior is observed for data configurations 2 and 3 (see Appendix A.2).

From these figures we conclude that almost in all cases a significant decrease in classification results is observed. There is not any statistically significant decrease of *F1*-score with this feature selection only for GR (one of the worst methods in terms of *F1*-score without dimensionality reduction, see Section 4.3.1.1) and for TRR with 90% features. Especially inappropriate results of weight-based term filtering are obtained with IDF: the *F1*-score is lower than 0.2 even with 90% features. Therefore, we may conclude that weight-based term filtering is not appropriate as a dimensionality reduction on the considered corpus. The reason lies in the fact that only very short utterances for classification (not more than 20 words) are used in the "Speech Cycle" corpus. Every word and every form of a word in the utterance may be useful for classification.

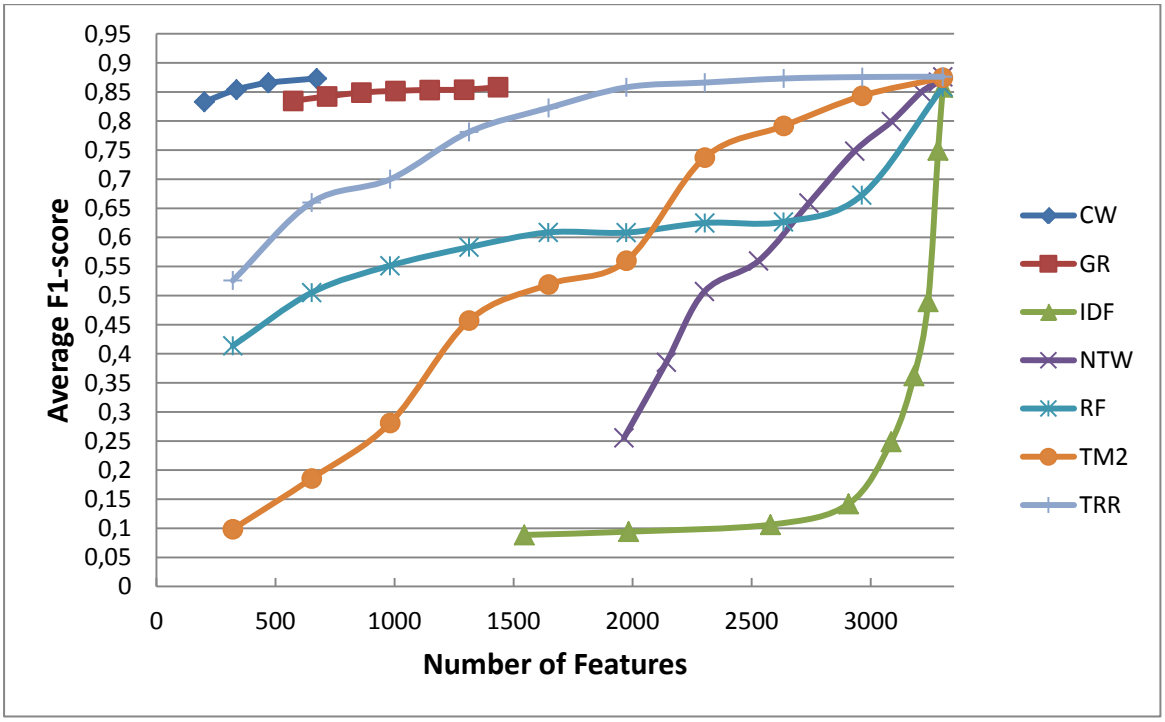

Figure 4.5 – Weight-based term filtering with *k*-NN for data configuration 1

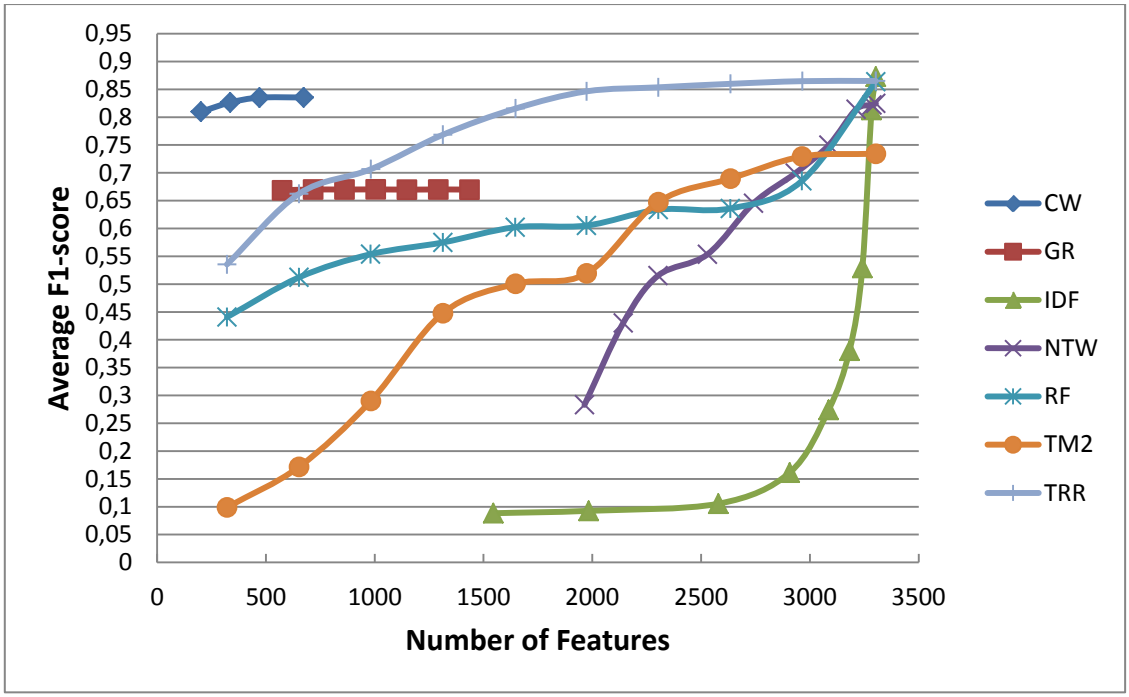

Figure 4.6 - Weight-based term filtering with SVM-FLM for data configuration 1

4.3.1.3. "Stop-word" Filtering + Stemming

The combination of "stop-words" filtering and stemming reduces the average dimensionality from 3,304 to 2,482 (75.1% of the original dictionary); for GR from 1,436 to 1,076 (74.9%); for CW from 673 to 507 (75.3%).

A comparison of the best classification results with "stop-words" filtering + stemming and the best results without dimensionality reduction (with all terms, see Section 4.3.1.1) is presented in Figures 4.7 and 4.8 for *k*-NN and SVM-FML respectively.

The results illustrated in Figures 4.7 and 4.8 show that a significant decrease of the classification results (*F1*-score) is observed in all cases with "stop-word" filtering + stemming in comparison with the results obtained without dimensionality reduction. The best term weighting methods with "stop-word" filtering + stemming are the same as obtained in the case without dimensionality reduction (see Section 4.3.1.1 and Appendix A.4).

Generally, we may conclude based on the results in Sections 4.3.2 and 4.3.3 that term filtering approaches (weight-based term filtering or „stop-words" filtering) are not appropriate for the considered utterance classification problem. This means that useful information is lost after these procedures. Feature filtering can increase classification performance in the case of text classification problems with written documents, which are

larger than user utterances, and with a redundant dictionary (Rogati and Yang, 2002; Gabrilovich and Markovitch, 2004) but not for short user utterances. This conclusion confirms the importance of the first aim that has been formulated for this thesis: to perform a comparative study of text classification approaches for utterance classification. Such procedures as “stop-word” filtering or stemming are a “gold standard” for text classification. However, they are not appropriate for classification of short utterances.

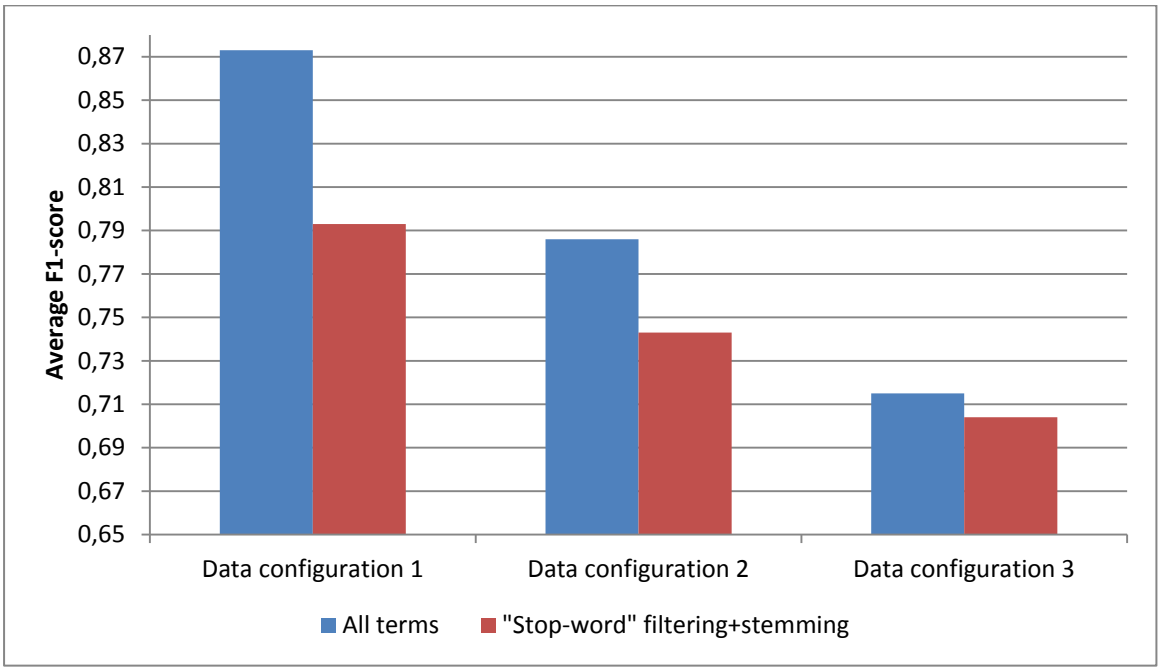

Figure 4.7 – Results of “stop-word” filtering + stemming with *k*-NN

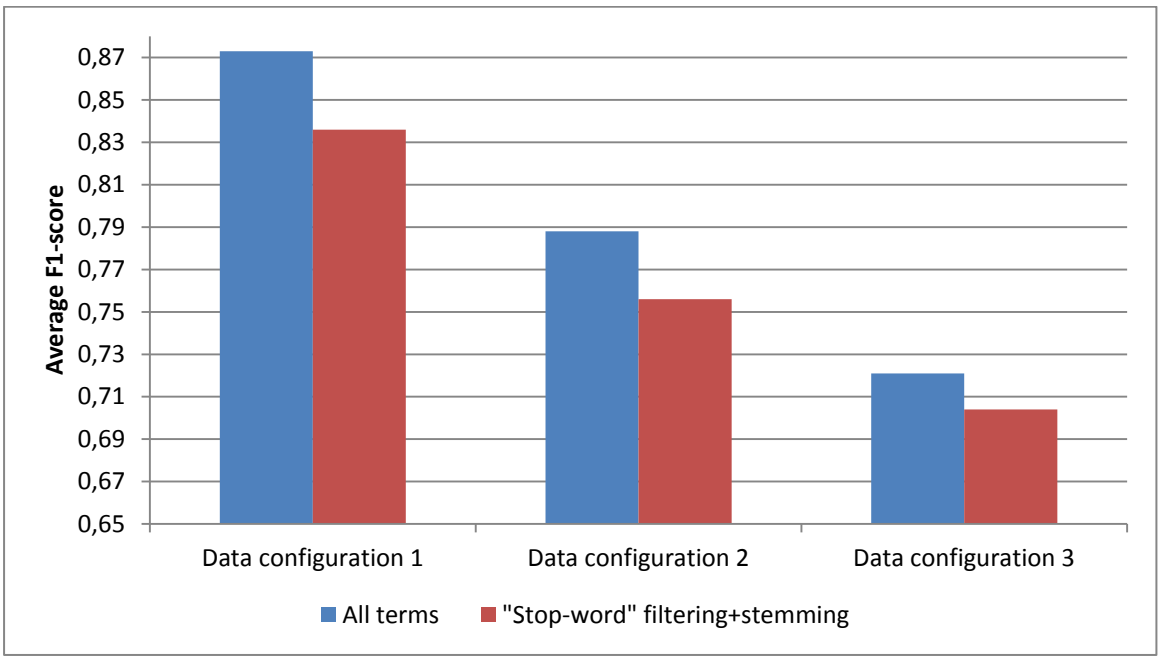

Figure 4.8 – Results of “stop-word” filtering + stemming with SVM-FML

4.3.1.4. Numerical Results with Feature Transformation Based on Term Clustering

The idea of using class-based language models by applying term clustering was proposed in (Momtazi and Klakow, 2009). It is possible to use the term clustering in the dictionary for dimensionality reduction. In this case we suggest preprocessing our dictionary such that words of equal or similar weights are placed in the same cluster and one common weight (a new feature) will be assigned to all words in this cluster.

We performed term clustering which is described in Algorithm 2.4 (see Section 2.5.7). In order to reduce the dictionary size we take hierarchical agglomerative clustering (Ward, 1963) with Euclidean metric. We set the maximal number of clusters for each class 10, 20, 50 and 100.

The results with feature transformation based on term clustering are presented in Figures 4.9 and 4.10 with *k*-NN for data configurations 1 and 3 correspondingly. The results with SVM-FML and with data configuration 2 demonstrate a very similar behavior (see Appendix A.3).

We may conclude that this technique is much better than term filtering. It is possible to decrease the dimensionality significantly without significant changes in classification results.

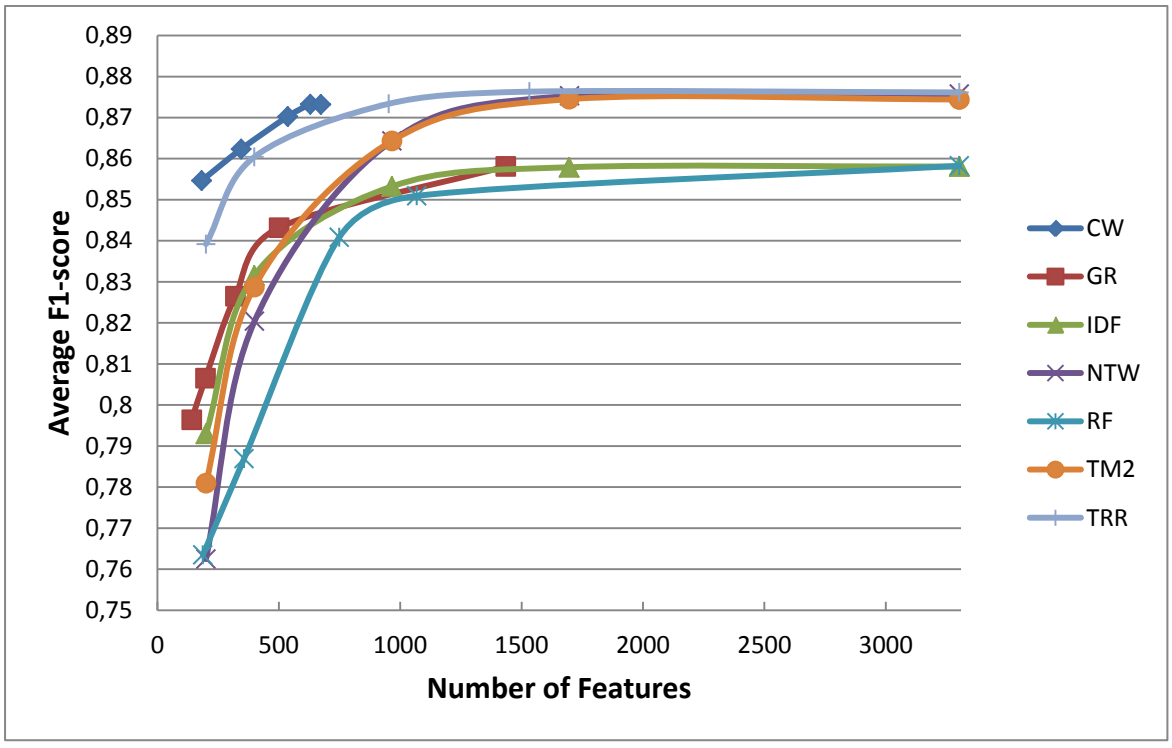

Figure 4.9 - Feature transformation based on term clustering with *k*-NN for data configuration 1

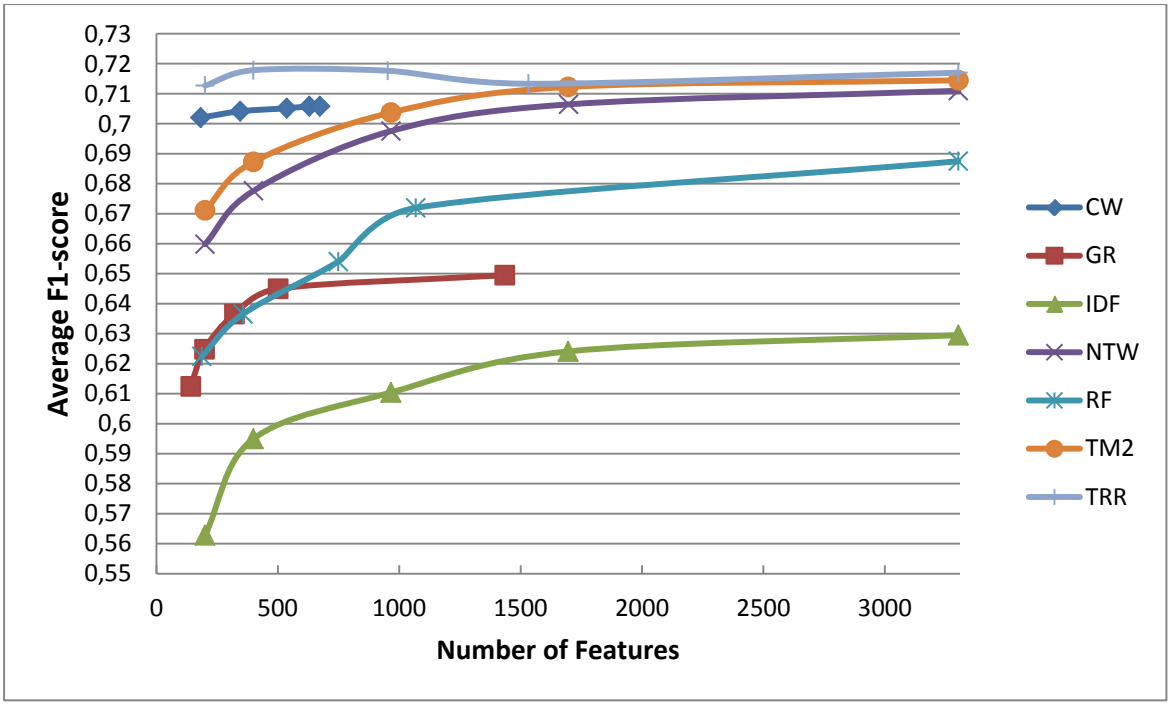

Figure 4.10 - Feature transformation based on term clustering with *k*-NN for data configuration 3

We obtain the best result in terms of *F1*-score for data configuration 1 only with 1532 features or 46.4% of the original dimensionality (TRR, *k*-NN), for data configuration 2 only with 953 features or 28.8% of the original dimensionality (TRR, *k*-NN) and data configuration 3 only with 399 features or 12.1% of the original dimensionality (TRR, *k*-NN) when feature transformation based on term clustering is applied.

Based on the results in Sections 4.3.1.2 and 4.3.1.4 we may conclude that TRR seems to be the best term weighting methods for dimensionality reduction and it provides the best classification accuracy, especially in combination with *k*-NN.

4.3.2. Results on the „Rafaeli“ Corpus

Due to the small size of „Rafaeli“ corpus, we have performed the Leave-One-Out (LOO) cross-validation for feature extraction, dimensionality reduction, and classification.

The GR method automatically yields 43.6% of the ordinary dictionary as terms with non-zero weights (367 terms) and the CW method automatically yields 6.0% terms with non-zero weights (51 term) on the “Rafaeli” corpus.

The numerical results with *k*-NN and SVM-FML in the case of using all terms (without dimensionality reduction) are presented in Figure 4.11.

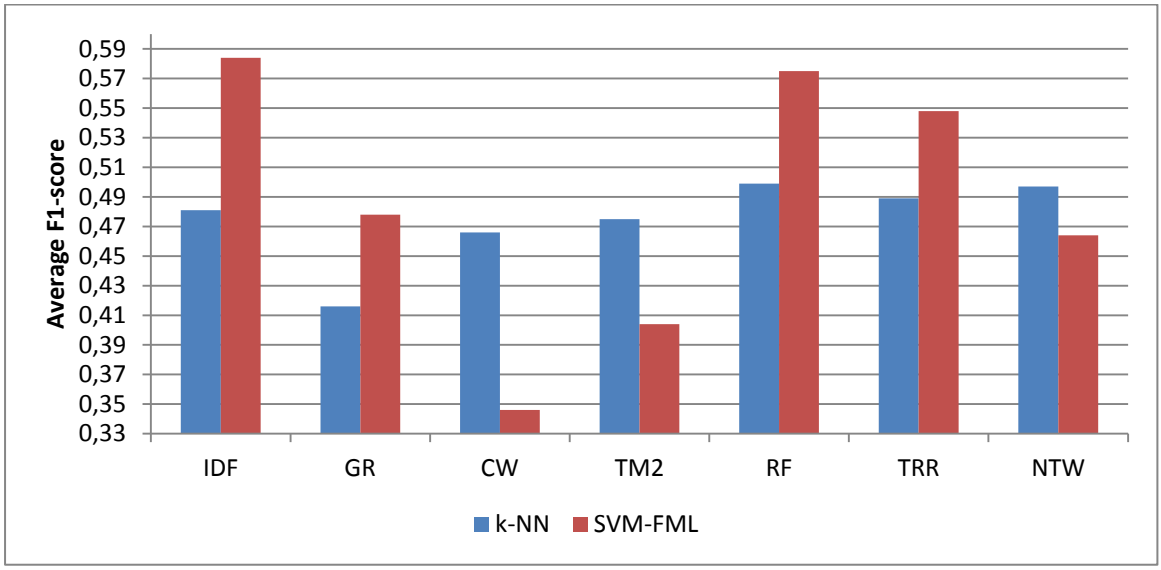

Figure 4.11 – Numerical results with all terms on the “Rafaeli” corpus

Figure 4.11 demonstrates that the best result with SVM-FML outperforms the best result with *k*-NN on this corpus. However, *F1*-scores vary significantly more for SVM-FML than for *k*-NN with different term weighting methods. The best term weighting method with SVM-FML is IDF; the best ones with *k*-NN are RF, TRR, and NTW.

Figure 4.12 illustrates the results for the best individual term weighting methods (“The best TWM”) with “stop-word” filtering + stemming in comparison with the results obtained without dimensionality reduction. “Stop-word” filtering + stemming provides a dimensionality reduction from 842 to 604 terms (71.7% of the original dictionary).

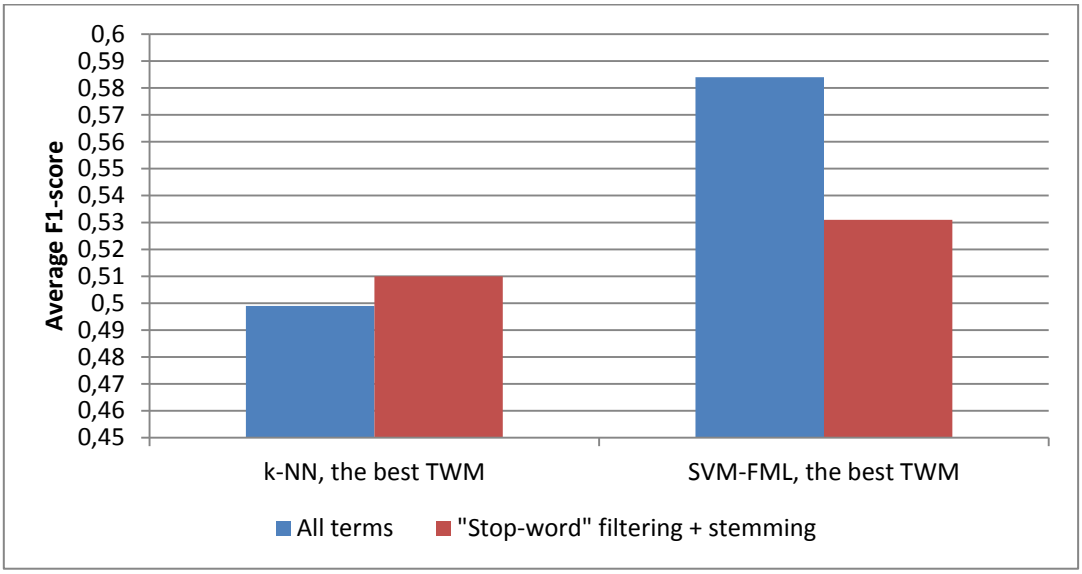

Figure 4.12 – Results of “stop-word” filtering + stemming on the “Rafaeli” corpus

„Stop-word“ filtering in combination with stemming obtains a slight improvement of *F1*-score with *k*-NN. However, the best result obtained by SVM-FML decreases with this method. Therefore, this dimensionality reduction seems to be not appropriate for the considered utterance classification problem. The same conclusion holds for the “Speech Cycle” corpus.

Figures 4.13 – 4.16 show the results with term weight-based feature filtering and feature transformation based on weight-based term clustering. The schemes of numerical experiments with these dimensionality reduction methods are the same as for the “Speech Cycle” corpus. The CW method was not tested for feature selection and feature transformation based on clustering because this method yields a very small number of features with non-zero values automatically. The GR method was tested only with feature transformation based on clustering.

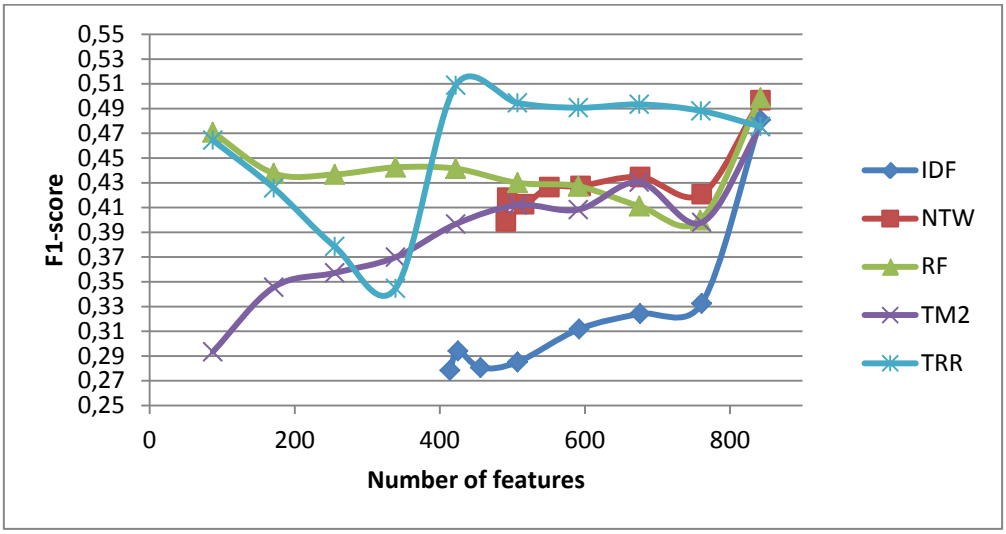

Figure 4.13 – Weight-based term filtering with *k*-NN on the “Rafaeli” corpus

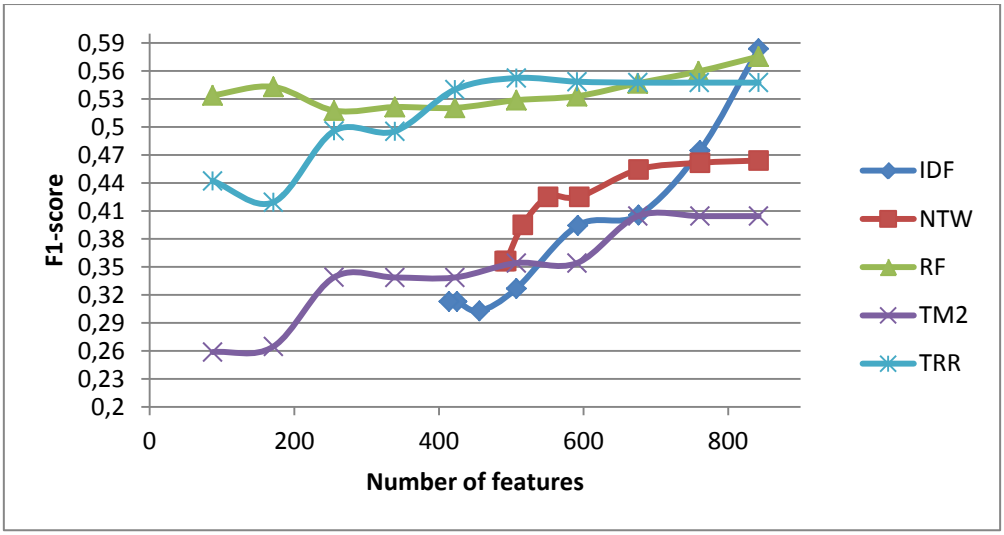

Figure 4.14 – Weight-based term filtering with SVM-FLM on the “Rafaeli” corpus

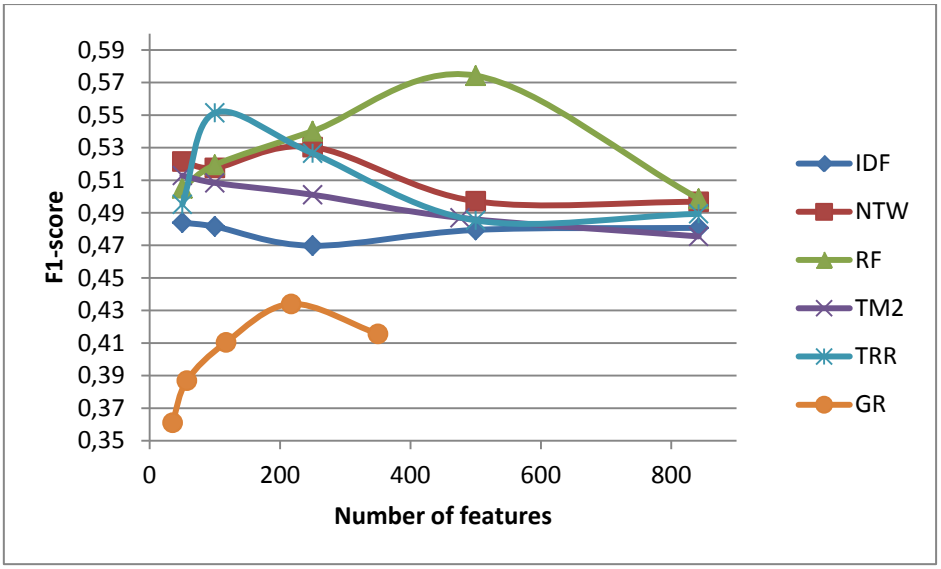

Figure 4.15 – Feature transformation based on term clustering with *k*-NN on the "Rafaeli" corpus

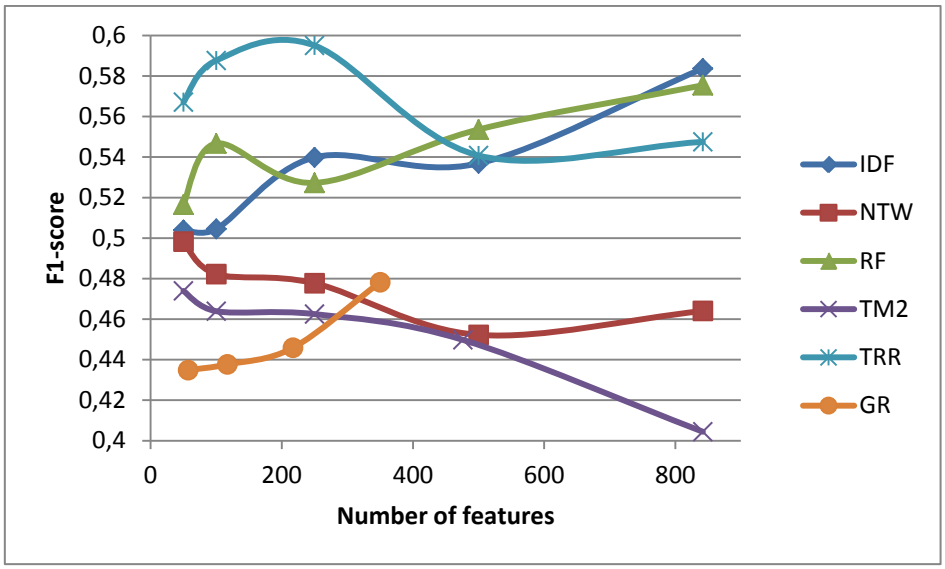

Figure 4.16 – Feature transformation based on term clustering with SVM-FML on the "Rafaeli" corpus

Based on the results presented in Figures 4.13-4.16 we can make the following conclusions that are very similar to the ones made for the „Speech Cycle" corpus:

- Regarding dimensionality reduction, feature transformation based on term clustering is more appropriate compared to term weight-based feature filtering (the same result we observe on the „Speech Cycle" corpus). It is possible to significantly improve the classification results with feature transformation based on term clustering. Weight-

based term filtering results in a significant decrease of classification results except for TRR (Figures 4.13 and 4.14).

- TRR seems to be the most appropriate term weighting for dimensionality reduction techniques (weight-based term filtering and weight-based term clustering). It provides the best classification result with weight-based term clustering and SVM-FML among the all other considered approaches (see Figure 4.16: the *F1*-score is almost equal 0.6).

Generally, the results on the "Rafaeli" corpus are worse than the results on the "Speech Cycle" corpus due to the deficient amount of the data for machine learning. Nevertheless, the best results (*F1*-scores up to 0.6) are encouraging for the five-class classification problem with hundreds of features and the scarce data for learning. Based on the results obtained on the „Speech Cycle" and the „Rafaeli" corpora we may conclude that the existing text classification are appropriate for domain detection of user utterances.

4.4. User State Recognition

4.4.1. Results on the Corpora for Verbal Intelligence Recognition

For feature extraction, feature selection, and classification we used Leave-One-Out (LOO) cross-validation with both corpora for verbal intelligence recognition. The first stage of the numerical experiments is establishing a dictionary based on the training documents. The dictionary size varies from 3,092 to 3,183 for the monologue corpus (average value equals 3,138) and from 6,892 to 7,082 (average value equals 6,987) for the dialogue corpus.

SVM-FML showed a rather weak performance for the considered problems of verbal intelligence recognition. For instance, without dimensionality reduction for each term weighting method SVM-FML classified all monologues into the class of higher verbal intelligence. This means that the classifier has the lowest possible performance. Therefore, we illustrate only the results obtained by *k*-NN with weight distances on the corpora for verbal intelligence recognition.

We performed validation of *k* form 1 to 30 based on LOO cross-validation (the LOO cross-validation inside the LOO cross-validation, because we use LOO cross-validation both for testing and for validation).

The GR method yields 1068 terms with non-zero values for the monologue corpus (34.0% of the original dictionary) and 2202 terms with non-zero values for the dialogue

corpus (31.5%) on average. The CW method yields 3.8 (0.12% of the original dictionary) and 4.7 (0.07% of the original dictionary) non-zero terms in average for the monologue and the dialogue corpora respectively.

The combination of "stop-word" filtering and stemming provides dimensionality reduction from 3,138 to 2,771 terms (88.3%) for the monologue corpus and from 6,987 to 6,526 (93.4%) for the dialogue corpus.

Table 4.4 contains the numerical results in terms of *F1*-score and accuracy with *k*-NN for the monologue corpus for two cases: without dimensionality reduction and with „stop-words" filtering + stemming. Table 4.5 illustrates the corresponding results for the dialogue corpus.

Table 4.4

Numerical results with *k*-NN on the monologue corpus

Term weighting method	**Without dimensionality reduction**		**"Stop-word" filtering + stemming**	
	***F1*-score**	**Accuracy**	***F1*-score**	**Accuracy**
IDF	0.54	0.60	**0.64**	**0.62**
GR	0.35	0.53	0.48	0.58
CW	0.58	0.60	0.33	0.45
TM2	0.38	0.60	0.49	0.58
RF	0.56	0.61	0.39	0.49
TRR	0.49	0.58	0.48	0.59
NTW	0.43	0.53	0.46	0.57

Table 4.5

Numerical results with *k*-NN on the dialogue corpus

Term weighting method	**Without dimensionality reduction**		**"Stop-word" filtering + stemming**	
	***F1*-score**	**Accuracy**	***F1*-score**	**Accuracy**
IDF	0.53	0.59	0.57	0.60
GR	0.38	0.48	0.48	0.55
CW	**0.76**	**0.76**	0.66	0.67
TM2	0.37	0.59	0.37	0.59
RF	0.53	0.58	0.52	0.58
TRR	0.50	0.57	0.57	0.60
NTW	0.58	0.62	0.60	0.62

The best result on the monologue corpus is obtained by IDF with "stop-word" filtering + stemming (*F1*-score equals 0.64, accuracy – 0.62). However, these results are pretty low for the two-class classification problem. Therefore, it is difficult to conclude that "stop-word" filtering + stemming significantly improves the classification results.

The best result on the dialogue corpus is obtained by the CW method without additional dimensionality reduction.

Figures 4.17 and 4.18 demonstrate the results of weight-based term filtering and weight-based term clustering on the monologue corpus. Figures 4.19 and 4.20 present the same results on the dialogue corpus. Only in the case of term clustering on the monologue corpus (Figure 4.18) we observe an improvement of the best classification results.

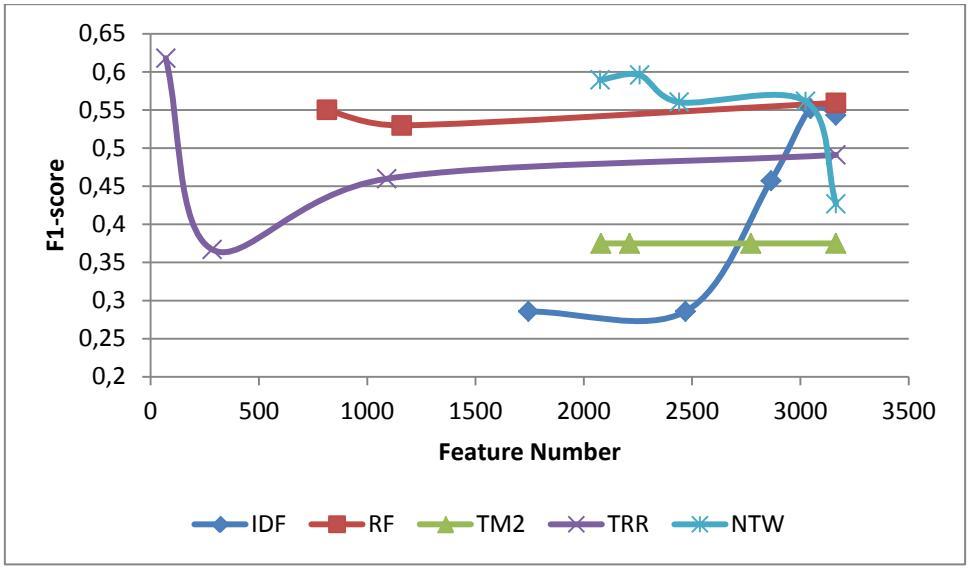

Figure 4.17 – Weight-based term filtering with *k*-NN on the monologue corpus

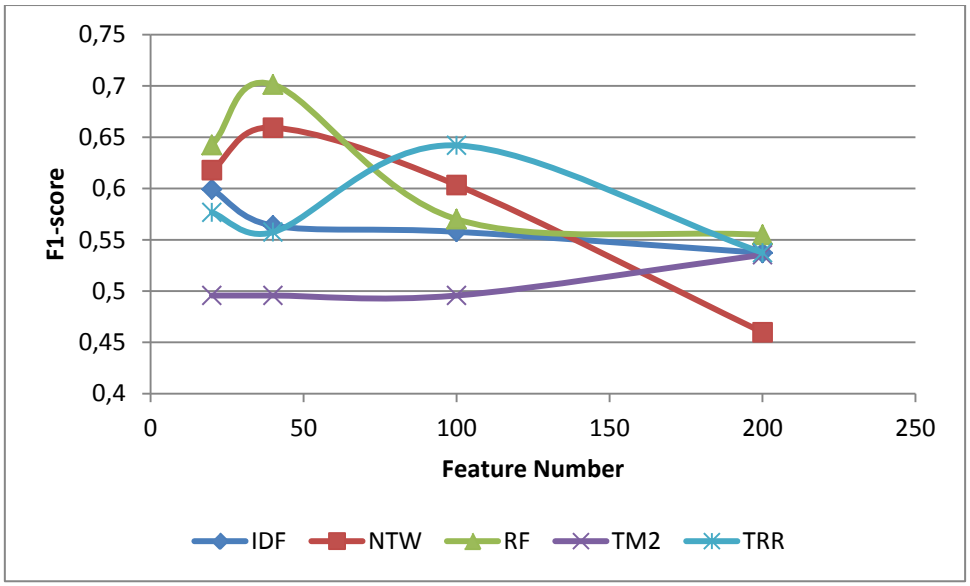

Figure 4.18 – Feature transformation based on term clustering with *k*-NN on the monologue corpus

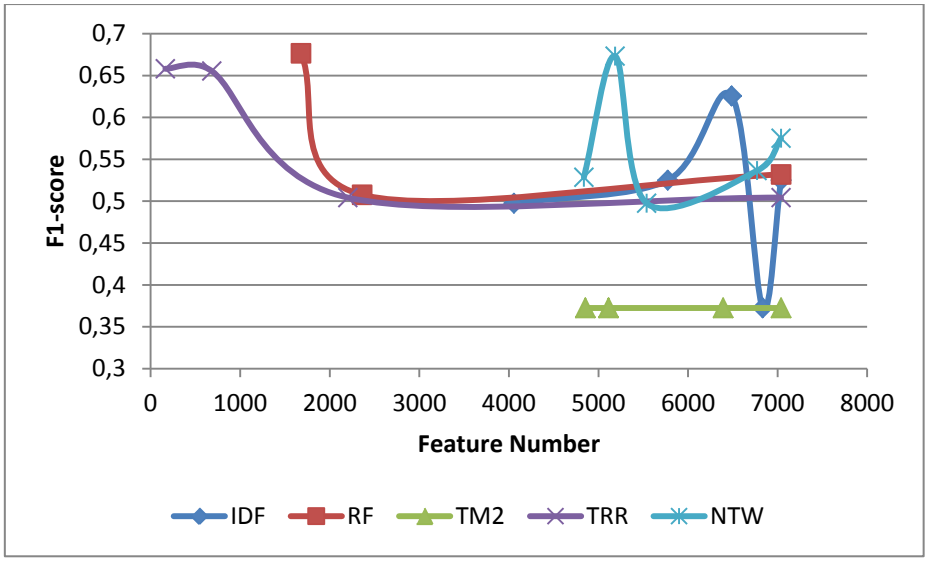

Figure 4.19 – Weight-based term filtering with *k*-NN on the dialogue corpus

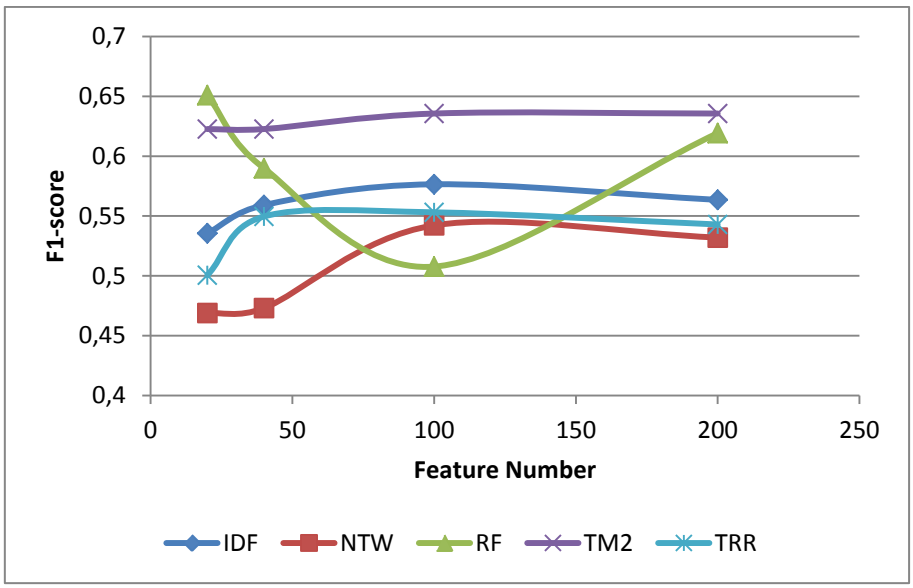

Figure 4.20 – Feature transformation based on term clustering with *k*-NN on the dialogue corpus

Almost all *F1*-scores presented in Figures 4.17 – 4.20 are low for the two-class classification problems. The majority of the *F1*-scores ranges in the interval [0.45; 0.65]. These values are close to the results obtained if a random choice is applied. Therefore, it is difficult to find any deterministic regularity in Figures 4.17 – 4.20. It seems that the values on these Figures behave stochastically.

The best *F1*-scores obtained on the monologue corpus are relatively low for the two-class classification problem (the best *F1*-score equals 0.70 with RF as a term

weighting method and feature transformation based on term clustering as a dimensionality reduction technique). The results on the dialogue corpus are better than on the monologue corpus. The best *F1*-score on the dialogue corpus equals 0.76 and the best classification accuracy also equals 0.76 by the CW method. These values are encouraging.

We can guess why the results for the dialogue corpus are better than for the monologue corpus. At first, the documents for the dialogue corpus are larger and the dictionary is twice as large as for the monologue corpus. Therefore, more linguistic information is available for verbal intelligence recognition. As a second explanation we can suppose that verbal intelligence is expressed more clearly during a dialogue.

The best result for the dialogue corpus is obtained by the Confident Weights method (CW). It yields zero values for a significant number of terms automatically. In our case the number of terms with non-zero values is extremely low. For the dialogue corpus the average number of terms with non-zero weights equals 4.7 (min = 3, max = 6). The CW method also determines the most appropriate class for each term from the dictionary automatically. For the dialogue corpus all terms with non-zero values belong to the second class which implies a higher verbal intelligence.

In the situation with extremely small number of terms yielding non-zero values, some samples have all features with zero values. The classification algorithm defines such a sample as an element of the first class (lower verbal intelligence) automatically. Therefore, the best term weighting method (CW) for the dialogue corpus determines only a small number of words that characterize the class of higher verbal intelligence.

We can explain why we obtain a very small number of terms with non-zero values. These terms must satisfy two contradictory conditions:

1) The terms should be well-known and in general use. Specific words (i.e. professional terms) can be used by a restricted number of people even with high verbal intelligence. This condition is especially critical for our very small corpora with not more than 100 different speakers.

2) The terms should characterize high verbal intelligence. In our case such words are ”missing”, ”higher”, ”syllabus” (translated from German).

We suppose that larger corpora for verbal intelligence recognition than the considered ones may allow to increase the number of significant terms and classification effectiveness. It is possible to use written text for creating larger corpora. Maybe it would also be possible to increase the number of classes for categorizing verbal intelligence.

4.4.2. Results on the VAM Corpus for Emotion Recognition

The numerical experiments for text-based emotion recognition were performed for 20 different divisions of the original database into training and test sets. As classification algorithms, *k*-NN and SVM-FML were applied.

The numerical results on the previous corpora showed that the combination of „stop-word“ filtering and stemming is not effective for speech-based text classification problems with short user utterances: a slight dimensionality reduction with this technique results in a significant decrease of classification results. Therefore, this combination was not applied for the VAM corpus of text-based emotion recognition.

Table 4.6 contains the results obtained with *k*-NN and SVM-FML without dimensionality reduction (with all terms). The best results for each classification algorithm are in bold. The results presented in Table 4.6 show that *k*-NN outperforms SVM-FML on the considered corpus. The best term weighting methods with *k*-NN are TM2 and NTW.

Table 4.6

Numerical results on the VAM corpus for emotion recognition (*F1*-score)

Term weighting method	*k*-NN		SVM-FML (*F1*-score)
	F1-score	The best *k*	
IDF	0.322	3	0.308
GR	0.292	13	0.307
CW	0.311	13	**0.320**
TM2	**0.345**	3	0.308
RF	0.299	4	0.308
TRR	0.329	4	0.308
NTW	**0.348**	4	0.308

Figures 4.21 and 4.22 contain the results of weight-based term filtering with *k*-NN and SVM-FML respectively. Figures 4.23 and 4.24 demonstrate the results of feature transformation based on term clustering with *k*-NN and SVM-FML respectively. These figures illustrate that both dimensionality reduction techniques mostly demonstrate a stochastic behavior without any significant regularities on the considered corpora due to the low classification results in general. Furthermore, the best results with weight-based term filtering and weight-based term clustering are lower than the best results obtained with *k*-NN without dimensionality reduction (see Table 4.6).

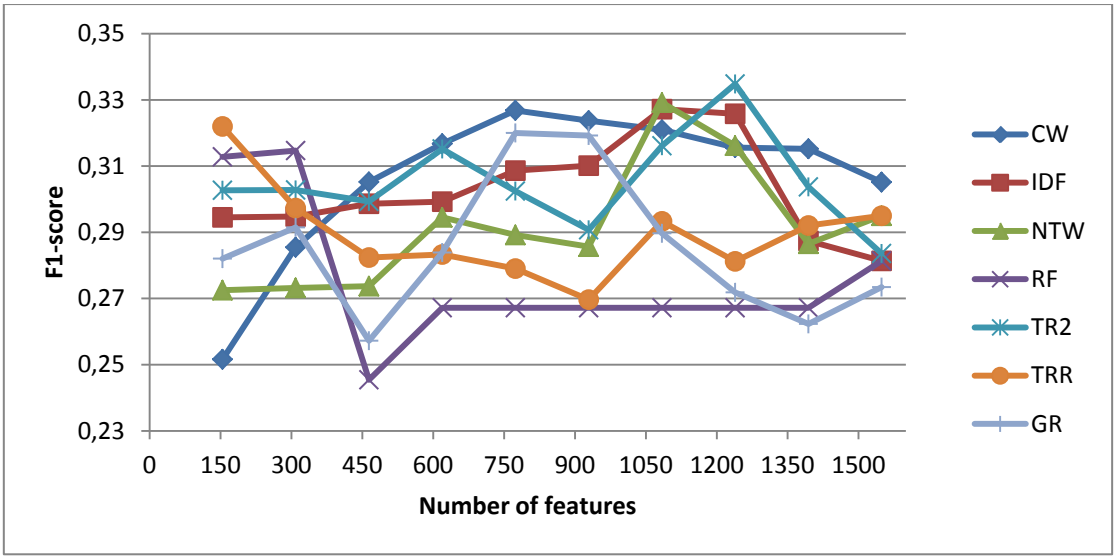

Figure 4.21 – Weight-based term filtering with *k*-NN on the VAM corpus

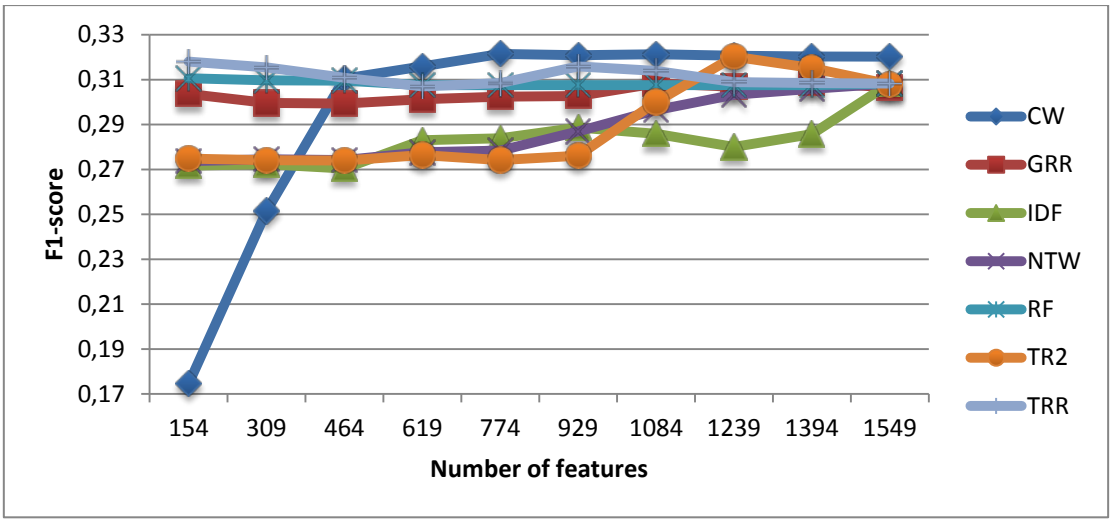

Figure 4.22 – Weight-based term filtering with SVM-FML on the VAM corpus

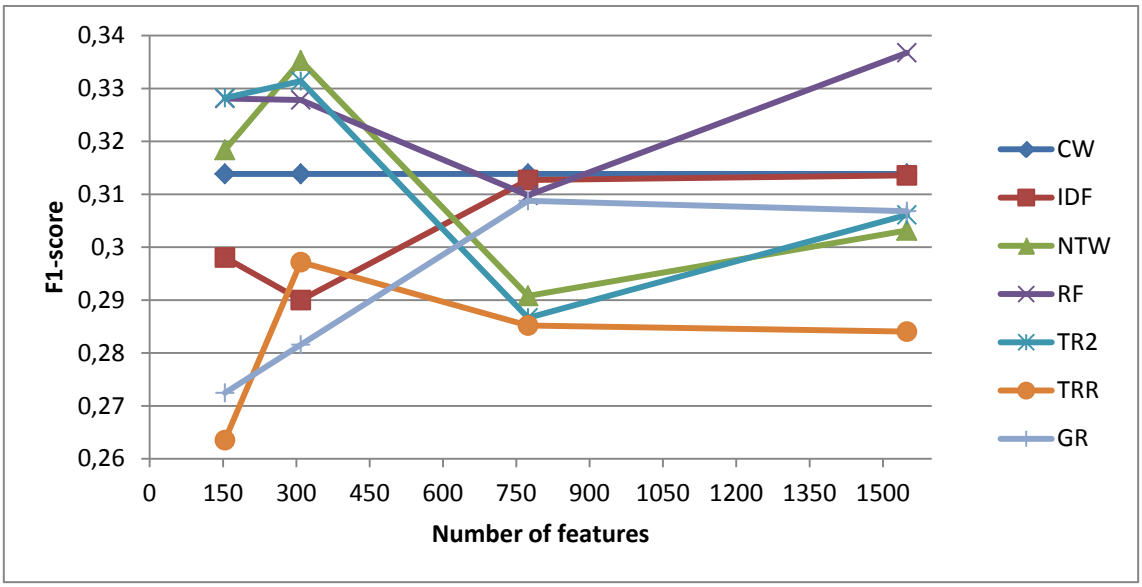

Figure 4.23 – Feature transformation based on term clustering with *k*-NN

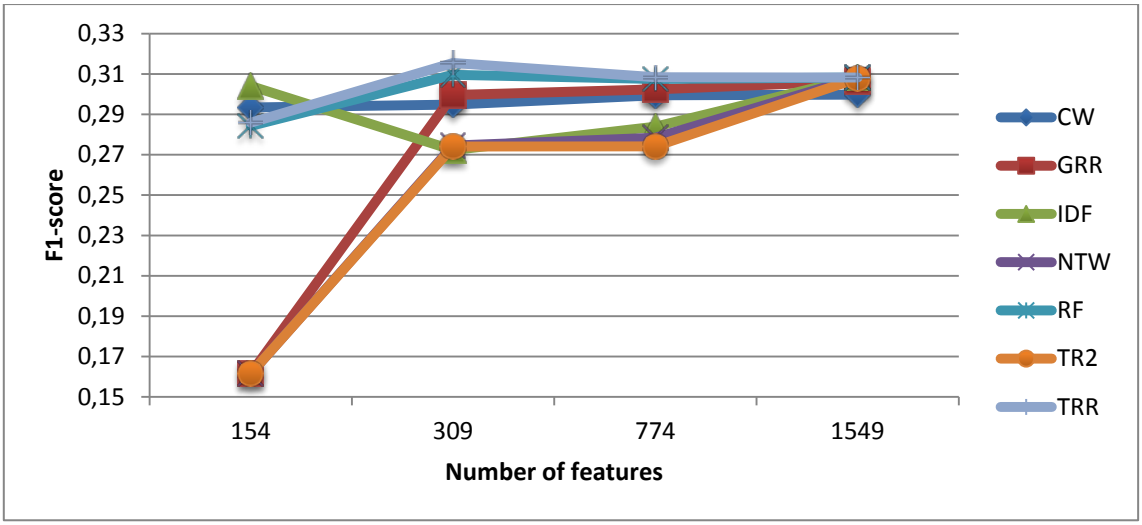

Figure 4.24 – Feature transformation based on term clustering with SVM-FML

Generally, the numerical results obtained on the VAM corpus showed that all methods provide low classification results in terms of *F1*-score for the text-based emotion recognition problem with four classes. This may be explained as follows:

- The VAM corpus does not contain sufficient data for effective machine learning. There are only 947 labeled utterances.

- The corpus is strongly unbalanced (see Section 3.3.2). Two classes out of four contain only 7% of total number of utterances. In this case, *F1*-score values should be significantly lower than accuracy values. E.g. the average accuracy without dimensionality reduction with the average term weighting method and with *k*-NN equals 0.544.

In general, linguistic information has shown to be less informative for emotion recognition. There are some existing results obtained on the VAM corpus using acoustic and video information (Sidorov et al., 2014a). The accuracy values obtained using acoustic and video information are up to 0.7. The accuracy values obtained using only linguistic information are up to 0.56.

Nevertheless, the obtained results are significantly higher than results obtained if a random choice is applied for the four-class classification problem. Therefore, the results of text-based emotion recognition may be useful for multi-modal emotion recognition (fusion with acoustic and video information).

4.5. Summary

A comparative study of state-of-the-art text classification approaches applied for utterance classification was performed. Two corpora („Speech Cycle“ and „Rafaeli“) for domain detection of user utterances were considered. Three corpora for text-based user state recognition (two for user verbal intelligence recognition and the VAM corpus for user emotion recognition) were also considered. Seven term weighting methods (IDF, CW, GR, TM2, RF, TRR, and NTW), three different dimensionality reduction methods (“stop-words” filtering in combination with stemming, weight-based term filtering, and weight-based term clustering), and three classification algorithms (*k*-NN, SVM-FML, and partly the Rocchio classifier) were applied.

We can make the following conclusions based on the numerical results:

- Text classification approaches are in general performing well for domain detection of user utterances. The best results are the following: *F1*-score 0.873 and accuracy 0.964 on the „Speech Cycle“ corpus with 20 classes and 268,550 utterances (these results are obtained on data configuration 1 that is closest to the real situation); *F1*-score 0.595 and accuracy 0.614 on the „Rafaeli“ corpus with 5 classes and 337 utterances. The numerical results show that the text classification approaches are appropriate for domain detection of user utterances in the case of excessive data (the „Speech Cycle“ corpus) and in the case of scarce data (the “Rafaeli” corpus) as well.

- The best combinations of a term weighting method and a classification algorithm for domain detection of user utterances seem to be the following: term relevance ratio (TRR) + *k*-NN or inverse document frequency (IDF) + SVM.

- Feature filtering (weight-based term filtering or stop-word filtering in combination with stemming) is not appropriate as dimensionality reduction for classification of short user utterances because useful information is lost. This result demonstrates the importance of the performed comparative study applied for utterance classification. Filtering approaches are very effective for text classification with written documents. However, they are not effective for utterance classification.

- Feature transformation based on weight-based term clustering seems to be very effective as a dimensionality reduction technique for utterance classification. It provides significant dimensionality reduction without a decrease in the classification results.

- TRR seems to be the most appropriate term weighting method for dimensionality reduction techniques for topic detection of user utterances.

- It seems to be easier to recognize verbal intelligence in dialogues than in monologues. The best *F1*-score obtained on the monologue corpus equals 0.70, for the dialogue corpus – 0.76.

- The best results for verbal intelligence recognition on the dialogue corpus are obtained with the confident weights (CW) method as term weighting and *k*-NN as a classification algorithm. The CW method for verbal intelligence recognition provides a considerable small number of useful terms that characterize only a class of higher verbal intelligence.

- Text-based emotion recognition does not demonstrate high performance with all approaches (the maximal *F1*-score equal 0.348) but these results may be helpful for multi-modal recognition.

5. Novel Approaches for Utterance Classification

5.1. Introduction

This chapter is related to the second and the third aims of the thesis:

- To improve utterance classification results for SDSs.
- To improve computational effectiveness of utterance classification for SDSs.

Regarding the second aim, collectives of term weighting methods are proposed (see Section 5.3). Their collectives are able to integrate advantages of different term weighting methods simultaneously.

For the third aim, a novel feature transformation method based on terms belonging to classes (see Section 5.2) is proposed. It is able to significantly reduce the dimensionality of the text classification task. Furthermore, a novel approach to neural network design (see Section 5.5) is investigated. It requires less computational resources.

Additionally, novel applications of a self-adjusting genetic algorithm (see Section 2.5.4.2) as an optimization algorithm are presented for both classification results (the second aim of the thesis) and computational effectiveness (the third aim of the thesis) of utterance classification. The self-adjusting procedure of this algorithm solves a difficult problem of the GA parameter setting. The self-adjusting genetic algorithm was applied for the following tasks:

- Weight optimization for voting with collectives of term weighting methods proposed in Section 5.3.
- Feature selection based on the wrappers (see Section 2.5.4.1).
- Neural network structure optimization (see Section 5.4).

It should be noted that all proposed novel approaches may be applied not only for utterance classification but also for text classification in general.

The following novel approaches require a validation: supervised approaches with the collectives of term weighting methods (rule induction and weighted voting), wrapper with the self-adjusting GA, and ANN structure optimization. As a rule, validation requires sufficient amounts of data for effectively dividing a corpus into three sets: the training, the validating, and the test ones. Only the „Speech Cycle“ corpus (see Section 3.2.1) provides enough data for effective validation. The “Rafaeli” corpus (Section 3.2.2)

and corpora for verbal intelligence recognition (Section 3.3.1) have only a small number of utterances (337, 100, and 91 utterances correspondingly). The VAM corpus for emotion recognition (Section 3.3.2) is highly unbalanced. It has some classes with a very small number of utterances. This implies that is difficult to perform effective validation with all these corpora. Therefore, only the simplest validation procedure - tuning of k for the k-NN algorithm – was performed with all corpora (see the results presented in Chapter 4). Another validation procedures (the supervised approaches with the collectives of term weighting methods, the GA-based wrapper, and the ANN structure optimization) were applied only on the „Speech Cycle“ corpus. The novel feature transformation which does not require additional validation was applied on all the considered corpora.

If the additional information is not described in the following, the scheme of the numerical experiments and settings of the algorithms applied in this chapter are the same as those described in Chapter 4.

5.2. Novel Feature Transformation Based on Terms Belonging to Classes

5.2.1. Description

As mentioned in Section 2.5, the dimensionality reduction is a very important stage of text classification. It can be inappropriately time-consuming or inefficient to apply machine learning algorithms for text classification yielding a high dimensionality. Dimensionality reduction is also significant for real-time classification systems such as classifiers that are incorporated into spoken dialogue systems.

In this section we propose a novel feature transformation method for text classification based on terms belonging to classes that significantly reduces the dimensionality. The number of features will be equal to the number of classes. The idea of the method is based on a possibility to assign each term from the dictionary of the considered text classification problem to the most appropriate class (category). Some supervised term weighting methods (GR, CW, RF, TRR, and NTW) perform such an assignment automatically. Descriptions of these assignments are illustrated in Table 5.1, where c_i is the most appropriate class of the term t_i, C is the set of all classes.

For the remaining term weighting methods that are considered in our study (IDF and TM2), it is possible to perform the assignment as for the NTW method. This implies

the choice of the class with the maximal relative frequency of the term among all classes (equation 2.26).

Table 5.1

Assignment of terms to the most appropriate classes in supervised term weighting methods

Method	Assignment
GR	$c_i = \arg\max_{c_j \in C} GR(t_i, c_j)$, where GR is calculated by equation 2.4.
CW	$c_i = \arg\max_{c_j \in C} str(t_i, c_j)$, where str is calculated by equation 2.8.
RF	$c_i = \arg\max_{c_j \in C} rf(t_i, c_j)$, where rf is calculated by equation 2.12.
TRR	$c_i = \arg\max_{c_j \in C} TRR(t_i, c_j)$, where TRR is calculated by equation 2.14.
NTW	$c_i = \arg\max_{c \in C} \frac{n_{ic}}{N_c}$, where n_{ic} is the number of documents of the c^{th} class which contain the i^{th} term, N_c is the number of all documents of the c^{th} class.

After assigning each term from the dictionary to the most appropriate class, the stages of the novel feature transformation methods based on terms belonging to classes are described in Algorithm 5.1.

Algorithm 5.1. The novel feature transformation method based on terms belonging to classes

1. Give the document D for classification.
2. Put $S_i = 0$, $i = 1 \ldots C$, where C is the number of classes (categories) of the considered text classification problem.
3. For each term t in the document D do:
3.1. $S_i = S_i + w_t$, where i is the class of the t^{th} term in correspondence with the assignment, w_t is the weight of the t^{th} term.
4. Put S_i, $i = 1 \ldots C$ as transformed features of the text classification problem.

Therefore, the novel feature transformation method significantly reduces the dimensionality without time-consuming calculations such as required by the PCA algorithm. The novel feature transformation method may be a very appropriate tool for real-time utterance classification, e.g. in spoken dialogue systems.

At a first glance, the novel method seems to be similar to feature transformation based on weight-based term clustering (see Section 2.5.7, Algorithm 2.4) when we set

only one cluster for each class. However, term clustering assigns one average weight for all terms in the cluster. In the novel method, all individual term weights are used. This implies that individual characteristics of each term are kept in the novel procedure.

5.2.2. Evaluation

- *Results obtained on the "Speech Cycle" corpus.*

The novel feature transformation method (the novel FT) significantly reduces the dimensionality on the "Speech Cycle" corpus: the dimensionality equals the number of classes (20 in our case, 0.61% of the original dimensionality).

A comparison of the best classification results obtained with the novel FT and obtained without dimensionality reduction (with all terms, these results are the best ones in terms of *F1*-score based on the experiments described in Chapter 4) is presented in Figures 5.1 and 5.2 for *k*-NN and SVM-FML respectively.

The results illustrated in Figures 5.1 and 5.2 show that a decrease of the classification results is observed with the novel FT in all cases. However, for data configuration 1 with *k*-NN the novel FT method demonstrates a slight decrease of the *F1*-score. For instance, this decrease is much less than the decrease obtained with "stop-word" filtering in combination with stemming (see Figure 4.7). Data configuration 1 is closest to a real situation (see Section 3.2.1). Therefore, the novel FT may be applied for real-time utterance classification systems where computational effectiveness is important.

The numerical results in Figures 5.1 and 5.2 demonstrate that the novel FT is more efficient with *k*-NN than with SVM-FML. The most advantage of the novel FT is observed for data configuration 1. For data configuration 3 the method demonstrates the worst results. We may explain such a behavior by the fact that the novel FT is sensitive to the problem of overfitting.

The best results with the novel FT are obtained with TRR as a term weighting method in all cases; the worst results are obtained with IDF in all cases (see Appendix A.5).

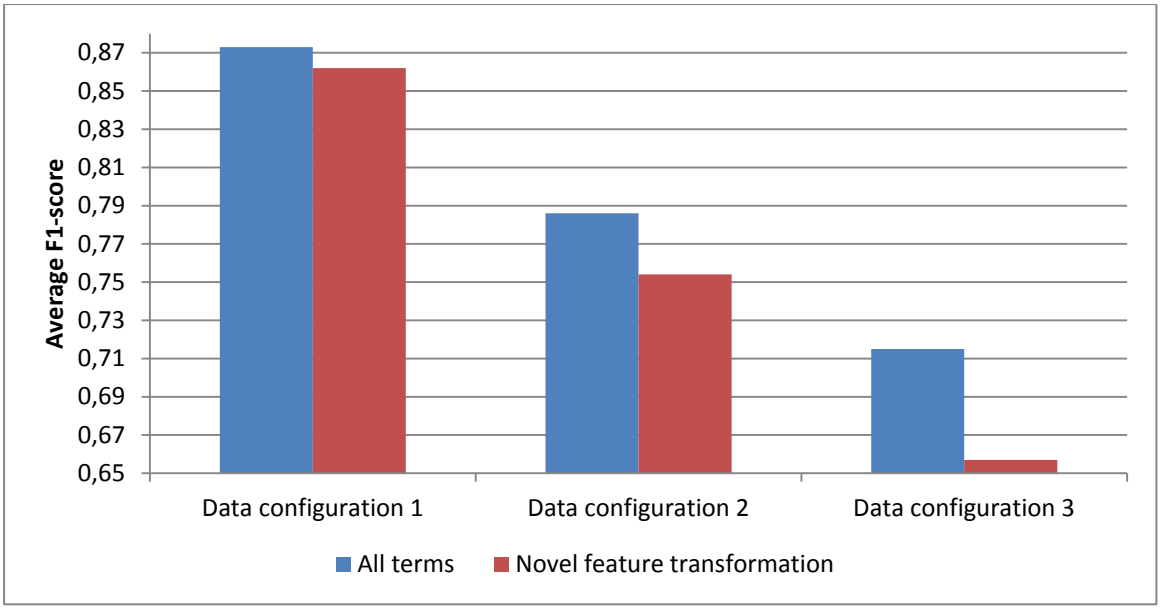

Figure 5.1 – The results of the novel FT with *k*-NN on the "Speech Cycle" corpus

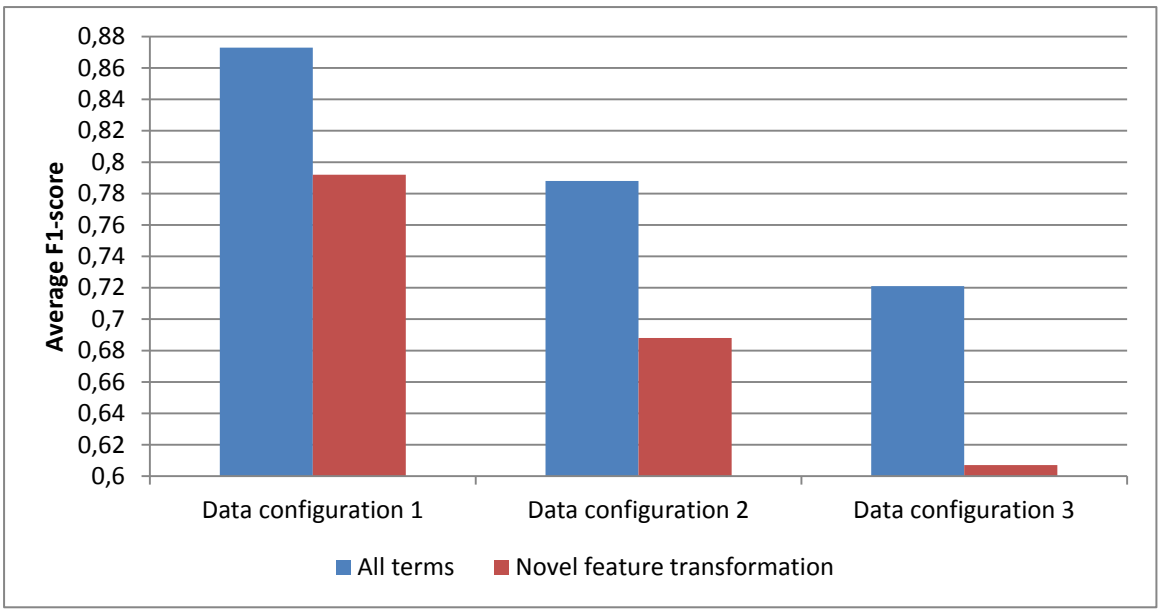

Figure 5.2 – The results of the novel FT with SVM-FML on the "Speech Cycle" corpus

- *Results obtained on the "Rafaeli" corpus.*

Figure 5.3 illustrates the results for the best individual term weighting methods ("The best TWM") with the novel FT in comparison with the best results obtained without dimensionality reduction.

From Figure 5.3 we observe that the novel FT improves the classification results with *k*-NN. However, we do not observe the same situation with SVM-FML. This is consistent with the comparison of the novel FT with *k*-NN and SVM-FML on the „Speech Cycle" corpus. The best term weighting method with the novel NT is RF with

both classification algorithms (see Appendix A.6). The novel FT yields only five features (0.36% of the original dimensionality) for the considered utterance classification problem.

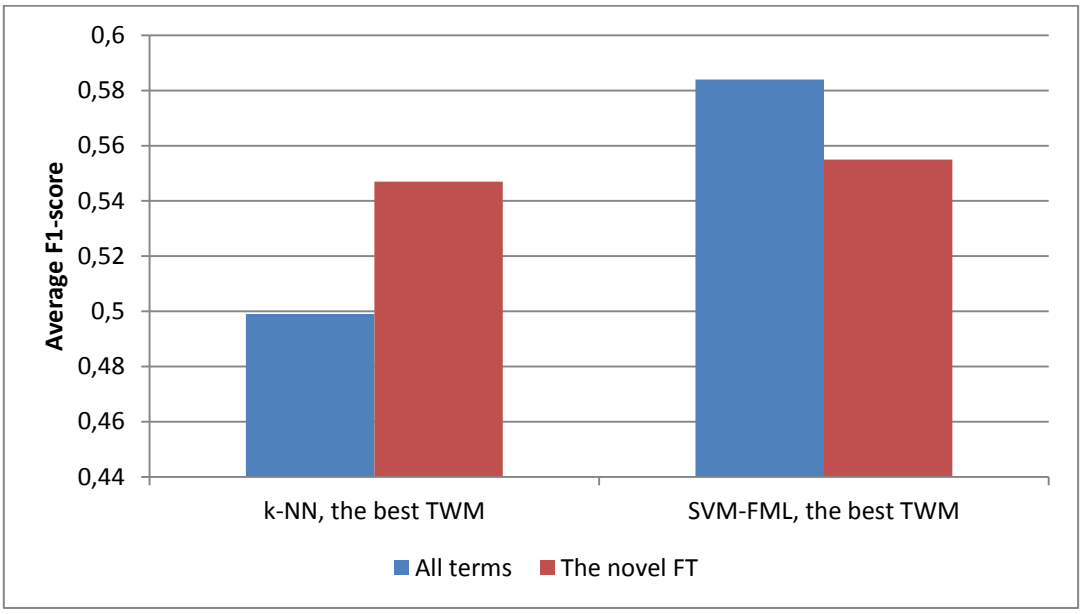

Figure 5.3 – The results of the novel FT on the "Rafaeli" corpus

- *Results obtained on the corpora for verbal intelligence recognition.*

Table 5.2 illustrates the results obtained on the corpora for verbal intelligence recognition with the novel FT in comparison with the results obtained without dimensionality reduction (all terms). The results are presented only with *k*-NN because the results with SVM-FML are very poor on the corpora for verbal intelligence recognition (see Section 4.5).

Table 5.2

The results of the novel FT on the corpora for verbal intelligence recognition with *k*-NN (*F1*-score)

Term weighting method	The monologue corpus		The dialogue corpus	
	All terms	The novel FT	All terms	The novel FT
IDF	0.54	0.57	0.53	0.53
GR	0.35	0.52	0.38	0.49
CW	0.58	**0.64**	**0.76**	0.71
TM2	0.38	0.59	0.37	0.66
RF	0.56	0.56	0.53	0.63
TRR	0.49	0.62	0.50	0.53
NTW	0.43	0.56	0.58	0.52

The results presented in Table 5.2 show that the novel FT outperforms the best result obtained for the monologue corpus. However, this is not observed for the dialogue corpus. The best results on both corpora are obtained with the CW method (see Table 5.2). The novel FT yields only two features for these classification problems (0.06% and 0.03% of the original dimensionalities for the monologue and the dialogue corpora respectively). The CW method yields non-zero weights of the terms that characterize only the class of higher verbal intelligence (see Section 4.4.1). This implies that the novel FT yields only one informative feature with the CW method for the considered corpora of verbal intelligence recognition. Theoretically, a more significant dimensionality reduction is not possible.

- *Results obtained on the VAM corpus for emotion recognition.*

Figure 5.4 illustrates the results for the best individual term weighting methods ("The best TWM") with the novel FT in comparison with the best results obtained without dimensionality reduction.

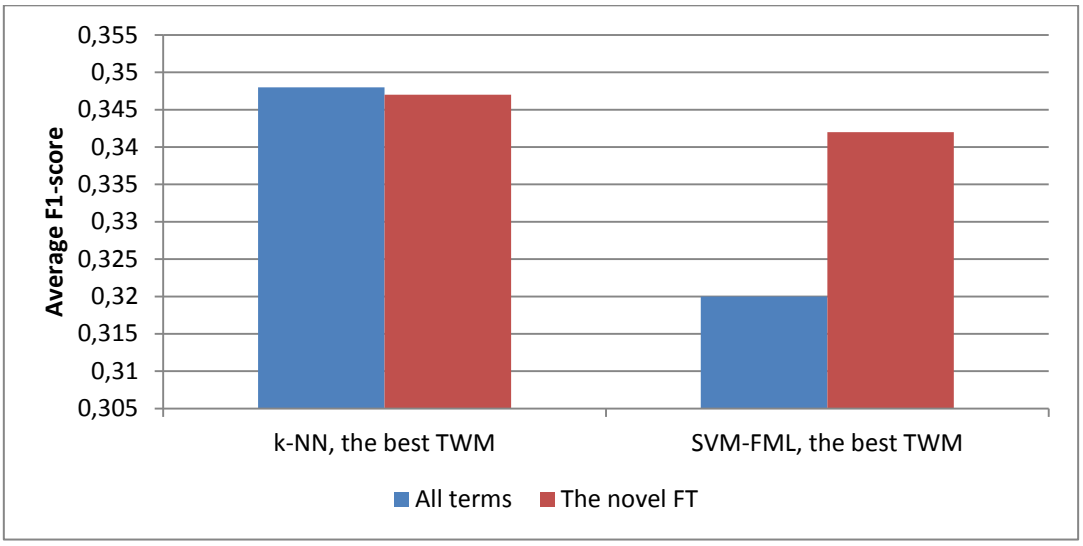

Figure 5.4 – The results of the novel FT on the VAM corpus

The results in Figure 5.4 show that the novel FT does not lead to significant changes in *F1*-scores with *k*-NN and demonstrates a significant improvement of the classification results with SVM-FML on the VAM corpus. At the same time, it provides very significant dimensionality reduction: it yields four features or 0.26% of the original dimensionality. The best term weighting method with the novel FT and with both classification algorithms on the considered corpus is RF.

We conclude that the novel FT may provide a slight improvement or a slight decrease of the classification results. The novel FT seems to be more efficient with *k*-NN than with SVM-FML. The novel FT results in a significant dimensionality reduction: the method yields the number of features that is less than 1% of the original dimensionality on the all considered corpora. Therefore, it may be applied for real-time utterance classification systems where computational efficiency is important.

5.3. Collectives of Term Weighting Methods

5.3.1. Description

Some classification approaches are based on collectives of machine learning algorithms, such as majority vote, bagging (Breiman, 1996), and boosting (Schapire and Singer, 2000). These collectives show better results than those individually obtained for the best algorithm in the collective. This fact was also shown for text classification (Morariu et al., 2005). Therefore, meta-classification is very popular in the field of machine learning. In our research we follow the idea that collectives of different term weighting methods may also improve the text classification accuracy even if the same classification algorithm is used. The procedure of forming *collectives of term weighting methods* is the follows (Algorithm 5.2):

Algorithm 5.2. Forming collectives of term weighting methods

1. Choose a classification algorithm with the predefined of fixed after validation settings.
2. Apply the classification algorithm chosen in step 1 for the considered text classification problem with seven different term weighting methods (IDF, GR, CW, TM2, RF, TRR, and NTW).
3. For each sample from the test set, extract the predicted class with each term weighting method.
4. Put class labels found in step 2 as categorical features of the collective.
5. Add a true label for each sample from the test set.

After establishing collectives of term weighting methods, a meta-classification should be performed. Two approaches for meta-classification are considered: a meta-classifier based on rule induction (Cohen, 1995) and a majority voting procedure.

The first approach is a meta-classifier based on rule induction with Repeated Incremental Pruning to Produce Error Reduction (RIPPER, Cohen, 1995). This algorithm is able to process categorical attributes. In the following, a short description of the RIPPER algorithm is presented. Starting with the less prevalent classes, the algorithm iteratively grows and prunes rules until there are no positive examples left or the error rate exceeds 50%. In the growing phase, for each rule greedily conditions are added to the rule until 100% accuracy has been achieved. The procedure tries every possible value of each attribute and selects the condition with the highest information gain. In the pruning phase, for each rule any final sequences of the antecedents is pruned using a pruning metric. The RIPPER algorithm as a rule set learner is often compared to decision tree learners. Rule sets have the advantage that they are easy to understand, representable in first order logic (easy to implement in languages like *Prolog*) and that prior knowledge can be added to them easily. The major disadvantages of rule sets are that they scale poorly with the training set size and weakly perform under noisy conditions. The RIPPER algorithm pretty much overcomes these disadvantages. The major problem with decision trees is overfitting, i.e. the model works very well on the training set but does not perform well on the validation set. Reduced Error Pruning (REP) in the RIPPER algorithm is a technique that tries to overcome overfitting.

After classification with a machine learning algorithm with all term weighting methods the classification results (the predicted class by each term weighting method) are used as categorical features for RIPPER. We propose two schemas for rule induction. As a first option, we perform the RIPPER algorithm on the entire training set. However, in this case the sets for machine learning at the first stage and for rule induction as a meta-classifier are the same. Overfitting may therefore occur. Alternatively we divide the training set into two parts. The first division (of larger size, as a rule) is used for the first level of machine learning and the second one (validating set) is used for rule induction. In this case we should apply algorithm 3.1 for the test set and for the validating set independently. *RapidMiner* was used as software for rule induction based on the RIPPER algorithm application (Shafait et al., 2010).

The second approach for meta-classification is majority voting. This means that we choose the class that was predicted by the majority of term weighting methods as a result of meta-classification. If we have some classes with an equal number of votes, we choose one of them randomly. The majority voting procedure does not require additional learning. Therefore, it is not necessary to handle validating sets.

It may be possible to improve the performance of the collectives of term weighting methods proposed with weighted voting. In this case the meta-classification procedure is similar to majority voting. However, weights are added for each term weighting method. The voting result is calculated according to these weights. We propose two different definitions of the weighted voting with the collectives of term weighting methods:

- With 7 weights: each weight corresponds to one term weighting method (IDF, GR, CW, TM2, RF, TRR, and NTW). The weights vary within the interval [0;1]. This weighted voting is described in Algorithm 5.3.

Algorithm 5.3. Weighted voting with collectives of term weighting methods 1

1. Put the results for each class as zero: $R_i = 0$, $i = 1 \ldots N$, N is the number of classes
2. For the classification result obtained with each term weighting method j (IDF, GR, CW, TM2, RF, TRR, and NTW) do ($j = 1 \ldots 7$):
   2.1. Recalculate $R_p = R_p + w_j$, where p is the class predicted with the current j^{th} term weighting method, w_j is the weight of the current j^{th} term weighting method.
3. Find the final results of the weighted voting as follows: $result = \arg\max_{i \in 1..N} R_i$

- With $7*N$ variables, where N is the number of classes of the considered text classification problem: each weight corresponds to one term weighting method (7 methods) with the specified class (the predicted by the method class, N different classes). The weights vary within the interval [0;1]. This problem definition may be more flexible and more effective than the previous one because it may take specializations of different term weighting methods for different classes into account. This weighted voting is presented in Algorithm 5.4.

Algorithm 5.4. Weighted voting with collectives of term weighting methods 2

1. Put the results for each class as zero: $R_i = 0$, $i = 1 \ldots N$, N is the number of classes
2. For the classification result obtained with each term weighting method j (IDF, GR, CW, TM2, RF, TRR, and NTW) do ($j = 1 \ldots 7$):
   2.1. Recalculate $R_p = R_p + w_{jp}$, where p is the class predicted with the current j^{th} term weighting method, w_{jp} is the weight of the j^{th} term weighting method for the p^{th} class.
3. Find the final results of the weighted voting as follows: $result = \arg\max_{i \in 1..N} R_i$

The weights for the voting should be optimized on the validating set. This optimization task is complicated due to the large amount of training data and procedural definition of fitness function. In order to solve this optimization problem it seems to be appropriate to apply the self-adjusting genetic algorithm described in Section 2.5.4.2. It is appropriate for such complicated optimization problems and able to solve the problem of GA parameter settings. As a fitness function for the described optimization problem we used *F1*-score on the validating set of the classification task.

5.3.2. Evaluation

We have designed the collectives by including seven term weighting methods: IDF, GR, CW, TM2, RF, TRR, and NTW. A collective of term weighting methods contains predictions that were obtained by each term weighting method under the same conditions (the same problem definition, the same classification algorithm, the same dimensionality reduction technique). Supervised approaches with the collectives of term weighting methods (rule induction and weighted vote) were applied only on the „Speech Cycle“ corpus (see the explanation provided in Section 5.1). The majority vote procedure was additionally applied on the “Rafaeli” corpus.

- *Results obtained on the “Speech Cycle” corpus.*

Two approaches for handling the collectives are considered for the “Speech Cycle” corpus: the meta-classifier based on the rule induction with the RIPPER algorithm (Cohen, 1995) and the majority vote procedure. For comparison of these two approaches, we tested them at first only for data configuration 2 with *k*-NN and without dimensionality reduction.

The first approach is a meta-classifier based on the rule induction. After classification with *k*-NN for all term weighting methods, we used the classification results (the predicted class by each term weighting method) as categorical features for the rule induction. We propose two schemas for rule induction learning. With the first way we perform rule induction learning for the whole training sample but in this case samples for *k*-NN learning and rule induction learning are the same. With the second way we divided the training sample into two parts in the proportion 4:1 for validation. The first division is used for *k*-NN learning and the second one (validating sample) is used for rule induction learning. *RapidMiner* 5.0 was used as software for rule induction application (Shafait et al., 2010) with standard settings that are described in Table 5.3.

Table 5.3

The standard settings of the Rule Induction algorithm in *RapidMiner* 5.0

Parameter	Value
classification strategies	1 against all
use local random seed	False
parallelize learning process	False

Table 5.4 contains the results for different approaches for the collective handling (average *F1*-score) and the comparison with the best term weighting methods (TRR, TM2, and NTW for data configuration 2 with *k*-NN and without dimensionality reduction, see the results in Section 4.3.1.1) based on *t*-test.

Table 5.4

Comparison of approaches for meta-classification with collectives of term weighting methods

Approach	*F1*-score	*t*-test		
		TRR	TM2	NTW
Rule induction without validation	0.789	0.169	0.370	0.124
Rule induction with validation	0.788	0.811	0.910	0.760
Majority vote	**0.805**	0.000	0.000	0.000

The results of the numerical experiments allow the following conclusions. Rule induction does not provide a statistically significant improvement of the *F1*-score with the collectives of term weighting methods. The best results are obtained with the majority vote which significantly improves the *F1*-score. The average increase of the *F1*-score with the majority vote equals 0.019. Therefore, we decided to apply majority vote as an approach for the handling of the collectives for all other situations and corpora.

The results of the collectives based on majority vote are presented in Tables 5.5 – 5.10. The majority vote was applied with *k*-NN and SVM-FML; with all terms (without dimensionality reduction), with “stop-word” filtering + stemming, and with the novel FT. The best results of individual term weighting methods are illustrated for comparison (strings “The best TWM”). Detailed information about the best term weighting methods on the considered corpus is provided in Section 4.3. The rows „Difference“ contain differences of *F1*-score and accuracy („Accur.“) between the best individual term weighting and the results with the collectives.

Table 5.5

Numerical results with *k*-NN and without dimensionality reduction

Combination	Data configuration 1		Data configuration 2		Data configuration 3	
	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**
The best TWM	0.873	0.964	0.786	0.839	0.715	0.785
Collective	**0.883**	**0.967**	**0.805**	**0.853**	**0.730**	**0.796**
Difference	*0.010*	*0.003*	*0.019*	*0.014*	*0.015*	*0.011*

Table 5.6

Numerical results with SVM-FLM and without dimensionality reduction

Combination	Data configuration 1		Data configuration 2		Data configuration 3	
	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**
The best TWM	**0.873**	**0.963**	**0.788**	**0.833**	**0.721**	**0.788**
Collective	0.846	0.956	0.746	0.805	0.686	0.769
Difference	*-0.027*	*-0.007*	*-0.005*	*-0.002*	*0.001*	*0.002*

Table 5.7

Numerical results with *k*-NN and "stop-word" filtering + stemming

Combination	Data configuration 1		Data configuration 2		Data configuration 3	
	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**
The best TWM	0.793	**0.917**	0.743	0.796	0.704	0.774
Collective	**0.799**	0.903	**0.752**	**0.802**	**0.715**	**0.784**
Difference	*0.006*	*-0.014*	*0.009*	*0.006*	*0.011*	*0.010*

Table 5.8

Numerical results with SVM-FLM and "stop-word" filtering + stemming

Combination	Problem definition 1		Problem definition 2		Problem definition 3	
	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**
The best TWM	**0.836**	**0.954**	**0.756**	**0.817**	**0.704**	**0.781**
Collective	0.806	0.945	0.726	0.800	0.681	0.768
Difference	*-0.030*	*-0.009*	*-0.030*	*-0.017*	*-0.023*	*-0.013*

Table 5.9

Numerical results with *k*-NN and novel feature transformation

Combination	Data configuration 1		Data configuration 2		Data configuration 3	
	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**
The best TWM	0.862	0.960	0.754	0.808	0.657	0.735
Collective	**0.876**	**0.965**	**0.787**	**0.854**	**0.689**	**0.763**
Difference	*0.014*	*0.005*	*0.033*	*0.046*	*0.032*	*0.028*

Table 5.10

Numerical results with SVM-FLM and novel feature transformation

Combination	Data configuration 1		Data configuration 2		Data configuration 3	
	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**	***F1*-score**	**Accur.**
The best TWM	**0.792**	**0.937**	**0.668**	**0.734**	**0.607**	**0.696**
Collective	0.718	0.926	0.629	0.728	0.567	0.693
Difference	*-0.074*	*-0.011*	*-0.039*	*-0.006*	*-0.040*	*-0.003*

Observing the results in Tables 5.5 – 5.10 we can make the following conclusions:

- Collectives of term weighting methods have shown to be efficient with *k*-NN rather than with SVM-FML.

- Collectives of term weighting methods with *k*-NN result in a significant improvement of the *F1*-score in all cases for three data configurations.

- The most significant effect of the collectives is observed with the novel feature transformation method.

The next stage of investigation is the weighting vote with all seven term weighting methods based on weight optimization. As an optimization algorithm we used a self-tuning genetic algorithm that was proposed by Semenkin and Semenkina (2012, see Algorithm 2.3 in Section 2.5.4.2). Weight optimization was applied only for results with *k*-NN because the results with SVM-FML are not promising. We proposed two definitions of the considered optimization problem:

- 7 variables: each variable means a weight for a term weighting method. Variables are varied in the interval [0;1] (see Algorithm 5.3).

- 7*20 (140) variables: each variable means a weight for a term weighting method (7 methods) with specified class (a predicted by a method class, 20 classes). Variables are varied in the interval [0;1] (see Algorithm 5.4).

As fitness function we used the *F1*-score on the validating set (1/4 of the training set). The results of the optimization are presented in Tables 5.11 – 5.13 for three data configurations respectively. The optimization algorithm was applied 20 times for each problem definition in three different situations: without dimensionality reduction (all terms), “stop-word” filtering + stemming, and with novel feature transformation. The number of generations equals to 250. The row “7, best” means the results with 7 variables and with choice the best *F1*-score on the validating set by 20 algorithm runs. The string “7, average” means the results with 7 variables and with the average *F1*-score by 20 algorithm runs. The same explanation applies for the “7*20” rows. Those values that were significantly improved with optimization are in bold.

The results in Tables 5.11 – 5.13 showed that the optimization results in a significant improvement of the *F1*-score in some cases. The most significant improvement is observed for the problem definition 3 with the novel feature transformation (+0.009). Therefore, we can conclude that collectives of all seven term weighting methods based on weighted voting provide the best classification effectiveness in all cases with *k*-NN.

Table 5.11

Optimization of weights for voting procedure (*F1*-score, data configuration 1, *k*-NN)

	Without dimensionality reduction	**"Stop-word" filtering + stemming**	**Novel feature transformation**
Without optimization	*0.883*	*0.799*	*0.876*
7, best	**0.885**	0.771	0.878
7, average	**0.885**	0.770	0.878
7*20, average	**0.887**	0.776	**0.878**
7*20, best	**0.886**	0.774	**0.878**

Table 5.12

Optimization of weights for voting procedure (*F1*-score, data configuration 2, *k*-NN)

	Without dimensionality reduction	**"Stop-word" filtering + stemming**	**Novel feature transformation**
Without optimization	*0.805*	*0.752*	*0.787*
7, best	0.805	0.753	0.785
7, average	0.805	0.752	0.785
7*20, average	0.805	**0.756**	0.787
7*20, best	0.806	**0.755**	0.786

Table 5.13

Optimization of weights for voting procedure (*F1*-score, data configuration 3, *k*-NN)

	Without dimensionality reduction	**"Stop-word" filtering + stemming**	**Novel feature transformation**
Without optimization	*0.730*	*0.715*	*0.689*
7, best	0.731	0.715	0.690
7, average	0.731	0.715	0.691
7*20, average	0.732	0.716	**0.698**
7*20, best	0.731	0.715	**0.697**

In total, the collectives of term weighting methods with weight optimization provide the following significant improvements of *F1*-score in comparison with the best individual term weighting methods:

- For data configuration 1 with all terms: from 0.873 to 0.887 (+0.014).
- For data configuration 1 with the novel FT: from 0.862 to 0.878 (+0.016).
- For data configuration 2 with all terms: from 0.788 to 0.806 (+0.018).
- For data configuration 2 with the novel FT: from 0.754 to 0.787 (+0.033).
- For data configuration 3 with all terms: from 0.721 to 0.731 (+0.010).
- For data configuration 3 with the novel FT: from 0.657 to 0.698 (+0.041).

A significant improvement of the classification effectiveness is observed with the novel feature transformation method. The final results with the novel feature transformation are very close to those obtained without dimensionality reduction. At the same time, the novel feature transformation method significantly reduces the dimensionality and may be useful for real-time classification systems such as natural language call routing.

- *Results obtained on the "Rafaeli" corpus*.

The majority vote was applied as a meta-classification method with collectives of term weighting methods on the "Rafaeli" corpus. The results are presented in Figure 5.5. The supervised approaches with the collectives of different term weighting methods (rule induction and the weighted voting) were not applied on the „Rafaeli" corpus due to the small amount of data for validation (see the explanation provided in Section 5.1).

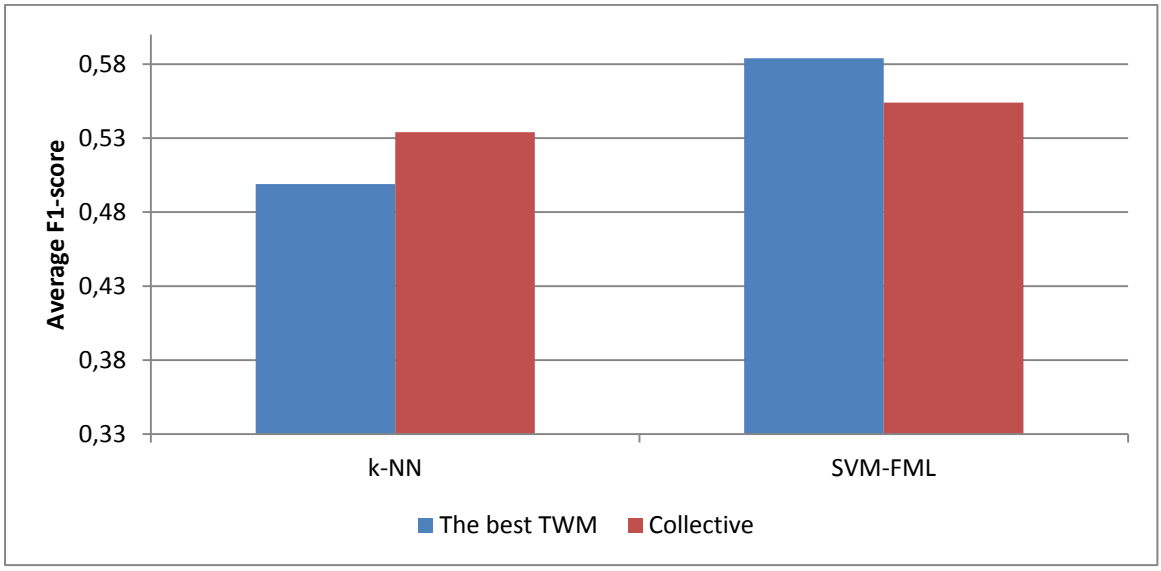

Figure 5.5 – Collectives of term weighting methods on the "Rafaeli" corpus

Figure 5.5 illustrates the results obtained with the collectives of term weighting methods and the best individual term weighting methods. From Figure 5.5 we conclude that collectives of term weighting methods are effective with *k*-NN and not effective with SVM-FML (the same result has been obtained on the "Speech Cycle" corpus).

Collectives of term weighting methods were not applied on the corpora for verbal intelligence and the VAM corpus for emotion recognition due to the following reasons: At first, these corpora do not contain enough amounts of the data at all (the corpora for verbal intelligence recognition) or for the smallest classes (the VAM corpus). In this case,

it is difficult to increase the common classification score. Additionally, the CW method on the corpora of verbal intelligence recognition and the RF method with the novel FT on the VAM corpus for emotion recognition significantly outperform the other term weighting methods. The collectives of term weighting methods may be useful in the case of some competitive term weighting methods that demonstrate different efficiency for different classes. This means that different term weighting methods should demonstrate their advantages with different test samples for classification. This allows to make use of these advantages with the collectives of term weighting methods and to increase the total classification score. Such a situation is observed on the “Speech Cycle” and the “Rafaeli” corpora. Therefore, the collectives of term weighting methods were applied only on these two corpora for domain detection of user utterances.

The general conclusions with the collectives of term weighting methods are the followings:

- The collectives of term weighting methods based on the majority vote or the weighted vote may improve the utterance classification results when sufficient data for learning exists and different term weighting methods yield the specific advantages with different samples for classification.
- The collectives of term weighting methods seem to be efficient with *k*-NN and not efficient with SVM-FML.
- The collectives of term weighting methods seem to be especially efficient with the novel feature transformation proposed in Section 5.2.

5.4. Overall Comparison

The results presented in Section 5.3 on the “Speech Cylce” corpus demonstrate that the collectives of different term weighting methods are especially efficient with the novel feature transformation. Therefore, it is interesting to compare the results obtained with these two novel techniques to the best results obtained with state-of-the-art approaches (see results presented in Section 4.3.1).

In this section the overall comparison of the numerical results obtained on the “Speech Cycle” corpus with the novel approaches (collectives of term weighting method s and the novel feature transformation) is presented. We show the best results in terms of *F1*-score without (see the results in Section 4.3.1) and with our novel techniques: collectives of term weighting methods based on weighted voting and the novel feature

transformation method (Figure 5.6 for the data configuration 1, Figure 5.8 for data configuration 2, Figure 5.10 for data configuration 3). We also present the least dimensionalities that provide the best *F1*-scores without the novel feature transformation in comparison with the dimensionality for the novel FT (Figures 5.7, 5.9, 5.11) in logarithmic scale. These best results without the novel FT are obtained with TRR as a term weighting method, *k*-NN as a classification algorithm, and feature transformation based on term clustering as a dimensionality reduction method for all three data configurations (see Section 4.3).

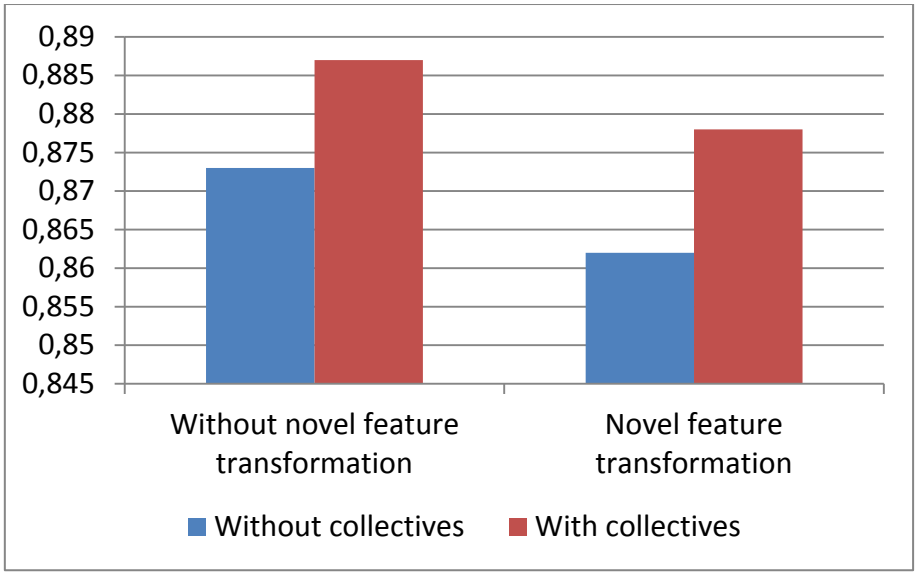

Figure 5.6 – Overall comparison for data configuration 1 in terms of *F1*-score

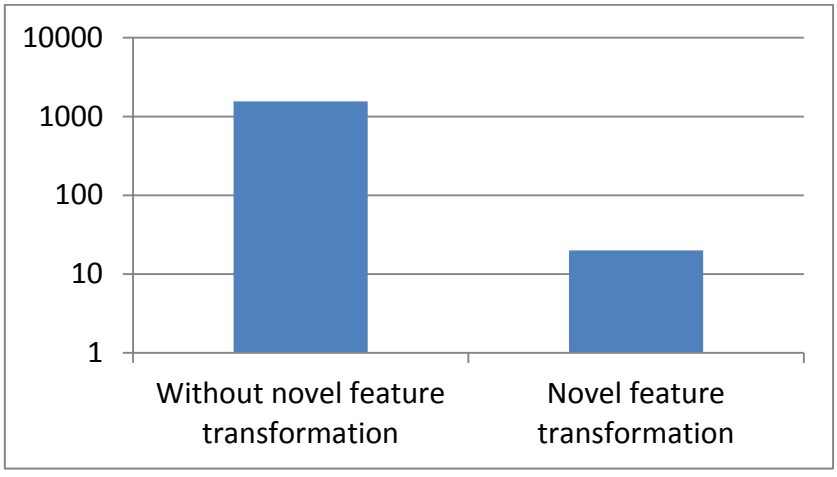

Figure 5.7 – Overall comparison for data configuration 1 in terms of dimensionality

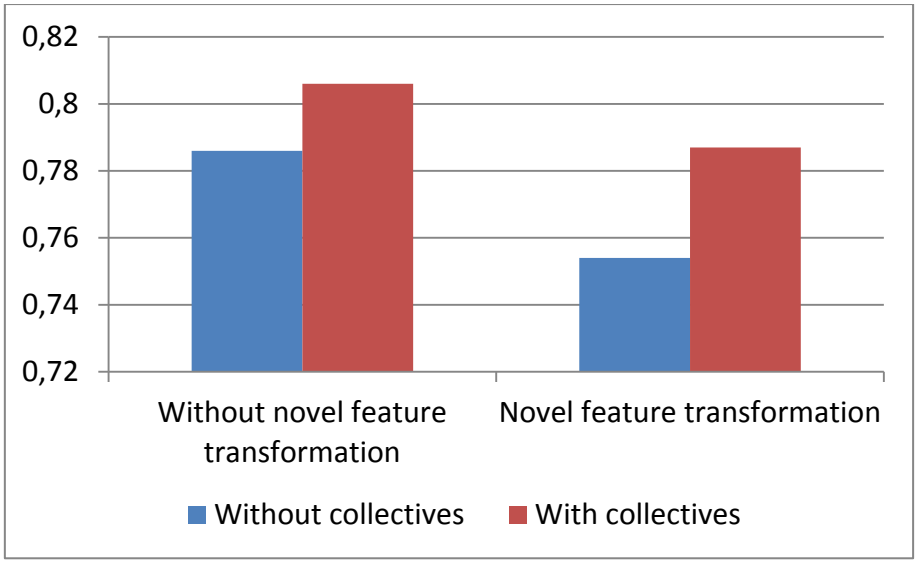

Figure 5.8 – Overall comparison for data configuration 2 in terms of *F1*-score

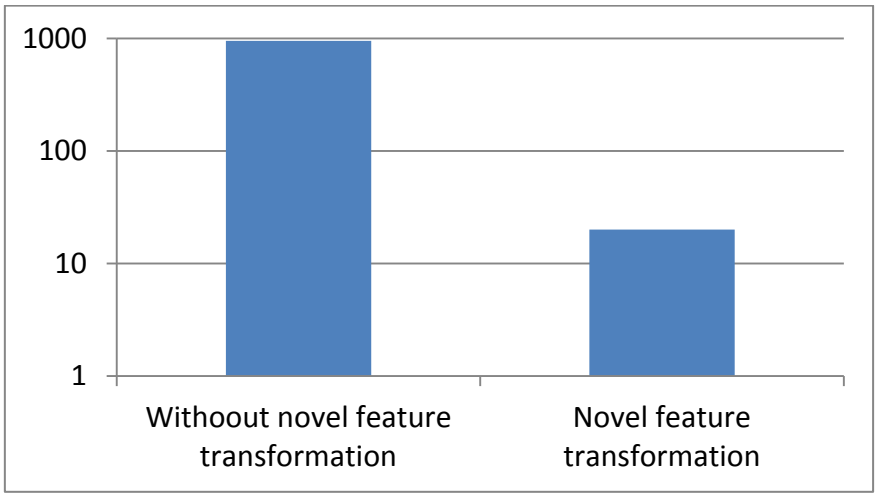

Figure 5.9 – Overall comparison for data configuration 2 in terms of dimensionality

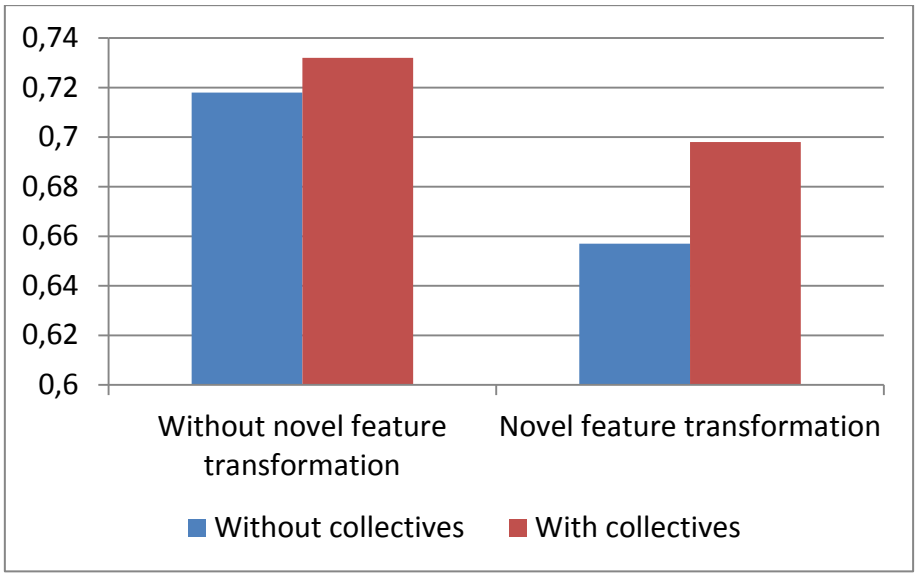

Figure 5.10 – Overall comparison for data configuration 3 in terms of *F1*-score

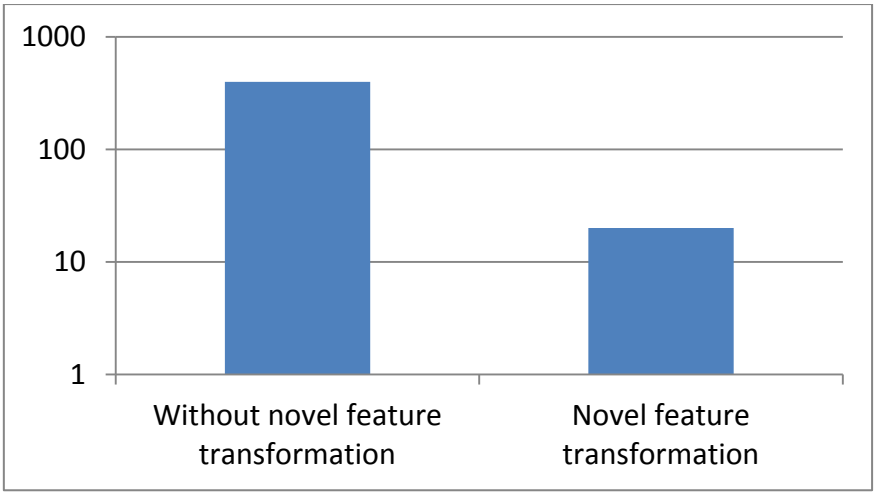

Figure 5.11 – Overall comparison for data configuration 3 in terms of dimensionality

We make the following conclusions based on the results presented in Figures 5.6 – 5.11.

- For data configuration 1 the combination of novel techniques (collectives of term weighting methods and the novel feature transformation method) results in a significant improvement of the classification performance in terms of *F1*-score.
- For data configuration 2 there is no significant difference in terms of *F1*-score between the combination of novel techniques and the results obtained without these techniques.
- For data configuration 3 there is a slight decrement of *F1*-score when using the combination of novel techniques.
- Combination of novel techniques results in a significant dimensionality reduction.

Data configuration 1 is the closest one to the real situation. Therefore, the novel feature transformation method in combination with the collectives of term weighting methods significantly reduces the dimensionality and improves the classification performance. This method may be useful for real-time classification systems such as SDSs.

The Figures 5.12 and 5.13 demonstrate an importance of dimensionality reduction in terms of computational time. The diagrams show the average computation time in minutes for the learning stage with *k*-NN and SVM-FML without dimensionality reduction and with the novel feature transformation methods for one training set and one test set. The computational system is the following: Intel Core i7 2.90 GHz, RAM 8 GB,

OS Windows 7 Professional. Software that implements classification algorithms: *RapidMiner* v.5.0. The average computational time with *k*-NN and without dimensionality reduction equals 165 min; with the novel feature transformation: 3.5 min. The average computational time with SVM-FML and without dimensionality reduction equals 67 min; with the novel feature transformation: 4 min.

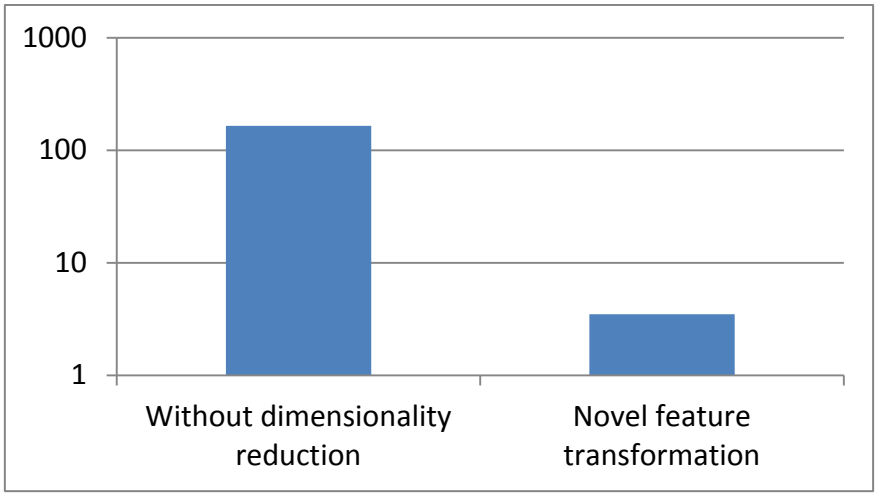

Figure 5.12 – Computational time for the learning stage with *k*-NN, minutes

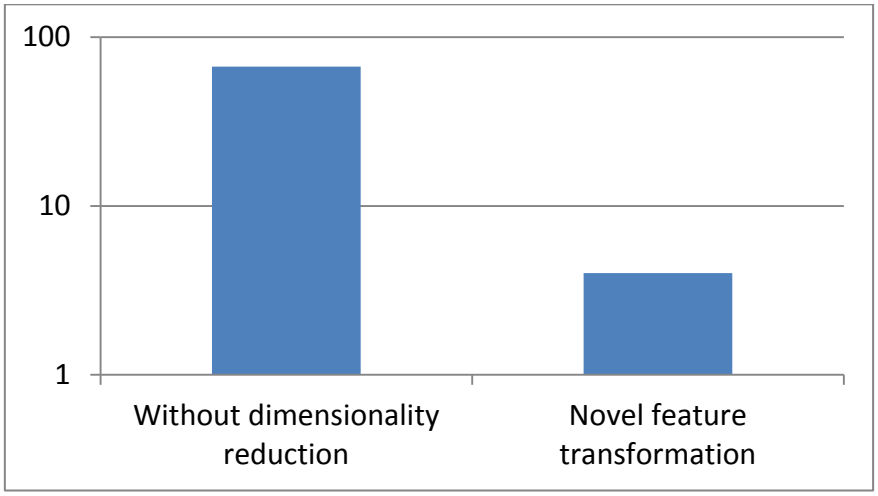

Figure 5.13 – Computational time for the learning stage with SVM-FML, minutes

The general conclusions are the following:

- Collectives of term weighting methods significantly improve the classification performance for all data configurations with *k*-NN.

- The novel feature transformation method provides appropriate classification results with a very compact feature set that is important in terms of computational time.

5.5. Novel Approach to Neural Network Design

ANNs which are described in Section 2.6.6 may be an effective tool for text classification. However, ANNs are very sensitive to the parameter settings. The neuron weights and parameters that characterize the structure of an ANN should be optimized. An ANN includes the following structure parameters:

- Number of hidden layers.
- Number of neurons in each layer.
- Connections between neurons.
- Types of an activation function for each neuron.

The ANN design should include an optimization of neuron weights and a structure optimization. An effective tool for the optimization of neuron weights is the error backpropagation algorithm (see Section 2.6.6, Algorithm 2.5). We use an implementation of the error backpropagation algorithm in *RapidMiner* 5.0 with the standard settings listed in Table 5.14.

Table 5.14

The standard settings of the Error Backpropagation algorithm in *RapidMiner* 5.0

Parameter	Value
training cycles	500
learning rate	0.3
momentum	0.2
decay	False
shuffle	True
normalize	True
error epsilon	1.0E-5
use local random seed	False

RapidMiner 5.0 also provides a fixed default structure of an ANN. The layer size is calculated from the number of attributes and the number of classes of the considered classification problem. In this case, the layer size will be set to (number of attributes + number of classes) / 2 + 1. One default hidden layer with sigmoid activation function (see equation 5.3) and the above predefined size is created and added to the net. Although this fixed ANN structure is rational, it may be not optimal or even sub-optimal for complicated classification tasks. Therefore, structure optimization should be applied.

It is impossible to apply an optimization algorithm such as error backpropogation for the ANN structure optimization because this algorithm requires an analytical and differentiable form of the fitness function. The fitness function for the structure

optimization of ANNs is procedural. Furthermore, some parameters of the ANN structure are discrete (number of layers and number of neurons) or categorical (types of an activation function for each neuron, connections between neurons). Due to these characteristics of the optimization task, more complicated optimization algorithms such as genetic algorithms (GAs) should be applied.

A GA requires a specified representation of the ANN structure as a binary string (see Section 2.5.4.1). We consider the Ward ANN structure representation presented by Frederick (1995) as a baseline one (see Section 5.5.1).

We propose a novel simplified ANN structure representation (see Section 5.5.2) for the GA. The novel approach provides an effective ANN structure optimization within reasonable computational requirements.

Therefore, we apply an optimization approach which is based on a combination of deterministic and stochastic optimization methods: the error backpropagation algorithm for neuron weights optimization and the GA for structure optimization; the GA-based part of the algorithm has a simplified solution representation.

Furthermore, we implement our novel ANN structure representation with the self-adjusting GA (see Section 2.5.4.2) which is able to solve a complicated problem of the GA parameters setting.

All results in this section are obtained on the “Speech Cycle” corpus.

5.5.1. Baseline ANN Structure Optimization with GAs

In the definition of the GA, each individual contains all the information about the ANN structure (the number of neurons in each hidden layer, the kinds of their activation functions). It is critical for the classical GA to represent solutions as binary strings of a fixed length. We consider the Ward ANN, where neurons in each layer are divided into blocks having the same activation function. This approach uses the following solution encoding: there is some superstructure, which can represent any ANN of a smaller size. A potential solution is located in the space of multilayer feedforward ANNs (see Section 2.6.6) where any two neurons from neighbouring layers surely have a connection. In order to define this structure, it is necessary to set the sizes of blocks s_1 and s_2 for hidden and output layers accordingly, the maximal number of neurons in a hidden layer a_1, the number of neurons in the output layer a_2, the maximal number of hidden layers n_1 and the number of bits k required for coding an activation function type. Each neuron of a hidden

layer occupies k+1 bits (the first bit is equal to 1 if the neuron and the corresponding connections with other neurons exist, otherwise it is equal to 0). In that way, the general length of the binary string is calculated as follows:

$$l = n_1 \frac{a_1}{s_1}(k+1) + \frac{a_2}{s_2}k, \tag{5.1}$$

where s_1 and s_2 are to be aliquot parts of a_1 and a_2 accordingly. The approach proposed in (Bukhtoyarov et al., 2011) implements a special case of this encoding when $s_1 = s_2 = 1$.

The following activation functions are used:

- Linear function:

$$f_0(x) = \frac{a}{10}x + 0.5\,. \tag{5.2}$$

- Sigmoidal function:

$$f_1(x) = \frac{1}{1+e^{-ax}}\,. \tag{5.3}$$

- Hyperbolic tangent:

$$f_2(x) = \frac{\tanh(ax)}{2} + 0.5\,. \tag{5.4}$$

- Rational sigmoidal function:

$$f_3(x) = \frac{x}{2(|x| + 1/a)} + 0.5\,. \tag{5.5}$$

In this encoding, a is constant and equals to 1.

As a weight optimization algorithm for the fitness function calculation we use error backpropagation. Due to the features of the algorithm, each activation function is required to have a continuous derivative.

However, this encoding yields some essential disadvantages. The first one is transposition sensitivity. It means that there can be two different binary strings (genotypes, see Section 2.5.4.1) representing the same ANN structure (phenotype). This effect increases the search space and slows the algorithm down. Another disadvantage is a risk of so-called destructive recombination. There are different entities in the encoding (neurons belonging to different layers), and recombination between them does not make any sense. It is possible to avoid this risk using the uniform recombination operator only.

5.5.2. Novel ANN structure optimization with GA

Based on the disadvantages of the baseline approach described in Section 5.5.1 we found out that the baseline encoding was excessive and proposed a novel approach to the ANN structure representation. In fact, the novel encoding is a simplified version of the original one. It deals with whole ANN layers, not with separate blocks of neurons; all the neurons in one layer have the same activation function. However, a new tuning tool was added; all the activation functions were parameterized with the specific parameter *a* (see equations 5.1 – 5.5). It is possible to get different forms of each activation function changing this parameter.

In that way, it is necessary to set the maximal number of hidden layers, the maximal number of neurons in each hidden layer, the number of activation functions and to discretize the parameter *a* of the activation functions. The required number of bits for each entity can be found after these specifications. The general length of the binary string is calculated as follows:

$$l = n_1(m + k + k_a) + k + k_a, \tag{5.6}$$

where n_1 is the max number of hidden layers, m is the number of bits for coding the number of neurons in each hidden layer, k is the number of bits for coding the activation function kind, and k_a is the number of bits for coding the parameter a. We obtained a less flexible model than the original one. However, it has reasonable dimensionality, fewer parameters for tuning, and there is no transposition sensitivity. It requires much less resource of the GA and works faster.

Therefore, the proposed approach of neural network design may be useful for text classification and especially for real-time utterance classification in SDSs.

5.5.3. Evaluation

At first, a comparison of the novel approach with the baseline one (the Ward ANN, see Section 5.5.1) should be performed based on numerical experiments. Due to the time-consuming structure optimization of ANNs, such a comparison was performed with the lowest dimensionality obtained by the novel FT (20 features), only with the best term weighting – TRR, and only for data configuration 2.

The self-adjusting genetic algorithm (see Section 2.5.4.2) was applied for the baseline and the novel approaches of ANN structure optimization. The error backpropagation algorithm was applied for neuron weight optimization for both cases.

The numerical experiment in *RapidMiner* 5.0 with a fixed ANN structure showed that an ANN with one hidden layer of at least 21 neurons is sufficient for the considered classification problem. The classification efficiency could not be improved with an ANN containing more than one hidden layer: the ANN remembered the training examples but lost the ability to generalize them. Therefore, it was decided to set the maximal number of hidden layers equal to 1 and the maximal number of neurons in a hidden layer equal to 32 with a reserve on the safe side.

We obtained some optimization tasks of different dimensionality with the baseline ANN encoding (Ward encoding). The dimensionality depends on the sizes of blocks in the hidden and output layers (see Section 5.5.1). These sizes were chosen as follows: all aliquot parts of 32 and 20 were sorted in ascending order and then were grouped in pairs. There were 6 different pairs in total.

We used the resource of the GA algorithm proportional to the length of a binary string in each case. The maximal length of the binary string with the blocks (1,1) is equal to 136. In this case we used 110 generations of GA and a population size of GA equal to 110. The minimal length of the binary string with the blocks (32,20) is equal to 5. In this case we used exhaustive search instead of GA due to the small volume of the search space.

The solutions include one hidden layer with 30 neurons (for the blocks pairs containing the first block size which is an aliquot part of 30) or 32 neurons (for other blocks pairs) in it. A *t*-test with the confidential probability 0.95 was performed for all different pairs of the solutions. In general, the classification performance depends on the sizes of blocks: the ANN with the blocks (2,2) works significantly worse than with the other blocks (*F1*-score = 0.670), the ANN with the blocks (8,5) performs significantly better than with the other blocks (*F1*-score = 0.684), and there is no significant difference between the ANNs with the blocks (1,1), (4,4), (16,10) and (32,20). All these implementations of the ANN structural optimization perform significantly better than the ANN with the fixed ANN structure. However, they are more time-consuming; the simplest version requires approximately 10 minutes, and the most complicated one requires 11 hours of computational time.

After the experiments with the baseline approach, the novel approach was tested. In our case, the length of the binary string with the novel approach equals 19; it is 7 times smaller than the max binary string length for the baseline approach with the blocks (1,1).

Moreover, the difference between the dimensionalities increases exponentially when we increase the max structure of an ANN, i.e. in case of higher dimensionality.

We obtained a self-adjusting GA implementation with effective convergence and reasonable resource consumption (a population size was 50 individuals, the generation number was 50; the resource is 5 times less than for the baseline approach with the blocks (1,1)).

The results obtained on the "Speech Cycle" (data configuration 2, the novel FT) show that the novel approach of the ANN structural optimization performs well; the mean *F1*-score with TRR is equal to 0.684; according to the *t*-test with the confidential probability 0.95, it is significantly better than the *RapidMiner* implementation of a simple ANN trained with the error backpropagation algorithm without structure optimization. Furthermore, it is also significantly better than the baseline GA-based ANN structural optimization. There is only one case when the novel and the baseline encodings yield no significant difference: when the baseline approach uses the blocks (8,5). Generally, the structure of the solution obtained (1 hidden layer with 30 neurons) remains the same as with the baseline approach. At the same time, the novel approach requires much less resources and computational time than the baseline one in order to get a better result.

The comparison of different ANN structure optimization algorithms that use the error backpropagation weight optimization is illustrated in Table 5.15.

Table 5.15

Results of different ANN structure optimization algorithms

Algorithm	**Binary string length**	**Resource (calculations of fitness fucntion)**	**Mean *F1*-score**
Fixed structure ANN (*RapidMiner* implementation)	-	-	0.640
Ward ANN (1,1)	136	12100	0.673
Ward ANN (2,2)	68	8100	0.670
Ward ANN (4,4)	34	4900	0.678
Ward ANN (8,5)	20	2500	**0.684**
Ward ANN (16,10)	10	exhaustive search	0.675
Ward ANN (32,20)	5	exhaustive search	0.675
Novel encoding ANN	19	2500	**0.684**

Therefore, the solution encoding of the novel approach has an effective balance between the ANN structure representation flexibility and the problem dimensionality, also it has fewer parameters for tuning. Due to this encoding, the GA results in a significantly stronger improvement of the classification result (the mean *F1*-score is

0.684) than the baseline method, excluding the only implementation when the Ward ANN in the baseline approach contains blocks (8,5). Only in this case there is no significant difference between the encodings. However, if sizes of blocks in the Ward ANN have been chosen in a wrong way, the result becomes significantly worse. Moreover, the novel approach does not complicate the solution structure and requires less computational resources than the baseline approach. The optimization problem dimensionality is seven times lower than for the most complicated implementation of the baseline approach. The difference between the dimensionalities of the baseline and the novel approaches increases exponentially with an increase of the max structure of an ANN. This enables a more effective way of problem solving. An implementation of the novel approach with the self-adjusting GA solves the problem of GA parameter setting.

After the comparative study with the low dimensionality obtained by the novel FT, the novel approach to ANN structure optimization have been applied for those cases with all terms (without dimensionality reduction), with all seven considered term weighting methods, and for all three data configurations (see Section 3.2.1) on the "Speech Cycle" corpus. The results are presented in Table 5.16.

Table 5.16

Results of ANN without dimensionality reduction (*F1*-score)

Term weighting method	Data configuration 1	Data configuration 2	Data configuration 3
TRR	**0.844**	**0.703**	**0.631**
RF	0.837	0.682	0.615
NTW	0.832	0.699	0.623
TM2	0.715	0.621	0.601
CW	0.766	0.654	0.617
GR	0.731	0.662	0.626
IDF	0.699	0.679	0.629

The results demonstrate that TRR is the best term weighting method with ANN respect to the other algorithms in the previous results (see the results presented in Section 4.3). Generally, the classification results of ANN with all terms and with the novel FT are significantly worse than those obtained with *k*-NN and SVM-FML (see Section 4.3). For example, the best results of *k*-NN with all terms are the following: data configuration 1 – 0.873 (ANN – 0.844), data configuration 2 – 0.786 (ANN – 0.703), data configuration 3 – 0.715 (ANN – 0.631). The best result of *k*-NN with the novel FT for data configuration 2 – 0.754 (ANN – 0.684).

Finally, the self-adjusting GA (see Section 2.5.4.2) has been applied as a wrapper (see Section 2.5.4.1) with ANN generated by the self-adjusting GA with the novel encoding. The fitness function is the *F1*-score on the validating set. The final result is calculated on the test set after optimization. The wrapper tries to find a feature subset which provides the best classification results.

The results with the wrapper based on self-adjusting GA are presented in Figure 5.14 for three data configurations. The results are presented only with the best term weighting method - TRR.

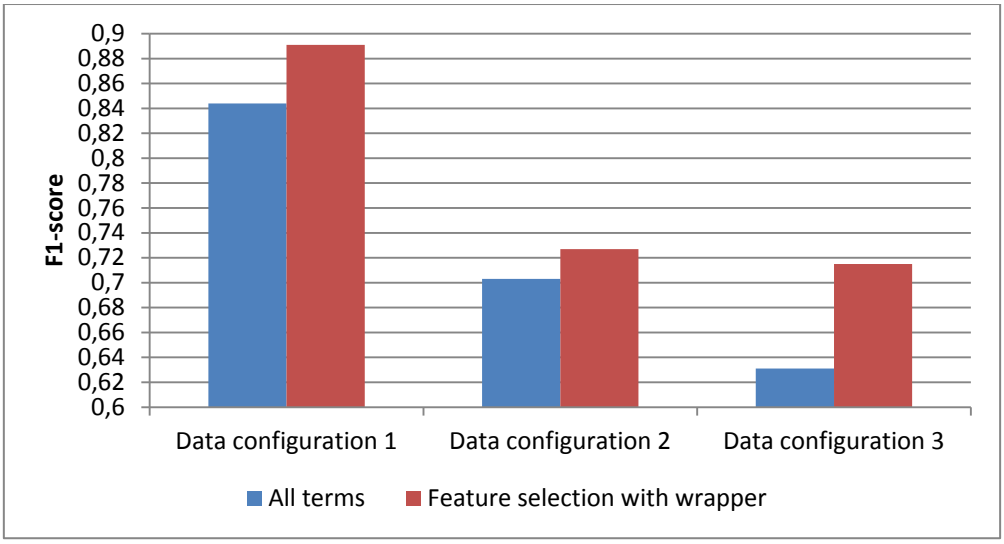

Figure 5.14 – Results of the GA-based wrapper for data configuration 1

The GA-based wrapper allows to significantly improve classification results with ANN. The result with ANN after feature selection for data configuration 1 significantly outperforms the best results with *k*-NN and SVM-FML presented in Section 4.3: (ANN – 0.891, *k*-NN – 0.873, SVM-FML – 0.873). For data configuration 3 ANN with the wrapper is able to achieve the same results as *k*-NN and SVM-FML (ANN – 0.715, *k*-NN – 0.715, SVM-FML – 0.721). Additionally, the best results with ANN are obtained with significantly reduced feature sets. The dimensionalities with ANN after performing the wrapper are the following: 2369 (71.7% of the original dimensionality) for data configuration 1; 2258 (68.3% of the original dimensionality) for data configuration 2; 2407 (72.9% of the original dimensionality) for data configuration 3.

We observe that the wrapper based on a self-adjusting GA is an effective tool with ANN and provides significant improvement of classification results simultaneously yielding a significant dimensionality reduction.

Therefore, the novel approach to neural network design with a simplified representation for structure optimization requires less computational resources than earlier proposed methods. Moreover, it has fewer parameters for tuning in comparison with the baseline ANN structure optimization approach. Term selection based on the wrapper with a self-adjusting GA is effective with the novel approach of ANN design.

5.6. Summary

The following novel techniques were proposed that may improve the existing text classification systems and may be also suitable for utterance classification in SDSs:

1. *Collectives of different term weighting methods* are able to integrate advantages of different approaches. Meta-classification with collectives of different term weighting may be performed with rule induction or a majority vote. The numerical experiments show that collectives of term weighting methods based on the majority voting procedure may significantly improve the classification performance of utterance classification with the *k*-NN algorithm. The performance of the vote procedure may be significantly improved with the weight optimization. For this task the self-adjusting genetic algorithm may be effectively applied. Therefore, collectives of different term weighting methods lead to improve the utterance classification results (the second aim of the thesis).

2. *A novel feature transformation method based on terms belonging to classes*. This method significantly reduces the dimensionality. In this case, the number of attributes equals the number of classes. The novel FT may reduce the dimensionality without a significant decrease of classification performance. The numerical experiments show that the method is the more effective with the *k*-NN algorithm than with SVM-FML. This approach may be especially useful for real-time speech-based text classification systems that can be incorporated into spoken dialogue systems. The approach improves the computational effectiveness of text and utterance classification (the second aim of the thesis).

The novel feature transformation method is especially effective in terms of classification accuracy in combination with collectives of term weighting methods. The simultaneous use of two novel approaches may significantly improve both the classification results and the computational effectiveness with the *k*-NN algorithm.

3. *A novel approach to neural network design with a simplified representation for structure optimization* that requires less computational resources than earlier proposed

methods. Moreover, it has fewer parameters for tuning in comparison with the baseline ANN structure optimization approaches. This approach also leads an improvement of the computational effectiveness of text classification (the second aim of the thesis). The proposed approach may be implemented with a self-adjusting genetic algorithm for solving the problem setting algorithm parameters.

Feature selection based on the wrapper with a self-adjusting GA demonstrates high efficiency with the novel approach of ANN design. It significantly improves of the classification results with ANN and significantly reduces dimensionality reduction.

6. Conclusions and Future Directions

6.1. Conclusions

In this thesis the following utterance classification problems were considered: domain detection of user utterances and user state recognition including user verbal intelligence recognition and text-based user emotion recognition. Effectively solving these problems will lead to a new generation of human-machine interfaces: multi-domain user-adaptive spoken dialogue systems that can be useful in very various applications such as intelligent houses or social robotics.

The research had three aims:

1. To identify the best combinations of term weighting methods, dimensionality reduction techniques, and classification algorithms for speech-based utterance classification problems for SDSs. Almost all existing comparative studies of text classification are based on corpora of written text. Spoken utterance classification problems have some specific features: shorter documents and smaller dictionaries in comparison with written text. Therefore, the comparative study of text classification approaches applied for utterance classification is novel.

2. To improve utterance classification performance for SDSs.

3. To improve computational performance of utterance classification for SDSs. This aim is important because SDSs are real-time systems that should be able to perform processing quickly with constrained computational resources.

Five speech-based corpora were considered in the thesis: two corpora for domain detection of user utterances in English („Speech Cycle“ and “Rafaeli”), two corpora for user verbal intelligence recognition in German (the monologue and the dialogue corpora), and one corpus for text-based emotion recognition in German (the VAM corpus).

Regarding the first research aim, different term weighting methods (TF-IDF, CW, GR, TM2, RF, TRR, and NTW), different dimensionality reduction methods („stop-word“ filtering in combination with stemming, weight-based feature selection, feature transformation based on term clustering), and different machine learning algorithms (*k*-NN, SVM, the Rocchio classifier) were applied to considered corpora.

The results obtained on the “Speech Cycle” and “Rafaeli” corpora showed that text classification approaches demonstrate an appropriate performance for domain detection of user utterances in general.

At first, the best combinations of a term weighting method and a classification algorithm for domain detection of user utterances were found with the full dimensionality (without dimensionality reduction). These combinations in this case seem to be the following: *k*-NN + TRR or SVM + IDF. After that, the dimensionality reduction techniques were tested. The results showed that feature selection based on filtering approaches (weight-based term filtering or „stop-word” filtering in combination with stemming) seems to be not appropriate as dimensionality reduction for utterance classification. This demonstrates differences between speech-based utterance and written text classification and the importance of the performed comparative study. The existing comparative studies of the approaches applied for written text show that filtering approaches are usually effective for written text classification. However, this is not observed for utterance classification. Another dimensionality reduction method, feature transformation with weigh-based term clustering, demonstrates more encouraging results for utterance classification. It significantly reduces the dimensionality without a decrease in the classification performance. Feature transformation with weigh-based term clustering is the most efficient for domain detection of user utterances with TRR as a term weighting method.

The results obtained on the corpora for verbal intelligence recognition showed that it seems to be easier to recognize verbal intelligence in dialogues than in monologues. It may be explained by the fact that verbal intelligence is expressed more clearly during a dialogue. The best results on the dialogue corpus for verbal intelligence recognition are obtained with the confident weights (CW) method as term weighting and *k*-NN as a classification algorithm. The CW method for verbal intelligence recognition provides extremely small number of useful terms that characterize only a class of higher verbal intelligence.

Emotion recognition based on linguistic information does not demonstrate high performance in comparison with audio-based and video-based emotion recognition. However, the results obtained with linguistic information may be helpful for multi-modal emotion recognition. The best results of text-based emotion recognition are obtained with NTW as a term weighting method, *k*-NN as a classification algorithm, and without dimensionality reduction.

Therefore, the best combinations of state-of-the-art approaches were identified for the different utterance classification tasks and the novel characteristics of these tasks were recognized.

After that, the novel approaches were proposed and tested for achieving the second and the third aims of the thesis.

Regarding the second research aim, the novel approach based on collectives of different term weighting methods was proposed. This approach allows to make use of the advantages of different term weighting methods. Collectives of term weighting methods based on the majority voting procedure may significantly improve the classification performance of utterance classification with *k*-NN algorithm. These results may be significantly improved by weighted voting with an optimization based on the self-adjusting genetic algorithm.

For the third research aim, the novel feature transformation based on term belonging to classes was proposed. It significantly reduces the dimensionality: it equals to the number of classes. This dimensionality reduction may be performed without a significant decrease in the classification results. The novel feature transformation significantly improves the computational performance of utterance classification. The novel feature transformation method is especially efficient in combination with the collectives of term weighting methods. The simultaneous use of two novel approaches may significantly improve both the classification results and the computational performance.

The novel approach to neural network structure optimization was also proposed for the third research aim of the thesis (to improve the computational performance). The novel approach has a simplified ANN structure representation and requires less computational resource than the baseline approach. Moreover, it has fewer parameters for tuning in comparison with the baseline ANN structure optimization approach. The self-adjusting genetic algorithm may be applied with this approach for solving the problem of GA parameter setting. Additionally, the results of the novel approach to ANN structure optimization may be significantly improved with feature selection based on the wrapper. The wrapper may be also performed by the self-adjusting GA. The novel approach to ANN design significantly improves the classification results and the computational performance of utterance classification with ANN.

Therefore, the novel approaches lead to improve the classification performance of utterance classification (the second aim of the thesis) and the computational performance of utterance classification (the third aim of the thesis) as well.

6.2. Main Contributions

The main contributions of this thesis fall into three fields: theoretical, practical, and experimental ones.

1. *Theoretical contributions*

a. Novel approach of collectives of term weighting methods. It is based on the majority or weighted voting. It is able to improve the utterance classification performance with *k*-NN. Weight optimization for voting may be effectively performed with the self-adjusting genetic algorithm. The novel approach may be applied for text classification in general. The results obtained with the collectives of term weighting methods are presented in (Sergienko et al., 2016a; Sergienko et al., 2016b; Sergienko et al., 2016c):

b. Novel feature transformation method based on terms belonging to classes. It is able to significantly decrease the dimensionality (it equals to number of the classes) of the classification problem and significantly improves the computational performance of utterance classification in terms of computational time. The simultaneous use of two novel approaches (collectives of term weighting methods and the novel feature transformation method) may significantly improve both the classification results and the computational performance with *k*-NN. The novel approach may be also applied for text classification in general. The results with the novel feature transformation method are presented in (Sergienko et al., 2016a; Sergienko et al., 2016d).

c. Novel approach to ANN structure optimization. It requires less computational resources and has fewer parameters for tuning than the baseline approaches. The novel method results in an improvement of the computational performance of utterance classification without a decrease in the classification performance. Additionally, the classification results of the novel approach may be improved with the wrapper based on the self-adjusting GA. The novel approach may be applied for all tasks with ANNs. The results with the novel approach to ANN design are presented in (Sergienko et al., 2015a).

2. *Practical contributions*

a. The scripts and the program modules written with the languages *C#*, *R*, and *Python* for implementations of the term weighting methods, the dimensionality reduction techniques, and the classification algorithms applied for utterance classification.

3. *Experimental contributions*

a. Extensive evaluations have been carried out in order to evaluate the developed algorithms and approaches. For domain detection of user utterances, the best combinations of state-of-the-art approaches including the term weighting methods, the dimensionality reduction techniques, and the classification algorithms were found term relevance ratio (TRR) + *k*-NN or inverse document frequency (IDF) + SVM with feature transformation based on term clustering as a dimensionality reduction technique. The use of the novel approaches improves the best results of the existing approaches. These results are presented in the (Sergienko et al., 2016a; Sergienko et al., 2016d).

b. For verbal intelligence recognition, the best results are obtained with the confident weights (CW) method as term weighting and *k*-NN as a classification algorithm. The CW method for verbal intelligence recognition provides a considerable small number of useful terms that characterize only a class of higher verbal intelligence. It seems to be easier to recognize verbal intelligence in dialogues than in monologues. These results are presented in (Sergienko and Schmitt, 2015).

c. For text-based emotion recognition, the best methods were found: NTW as a term weighting method, *k*-NN as a classification algorithm, and without dimensionality reduction. However, even the best methods do not demonstrate a high classification performance. Nevertheless, it may be possible to organize fusion linguistic information with audio and video features for multi-modal emotion recognition.

6.3. Future Directions

At first, the results of utterance classification may be improved based on a direct extension of the research described in this thesis. For instance, a fusion of text-, audio-, and video-based approaches should be performed for multi-modal user emotion

recognition. Such a fusion may be performed in a decision-level (the fusion of classifiers) or a feature-level (the fusion of features) ways.

The theoretical direction of future researches may include an investigation of some approaches that were not applied in the thesis. It means that novel effective approaches of utterances preprocessing (term weighting methods, dimensionality reduction techniques, or novel approaches of text representation) and utterance classification may be proposed. Novel approaches may improve both classification and computational performance of utterance classification. We illustrate two examples of approaches that may lead to improve the utterance classification results presented in this thesis:

- Application of „deep learning“ approaches (Lee et al., 2009; Hinton et al., 2012) based on deep belief neural networks for speech-based text classification. These approaches demonstrate encouraging results in different domains including natural language understanding (Sarikaya et al., 2014).

- Using *n*-gram approaches (Cavnar and Trenkle, 1994; Peng and Schuurmans, 2003) for speech-based text classification. In this case we put not only individual words or terms as features but also consequences of terms with length equals *n*. This may improve classification performance of the considered utterance classification problems.

A potential practical direction for future work is the implementation of utterance classification systems in real spoken dialogue systems. It means an implementation of a module for domain detection of user utterances and a user state recognition module which will be able to perform user verbal intelligence recognition and user emotion recognition in real spoken dialogue systems. Such a SDS would be multi-domain and user-adaptive. It will be necessary to train utterance classification systems with test users and evaluate utterance classification with real users. Theoretically, new characteristics of utterance classification may be recognized during such implementations and novel approaches should be developed and investigated. It would be interesting to test and evaluate multi-domain user-adaptive SDSs in different domains and applications: social robotics, intelligent houses etc.

Additionally, novel corpora may be considered for utterance classification. These corpora may contain more linguistic information and yield a more balanced distribution of the classes than the ones considered in this thesis. Corpora with direct conversations between a user and a SDS may be considered for user state recognition including verbal intelligence and emotion recognition. Furthermore, the results obtained on the corpora of various languages would be also interesting. The existing speech recognition systems are

effective for European languages such as English and German. However, speech recognition with other languages such as Chinese, Russian, or Arabic faces some difficulties. Therefore, it is important to investigate how a higher speech recognition error rate with these languages effects on an utterance classification performance. Moreover, non-European languages may yield some unknown characteristics of user state recognition. For instance, emotions are expressed in different ways in different cultures.

List of Own Publications

Parts of this thesis are based on the material published in the following papers.

As first author:

- **Book chapters:**
  - R. Sergienko, T. Gasanova, E. Semenkin and W. Minker. *Collectives of Term Weighting Methods for Natural Language Call Routing*. **Informatics in Control, Automation and Robotics**, Springer International Publishing Switzerland, Series: Lecture Notes in Electrical Engineering, pp. 99-110, January 2016.
- **Journal articles:**
  - R. Sergienko, M.Shan, W. Minker, E. Semenkin. *Topic Categorization Based on Collectives of Term Weighting Methods for Natural Language Call Routing*. **Journal of Siberian Federal University. Mathematics & Physics**, Num. 2, pp. 235-245, June 2016.
  - R. Sergienko, M. Shan, A. Khan, T. Gasanova and W. Minker. *Feature selection for text classification based on constraints for term weights*. **Bulletin of Siberian State Aerospace University**, Vol. 16, Num. 1, pp. 119-123, March 2015.
  - R. Sergienko, W. Minker, E. Semenkin and T. Gasanova. *Call Routing Problem Solving Method Based on a New Term Relevance Estimation*. **Program Products and Systems**, Num. 1, pp. 90-93, March 2013.
- **Conference Contributions:**
  - R. Sergienko, I. Kamshilova, E. Semenkin and A. Schmitt. *Weighted Voting of Different Term Weighting Methods for Natural Language Call Routing*. **Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2016)**, Lisbon, Portugal, July 2016.
  - R. Sergienko, M. Shan and W. Minker. *A Comparative Study of Text Preprocessing Approaches for Topic Detection of User Utterances*. **Proceedings of the 10th edition of the Language Resources and Evaluation Conference (LREC 2016)**, Portorož, Slovenia, May 2016.
  - R. Sergienko, M. Shan and A. Schmitt. *A Comparative Study of Text Preprocessing Techniques for Natural Language Call Routing*. **Proceedings of the 7th International Workshop On Spoken Dialogue Systems (IWSDS 2016)**, Saariselkä, Finland, January 2016.

- R. Sergienko and A. Schmitt. *Verbal Intelligence Identification Based on Text Classification*. **Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH 2015)**, Dresden, Germany, September 2015.
- R. Sergienko, O. Akhtiamov, E. Semenkin and A. Schmitt. *A novel approach to neural network design for natural language call routing*. **Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2015)**, Colmar, France, Vol. 01, pp. 102 - 109, July 2015.
- R. Sergienko, T. Gasanova, E. Semenkin and W. Minker. *Text Categorization Methods Application for Natural Language Call Routing*. **Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2014)**, Vienna, Austria, Vol. 2, pp. 827-831, September 2014.

Co-authored publications:

- **Journal articles:**
  - A. Spirina, M. Sidorov, R. Sergienko, E. Semenkin, W. Minker. *Human-human task-oriented conversation corpus for interaction quality modelling*. **Bulletin of Siberian State Aerospace University**, Num. 1, March 2016.
  - T. Gasanova, R. Sergienko, W. Minker and E. Zhukov. *A New Method for Natural Language Call Routing Problem Solving*. **Bulletin of Siberian State Aerospace University**, Num. 4, pp. 108-112, December 2013.
  - T. Gasanova, R. Sergienko, W. Minker and E. Semenkin. *Text Categirization by Coevolutionary Genetic Algorithm*. **Bulletin of Siberian State Aerospace University**, Num. 4, pp. 104-108, December 2013.
- **Conference Contributions:**
  - O. Akhtiamov, R. Sergienko and W. Minker. *An Approach to off-Talk Detection based on Text Classification within an Automatic Spoken Dialogue System*. **Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2016)**, Lisbon, Portugal, July 2016.
  - A. Spirina, M. Sidorov, R. Sergienko and A. Schmitt. *First Experiments on Interaction Quality Modelling for Human-Human Conversation*. **Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2016)**, Lisbon, Portugal, July 2016.

- S. Akhmedova, E. Semenkin and R. Sergienko. *Automatically Generated Classifiers for Opinion Mining with Different Term Weighting Schemes*. **Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2014)**, Vienna, Austria, Vol. 2, pp. 845-850, September 2014.
- T. Gasanova, R. Sergienko, E. Semenkin and W. Minker. *Dimension Reduction with Coevolutionary Genetic Algorithm for Text Classification*. **Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2014)**, Vienna University of Technology, Austria, Vol. 1, pp. 215-222, September 2014.
- T. Gasanova, R. Sergienko, S. Akhmedova, E. Semenkin and W. Minker. *Opinion Mining and Topic Categorization with Novel Term Weighting*. **Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis**, Association for Computational Linguistics, Baltimore, Maryland, USA, pp. 84–89, June 2014.
- T. Gasanova, R. Sergienko, W. Minker, E. Semenkin and E. Zhukov. *A Semi-supervised Approach for Natural Language Call Routing*. **Proceedings of the SIGDIAL 2013 Conference**, pp. 344-348, August 2013.

Bibliography

Akhmedova, S., Semenkin, E., and Sergienko, R. (2014). Automatically generated classifiers for opinion mining with different term weighting schemes. In *Informatics in Control, Automation and Robotics (ICINCO), 2014 11th International Conference on*, Vol. 2, 845-850. IEEE.

Baharudin, B., Lee, L. H., and Khan, K. (2010). A review of machine learning algorithms for text-documents classification. *Journal of advances in information technology*, 1(1), 4-20.

Baker, L. D. and McCallum, A. K. (1998). Distributional clustering of words for text classification. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, 96-103. ACM.

Balabanovic, M. and Shoham Y. (1997). Content-based, Collaborative Recommendation. *Communications of the Association for Computing Machinery* 40(3), 66-72, 1997.

Bijankhan, M., Sheikhzadegan, J., Roohani, M. R., Samareh, Y., Lucas, C., and Tebyani, M. (1994, December). FARSDAT-The speech database of Farsi spoken language. In the *Proceedings of the Australian Conference on Speech Science and Technology*, Vol. 2, 826-830.

Blanchard, A. (2007). Understanding and customizing stopword lists for enhanced patent mapping. *World Patent Information*, 29(4), 308-316.

Bos, J., Klein, E., Lemon, O., and Oka, T. (2003). DIPPER: Description and formalisation of an information-state update dialogue system architecture. In *4th SIGdial Workshop on Discourse and Dialogue*, 115-124.

Boser, B., Guyon, I., and Vapnik, V. (1992). A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on COLT*, 144-152, Pittsburgh.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2), 123-140.

Brown, P. F, Desouza, P. V., Mercer, R. L, Pietra, V. D., and Lai, J. C. (1992). Class-based n-gram models of natural language. *Computational linguistics*, 18(4), 467-479.

Bukhtoyarov, V. and Semenkina, O. (2010). Comprehensive evolutionary approach for neural network ensemble automatic design. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*. IEEE.

Bukhtoyarov, V., Semenkin E. and Sergienko, R. (2011). Evolutionary Approach for Automatic Design of Neural Networks Ensembles for Modeling and Time Series Forecasting. *Proceedings of the IADIS International Conference Intelligent Systems and Agents*, 93-96.

Burstein, J., Chodorow, M., and Leacock, C. (2003). Criterion online essay evaluation: An application for automated evaluation of student essays. In *Automated essay scoring: a cross disciplinary approach*. M. D., Shermis and J. C. Burstein, Eds. Lawrence Erlbaum Associates.

Cambria, E. and Bruce W. (2014). Jumping NLP curves: a review of natural language processing research. *Computational Intelligence Magazine*, IEEE 9.2, 48-57.

Cannas, L. M., Dessì, N. and Dessì, S. (2012). A Model for Term Selection in Text Categorization Problems. *Database and Expert Systems Applications (DEXA), 2012 23rd International Workshop on*, 169-173. IEEE.

Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., and Rosen, D. B. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *Neural Networks, IEEE Transactions on*, 3(5), 698-713.

Cavnar, W. B., and Trenkle, J. M. (1994). N-gram-based text categorization. *Ann Arbor MI*, 48113(2), 161-175.

Cianciolo, A. T. and Sternberg, T. J. (2004). *Intelligence: a Brief History*. Blackwell Publishing.

Cohen, W. W. (1995). Fast effective rule induction. In *Proceedings of the twelfth international conference on machine learning*, 115–123.

Colas, F. and Brazdil, P. (2006). Comparison of svm and some older classification algorithms in text classification tasks. *Artificial Intelligence in Theory and Practice*, 169-178.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), 273-297.

Debole, F. and Sebastiani, F. (2004). Supervised term weighting for automated text categorization. In: *Text mining and its applications*, 81–97. Springer.

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R A. (1990). Indexing by latent semantic analysis. *JASIS*, 41(6), 391-407.

Deng, Z. H., Tang, S. W., Yang, D. Q., Li, M. Z. L. Y., and Xie, K. Q. (2004). A comparative study on feature weight in text categorization. In *Advanced Web Technologies and Applications*, 588-597. Springer Berlin Heidelberg.

Dragut, E., Fang, F., Sistla, P., Yu, C., and Meng, W. (2009). Stop word and related problems in web interface integration. *Proceedings of the VLDB Endowment*, 2(1), 349-360.

Dumais, S., Platt, J., Heckerman, D., and Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. *Proceedings of the seventh international conference on Information and knowledge management*, 148-155. ACM.

Dybkjær, L., and Minker, W. (Eds.). (2008). *Recent trends in discourse and dialogue*, Vol. 39, Springer Science & Business Media.

Elliot, S. (2003). Intellimetric: from here to validity. In Automated essay scoring: a cross disciplinary approach, M. D. Shermis and J. C. Burstein, Eds. Lawrence Erlbaum Associates, 2003.

Fan, R. E., Chang, K. W., Hsieh C. J., Wang X. R., Lin C. J. (2008). Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9, 1871–1874.

Fensel, D. (2001). *Ontologies* (pp. 11-18). Springer Berlin Heidelberg.

Fernández-Martínez, F., Zablotskaya, K., and Minker, W. (2012). Text categorization methods for automatic estimation of verbal intelligence. *Expert Systems with Applications*, 39(10), 9807-9820.

Fix, E., Hodges J. J. L. (1951). *Discriminatory analysis-nonparametric discrimination: consistency properties*. Tech. rept. DTIC Document.

Fong, T., Nourbakhsh, I., & Dautenhahn, K. (2003). A survey of socially interactive robots. *Robotics and autonomous systems*, 42(3), 143-166.

Fox, C. (1989). A stop list for general text. In: *ACM SIGIR Forum*, ACM, vol. 24, 19–21.

Frederick, M. D. (1995). Neuroshell 2 user's manual.

Fuhr, N., Hartmann, S., Lustig, G., Schwantner, M., Tzeras, K., and Knorz, G. (1991). *AIR, X: a rule based multistage indexing system for large subject fields*. Citeseer.

Gabrilovich E. and Markovitch S. (2004) Text categorization with many redundant features: Using aggressive feature selection to make svms competitive with c4. 5. In *Proceedings of the twenty-first international conference on Machine learning*, 41. ACM.

Ganiz, M. C., George, C., and Pottenger, W. M. (2011). Higher order Naive Bayes: a novel non-IID approach to text classification. *Knowledge and Data Engineering, IEEE Transactions on*, 23(7), 1022-1034.

Galavotti, L., Sebastiani, F., and Simi, M. (2000). Experiments on the use of feature selection and negative evidence in automated text categorization. *Research and Advanced Technology for Digital Libraries*, 59-68. Springer.

Gasanova, T., Sergienko R., Minker W., Semenkin, E., and Zhukov, E. (2013). A Semi-supervised Approach for Natural Language Call Routing. *Proceedings of the SIGDIAL 2013 Conference*, 344-348.

Gasanova, T., Sergienko, R., Akhmedova, Sh., Semenkin, E., and Minker, W. (2014a). Opinion mining and topic categorization with novel term weighting. In: *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, ACL 2014*, 84–89. ACL.

Gasanova, T., Sergienko, R., Semenkin, E., and Minker, W. (2014b). Dimension Reduction with Coevolutionary Genetic Algorithm for Text Classification. In: *Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 215-222.

Geutner, P., Denecke, M., Meier, U., Westphal, M., and Waibel, A. (1998). Conversational speech systems for on-board car navigation and assistance. In *ICSLP*, Vol. 98, 14447-1450.

Gharavian, D., Sheikhan, M., Nazerieh, A., and Garoucy, S. (2012). Speech emotion recognition using FCBF feature selection method and GA-optimized fuzzy ARTMAP neural network. *Neural Computing and Applications*, 21(8), 2115-2126.

Goddard, C. (2011). *Semantic analysis: A practical introduction.* Oxford University Press.

Goethals, G. R., Sorenson, G. J., and Bruns, J. M. (Editors). (2004). *Encyclopedia of Leadership*. Sage Publications (CA).

Goldberg, D. E. and Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine learning*, 3(2), 95-99.

Goutte C. and Gaussier, E. (2005) A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In: *Advances in information retrieval*, 345–359. Springer.

Grimm, M., Kroschel, K., and Narayanan, S. (2008). The Vera am Mittag German audio-visual emotional speech database. In *Multimedia and Expo, 2008 IEEE International Conference on*, 865-868. IEEE.

Han, E. H. S., Karypis, G., and Kumar, V. (2001). *Text categorization using weight adjusted k-nearest neighbor classification*, 53-65. Springer Berlin Heidelberg.

Harb, H. and Chen, L. (2003). Gender identification using a general audio classifier. In *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*, Vol. 2, II-733. IEEE.

Hassan, S., Mihalcea, R., and Banea, C. (2007). Random walk term weighting for improved text classification. *International Journal of Semantic Computing*, 1(04), 421-439.

Hecht-Nielsen, R. (1989). Theory of the backpropagation neural network. In *Neural Networks International Joint Conference on (IJCNN 1989)*, 593-605. IEEE.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... and Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine*, IEEE, 29(6), 82-97.

Holland, J. H. and Reitman, J. S. (1977). Cognitive systems based on adaptive algorithms. *ACM SIGART Bulletin*, (63), 49-49.

Hull, D. (1994). Improving text retrieval for the routing problem using latent semantic indexing. *SIGIR*, 282-291. Springer.

Ishibuchi, H., Nakashima, T., and Murata, T. (1999). Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 29(5), 601-618.

Ittner, D., Lewis, D., and Ahn, D. (1995). Text Categorization of Low Quality Images. In: *Symposium on Document Analysis and Information Retrieval*, Las Vegas, NV, 301-315.

Joachims, T. (1996). *A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization* (No. CMU-CS-96-118). Carnegie-mellon univ pittsburgh pa dept of computer science.

Joachims, T. (1998). *Text categorization with support vector machines: Learning with many relevant features*. Springer.

Joachims, T. (2006). Training linear SVMs in linear time. In *ACM KDD*.

John, G. H., Kohavi, R., and Peger, K. (1994). Irrelevant Features and the Subset selection Problem, *ICML*, vol. 94, 121-129.

Jokinen, K., and McTear, M. (2009). Spoken dialogue systems. *Synthesis Lectures on Human Language Technologies*, 2(1), 1-151.

Jolliffe, I. (2002). *Principal component analysis*. John Wiley & Sons, Ltd.

Kamp, H. and Reyle, U. (2013). *From discourse to logic: Introduction to modeltheoretic semantics of natural language, formal logic and discourse representation theory*, Vol. 42. Springer Science & Business Media.

Karabulut, E. M., Özel, S. A. and Brikçi, T. 2012. A comparative study on the effect of feature selection on classification accuracy. *Procedia Technology*, 1, 323-327.

Kim, S. B., Han, K. S., Rim, H. C., and Myaeng, S. H. (2006). Some effective techniques for naive bayes text classification. *Knowledge and Data Engineering, IEEE Transactions on*, 18(11), 1457-1466.

Ko, Y. (2012). A study of term weighting schemes using class information for text classification. In: *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, 1029–1030. ACM

Komatani, K., Kanda, N., Nakano, M., Nakadai, K., Tsujino, H., Ogata, T., and Okuno, H. G. (2009). Multidomain spoken dialogue system with extensibility and

robustness against speech recognition errors. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, 9–17. Association for Computational Linguistics.

Kwon, O. W., Chan, K., Hao, J., and Lee, T. W. (2003). Emotion recognition by speech signals. In *INTERSPEECH*.

Lan, M., Tan, C. L., Low, H. B., and Sung, S. Y. (2005). A comprehensive comparative study on term weighting schemes for text categorization with support vector machines. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, 1032-1033. ACM.

Lan, M., Tan, C. L., Su J., and Lu. Y. (2009). Supervised and traditional term weighting methods for automatic text categorization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(4), 721–735.

Landauer, T. K., Foltz, P. W. and Laham, D. (1998). An introduction to latent semantic analysis. *Discourse processes*, 25(2-3), 259-284.

Lee, C., Jung, S., Kim, S., and Lee, G. G. (2009). Example-based dialog modeling for practical multi-domain dialog system. *Speech Communication*, 51(5), 466–484.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, 609-616. ACM.

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

Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. *Machine learning: ECML-98*, 4-15. Springer.

Leggetter, C. J., and Woodland, P. C. (1995). Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech & Language*, 9(2), 171-185.

Masand, B. Linoff, G., and Waltz, D. (1992). Classifying news stories using memory based reasoning. *Proceedings of the 15th annual international conference on Research and development in information retrieval ACM SIGIR*, 59-65. ACM.

Meguro, T., Higashinaka, R., Minami, Y., & Dohsaka, K. (2010). Controlling listening-oriented dialogue using partially observable Markov decision processes. In *Proceedings of the 23rd International Conference on Computational Linguistics*, 761-769. Association for Computational Linguistics.

Mengistu, K. T., and Wendemuth, A. (2007). Telephone-Based Spoken Dialog System Using HTK-based Speech Recognizer and VoiceXML. *Fortschritte der Akustik*, 33(2), 625.

Momtazi, S., and Klakow, D. (2009). A word clustering approach for language modelbased sentence retrieval in question answering systems. *Proceedings of the 18th ACM conference on Information and knowledge management*, 1911-1914. ACM.

Morariu, D., Vintan, L. N., and Tresp, V. (2005). Meta-classification using SVM classifiers for text documents. *Intl. Jrnl. of Applied Mathematics and Computer Sciences*, 1(1).

Moulinier, I. and Ganascia, J.-G. (1996). Applying an existing machine learning algorithm to text categorization. *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*, 34, 343-354. Springer.

Nothdurft, F., Honold, F., Zablotskaya, K., Diab, A., and Minker, W. (2014). Application of Verbal Intelligence in Dialog Systems for Multimodal Interaction. In *Intelligent Environments (IE), 2014 International Conference on*, 361-364. IEEE.

Ng, H. T., Goh, W. B. and Low, K. L. (1997). Feature selection, perceptron learning, and a usability case study for text categorization. *ACM SIGIR Forum*, vol. 31, 67-73. ACM.

Noy, N. F. (2004). Semantic integration: a survey of ontology-based approaches. ACM Sigmod Record, 33(4), 65-70.

Pazzani M., Billsus, D. (1997). Learning and Revising User Profiles. *The Identification of Interesting Web Sites. Machine Learning*, 27(3), 313-331.

Peng, F., and Schuurmans, D. (2003). Combining naive Bayes and n-gram language models for text classification (pp. 335-350). Springer Berlin Heidelberg.

Polzehl, T., Schmitt, A., and Metze, F. (2011). Salient features for anger recognition in German and English ivr portals. In *Spoken dialogue systems technology and design*, 83-105. Springer New York.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program: electronic library and information systems*, 14(3), 130-137.

Porter, M. F. (2001) *Snowball: A language for stemming algorithms*.

Potamianos, G., Neti, C., Luettin, J., and Matthews, I. (2004). Audio-visual automatic speech recognition: An overview. *Issues in visual and audio-visual speech processing*, *22*, 23.

Rafaeli A., Ziklik L., and Doucet, L. (2008). The impact of call center employees' customer orientation behaviors on service quality. *Journal of Service Research*, vol. 10, no. 3, 239–255.

Rocchio, J. J. (1971). *Relevance feedback in information retrieval*.

Rogati, M. and Yang, Y. (2002). High-performing feature selection for text classification. In *Proceedings of the eleventh international conference on Information and knowledge management*, 659–661. ACM.

Rose, T., Stevenson, M., and Whitehead, M. (2002). The Reuters Corpus Volume 1-from Yesterday's News to Tomorrow's Language Resources. In *LREC 2002*, vol. 2, pp. 827-832.

Ruiz, M. E. and Srinivasan, P. (1999). Hierarchical neural networks for text categorization. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 281-282. ACM.

Sakamura, K. (2005). Intelligent house in the age of ubiquitous computing. *A House of Sustainability: PAPI, A+ U*, 56-65.

Salton G. and Buckley C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5), 513–523.

Sarikaya, R., Hinton, G. E., and Deoras, A. (2014). Application of deep belief networks for natural language understanding. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 22(4), 778-784.

Schapire, R. E., and Singer, Y. (2000). BoosTexter: A boosting-based system for text categorization. *Machine learning*, 39(2), 135-168.

Scherer, K. R. (1997). Emotion. In *Sozialpsychologie*, 293-330. Springer Berlin Heidelberg.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. *Proceedings of international conference on new methods in language processing*, Manchester, UK, vol. 12, 44-49.

Schmitt, A., Polzehl, T., Minker, W., & Liscombe, J. (2010). The Influence of the Utterance Length on the Recognition of Aged Voices. In *LREC 2010*.

Schmitt, A. and Minker, W. (2012). *Towards Adaptive Spoken Dialog Systems*. Springer Science & Business Media.

Schölkopf, B., Sung, K. K., Burges, C. J., Girosi, F., Niyogi, P., Poggio, T., & Vapnik, V. (1997). Comparing support vector machines with Gaussian kernels to radial basis function classifiers. *Signal Processing, IEEE Transactions on*, 45(11), 2758-2765.

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

Semenkin, E. and Semenkina, M. (2012). Self-configuring genetic programming algorithm with modified uniform crossover. In *2012 IEEE Congress on Evolutionary Computation*.

Sergienko, R. and Semenkin, E. (2010). Competitive cooperation for strategy adaptation in coevolutionary genetic algorithm for constrained optimization. In *2010 IEEE Congress on Evolutionary Computation*.

Sergienko, R., Akhtiamov, O., Semenkin E., and Schmitt, A. (2015a). A novel approach to neural network design for natural language call routing. *Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2015)*, Vol. 1, 102 – 109.

Sergienko, R. and Schmitt, A. (2015b). Verbal Intelligence Identification Based on Text Classification. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH 2015)*.

Sergienko, R., Muhammad, S., and Minker, W. (2016a). A comparative study of text preprocessing approaches for topic detection of user utterances. In *Proceedings of the 10th edition of the Language Resources and Evaluation Conference (LREC 2016)*.

Sergienko, R., Gasanova, T., Semenkin E., and W. Minker. (2016b). Collectives of Term Weighting Methods for Natural Language Call Routing. *Lecture Notes in Electrical Engineering: Informatics in Control, Automation and Robotics*, 99-110. Springer International Publishing Switzerland.

Sergienko, R., Shan, M., Minker, W., and Semenkin, E. (2016c). Topic Categorization Based on Collectives of Term Weighting Methods for Natural Language Call Routing. *Journal of Siberian Federal University. Mathematics & Physics*, Num. 2, 246-256.

Sergienko, R., Shan, M., and Schmitt, A. (2016d). A Comparative Study of Text Preprocessing Techniques for Natural Language Call Routing. *Proceedings of the 7th International Workshop On Spoken Dialogue Systems (IWSDS 2016)*.

Shafait, F., Reif, M., Kofler, C., and Breuel, T. M. (2010). Pattern recognition engineering. In: *RapidMiner Community Meeting and Conference*, Citeseer, vol 9.

Shalev-Shwartz S., Singer, Y., and Srebro, N. (2007). Pegasos: primal estimated sub-gradient solver for SVM. In *ICML*.

Sidorov, M., Schmitt, A., Zablotskiy, S., and Minker, W. (2013). Survey of Automated Speaker Identification Methods. In *Intelligent Environments (IE), 2013 9th International Conference on*, 236-239. IEEE.

Sidorov, M., Ultes, S., and Schmitt, A. (2014a). Emotions are a personal thing: Towards speaker-adaptive emotion recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, 4803-4807. IEEE.

Sidorov, M., Ultes, S., and Schmitt, A. (2014b). Comparison of Gender-and Speaker-adaptive Emotion Recognition. *International Conference on Language Resources and Evaluation (LREC)*, Reykjavik, Iceland.

Sidorov, M., and Minker, W. (2014c). Emotion Recognition and Depression Diagnosis by Acoustic and Visual Features: A Multimodal Approach. *Audio/Visual Emotion and Depression recognition Challenge and Workshop (AVEC 2014)*.

Sidorov, M., and Minker, W. (2014d). Emotion Recognition in Real-world Conditions with Acoustic and Visual Features. *The Second Emotion Recognition In The Wild Challenge (EmotiW 2014)*.

Sidorov, M., Ultes, S., and Schmitt, A. (2014e). Automatic Recognition of Personality Traits: A Multimodal Approach. *MAPTRAITS'14 - Personality Mapping Challenge and Workshop 2014*.

Slonim, N. and Tishby, N. (2001). The power of word clusters for text classification. In: *23rd European Colloquium on Information Retrieval Research*, vol. 1.

Song, Y.-I., Han, K.-S., and Rim, H.-Ch. (2004). A term weighting method based on lexical chain for automatic summarization. *Computational Linguistics and Intelligent Text Processing*, 636-639. Springer.

Soucy, P. and Mineau, G. W. (2005). Beyond tfidf weighting for text categorization in the vector space model. In: *IJCAI*, vol 5, 1130–1135.

Suhm, B., Bers, J., McCarthy, D., Freeman, B., Getty, D., Godfrey, K., and Peterson, P. (2002). A comparative study of speech in the call center: natural language call routing vs. touch-tone menus. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, 283-290. ACM.

Tong, S. and Koller, D. (2002). Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2, 45-66.

Trappey, A. J. C., Hsu, F.-Ch., Trappey, Ch. V., and Lin, Ch.-I. (2006). Development of a patent document classification and search platform using a back-propagation network. *Expert Systems with Applications*, 31(4), 755-765.

Ultes, S., Schmitt, A., and Minker, W. (2011). Attention, Sobriety Checkpoint! Can Humans Determine by Means of Voice, if Someone is Drunk… and Can Automatic Classifiers Compete?. In *Twelfth Annual Conference of the International Speech Communication Association (INTERSPEECH 2011)*.

Vapnik V. N. (1995). *The Nature of Statistical Learning Theory*, Springer-Verlag New York, Inc.

Vapnik, V. and Chervonenkis, A. *Theory of Pattern Recognition*. Nauka, Moscow, 1974.

Vieira, S. M., Sousa, J. and Kaymak, U. (2012). Fuzzy criteria for feature selection. *Fuzzy Sets and Systems*, 189(1), 1-18.

Ward J. J. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301), 236–244.

Wechsler, D. (1982). *Handanweisung zum Hamburg-Wechsler-Intelligenztest fuer Erwachsene (HAWIE)*. Separatdr., Bern; Stuttgart; Wien, Huber.

Whitley, D., Starkweather, T., and Bogart, C. (1990). Genetic algorithms and neural networks: Optimizing connections and connectivity. *Parallel computing*, 14(3), 347-361.

Wiener, E., Pedersen, J. O., Weigend, A. S. (1995). A neural network approach to topic spotting. *Proceedings of SDAIR-95, 4th annual symposium on document analysis and information retrieval*, 317-332. Citeseer.

Wilson, E. B. (1927). Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158), 209-212.

Xu, H. and Li, C. (2007). A novel term weighting scheme for automated text categorization. In: *Intelligent Systems Design and Applications, 2007. ISDA 2007. Seventh International Conference on*, 759–764. IEEE.

Yang, J., and Honavar, V. (1998). *Feature subset selection using a genetic algorithm*. In Feature extraction, construction and selection, 117-136. Springer US.

Yang, Y. and Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In: *ICML*, vol. 97, 412–420.

Yang, Y. and Liu, X. (1999). A re-examination of text categorization methods. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 42-49. ACM.

Youn, S., and McLeod, D. (2007). A comparative study for email classification. In *Advances and innovations in systems, computing sciences and software engineering*, 387-391. Springer Netherlands.

Yu, B., Xu, Z., and Li, Ch. (2008). Latent semantic analysis for text categorization using neural network. *Knowledge-Based Systems*, 21(8), 900-904.

Yu, D., and Deng, L. (2012). Automatic Speech Recognition. Springer Signal and communication technology.

Yuan P., Chen, Y., Jin H., Huang L. (2008). MSVM-kNN: *Combining SVM and k-NN for Multi-Class Text Classification*. 978-0-7695-3316-2/08, IEEEDOI10.1109/WSCS.2008.

Zablotskaya, K., Walter, S., and Minker, W. (2010). Speech Data Corpus for Verbal Intelligence Estimation. *International Conference on Language Resources and Evaluation LREC 2010*, 1077-1080.

Zablotskaya, K., Fernández Martínez, F., and Minker, W. (2012). Investigating verbal intelligence using the TF-IDF approach. *International Conference on Language Resources and Evaluation LREC 2012*, 1573-1576.

Zhang, H., and Li, D. (2007). Naive Bayes text classifier. *In Granular Computing, 2007. GRC 2007. IEEE International Conference on*, 708-708. IEEE.

Zhang, M. L. and Zhou, Z. H. (2006). Multilabel neural networks with applications to functional genomics and text categorization. *Knowledge and Data Engineering, IEEE Transactions on*, 18(10), 1338-1351.

Zhang, Y., Jin, R., and Zhou, Z. H. (2010). Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, 1(1-4), 43-52.

Zhou, Y., Li, Y., and Xia, S. (2009). An improved KNN text classification algorithm based on clustering. *Journal of computers*, 4(3), 230-237.

Appendix

A.1. Results on the „Speech Cycle“ Corpus without Dimensionality Reduction

The ranks of term weighting methods based on *t*-test (the confidence probability equals 0.95) are illustrated in the brackets in tables A.1-A.4. The columns “*k*” contain average optimal values of *k* for *k*-NN algorithm after validation on the interval 1..15 with the validation sets. The results with Rocchio classifier are presented for three different metrics: „taxi cabine“ metric (TaxiCab), Euclidean metric (Euclid), and cosine similarity (Cos.Sim.) in Table A.5. The best results based on *t*-test for each problem definition are in bold.

Table A.1

Numerical results with *k*-NN (*F1*-score, „Speech Cycle“ corpus, all terms)

Term weighting method	Data configuration 1		Data configuration 2		Data configuration 3	
	***F1*-score**	***k***	***F1*-score**	***K***	***F1*-score**	***k***
IDF	0.855 (5-7)	4.3	0.735 (6-7)	2.6	0.631 (7)	7.4
GR	0.852 (5-7)	3.2	0.737 (6-7)	2.1	0.646 (6)	7.9
CW	**0.870 (1-3)**	2.8	0.769 (4)	2.8	0.704 (3-4)	5.5
TM2	0.865 (4)	5.9	**0.783 (1-3)**	2.3	**0.713 (1-2)**	4.5
RF	0.855 (5-7)	5.7	0.758 (5)	4.3	0.692 (5)	8.1
TRR	**0.873 (1-3)**	4.4	**0.784 (1-3)**	3.3	**0.715 (1-2)**	5.5
NTW	**0.871 (1-3)**	3.5	**0.786 (1-3)**	1.6	0.709 (3-4)	4.9

Table A.2

Numerical results with *k*-NN (accuracy, „Speech Cycle“ corpus, all terms)

Term weighting method	Data configuration 1	Data configuration 2	Data configuration 3
IDF	0.953 (7)	0.766 (7)	0.679 (7)
GR	0.957 (6)	0.791 (6)	0.724 (6)
CW	0.962 (4)	0.818 (4)	0.774 (3-4)
TM2	**0.963 (1-3)**	**0.839 (1)**	**0.785 (1)**
RF	0.961 (5)	0.814 (5)	0.763 (5)
TRR	**0.964 (1-3)**	0.833 (2-3)	0.778 (2)
NTW	**0.964 (1-3)**	0.833 (2-3)	0.773 (3-4)

Table A.3

Numerical results with SVM-FLM (*F1*-score, „Speech Cycle“ corpus, all terms)

Term weighting method	Data configuration 1	Data configuration 2	Data configuration 3
IDF	**0.873 (1)**	**0.788 (1)**	**0.721 (1-2)**
GR	0.670 (7)	0.521 (7)	0.479 (7)
CW	0.835 (4)	0.727 (4)	0.679 (4)
TM2	0.734 (6)	0.617 (6)	0.563 (6)
RF	0.864 (2-3)	0.777 (2-3)	0.715 (3)
TRR	0.865 (2-3)	0.777 (2-3)	**0.721 (1-2)**
NTW	0.825 (5)	0.711 (5)	0.650 (5)

Table A.4

Numerical results with SVM-FLM (accuracy, „Speech Cycle“ corpus, all terms)

Term weighting method	Data configuration 1	Data configuration 2	Data configuration 3
IDF	**0.963 (1)**	**0.833 (1)**	**0.786 (1-3)**
GR	0.890 (7)	0.642 (7)	0.623 (7)
CW	0.954 (5)	0.790 (4)	0.760 (4)
TM2	0.945 (6)	0.749 (6)	0.711 (6)
RF	0.961 (2)	0.827 (2)	**0.788 (1-3)**
TRR	0.961 (3)	0.825 (3)	**0.787 (1-3)**
NTW	0.954 (4)	0.785 (5)	0.746 (5)

Table A.5

Numerical results with Rocchio classifier (*F1*-score, „Speech Cycle“ corpus, all terms)

Term weighting method	Data configuration 1			Data configuration 2			Data configuration 3		
	Taxi Cab	Euclid	Cos. Sim.	Taxi Cab	Euclid	Cos. Sim.	Taxi Cab	Euclid	Cos. Sim.
IDF	0.593	**0.762**	0.712	0.518	**0.661**	0.640	0.496	**0.625**	0.608
GR	0.605	0.647	0.668	0.520	0.528	0.556	0.501	0.501	0.524
CW	0.561	0.724	0.692	0.482	0.595	0.589	0.459	0.567	0.554
TM2	0.439	0.727	0.707	0.373	0.599	0.624	0.345	0.564	0.590
RF	0.588	0.721	0.706	0.501	0.611	0.605	0.478	0.582	0.569
TRR	0.644	0.745	0.702	0.561	0.618	0.612	0.534	0.584	0.582
NTW	0.604	0.741	0.718	0.530	0.610	0.629	0.507	0.576	0.596

A.2. Weight-based Term Filtering on the „Speech Cycle“ corpus

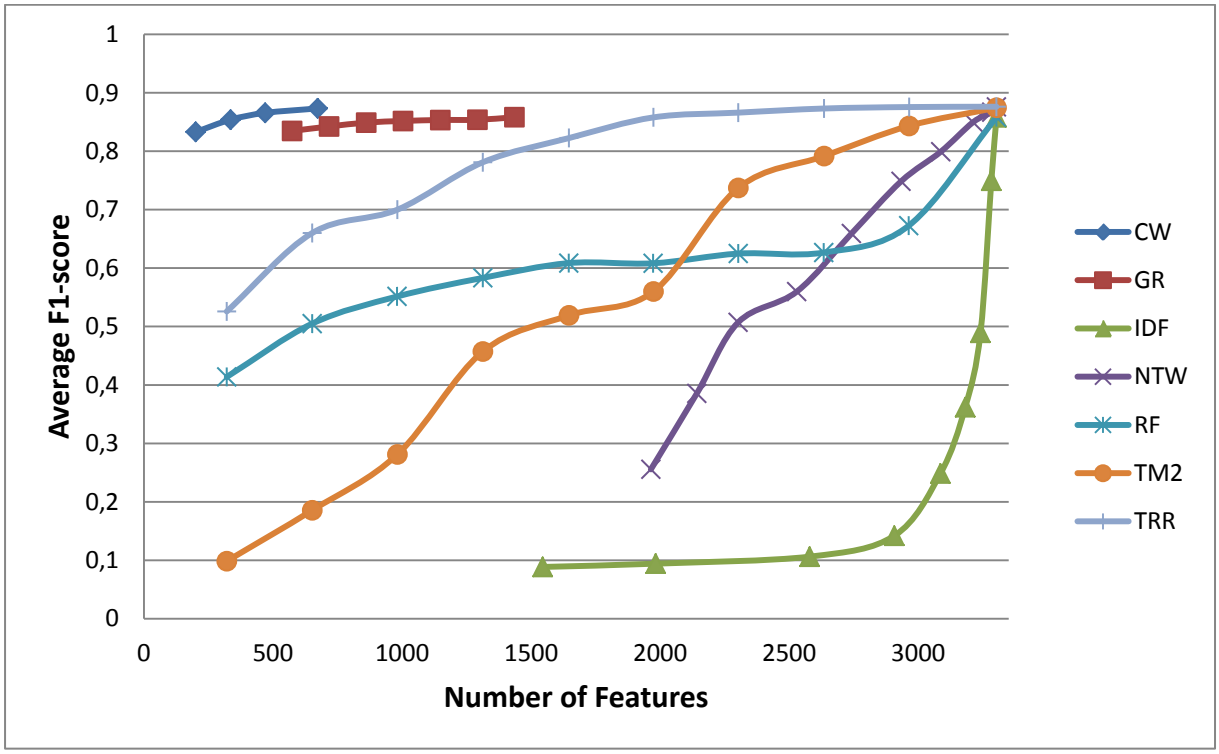

Figure A.1 - Feature selection with *k*-NN for data configuration 1

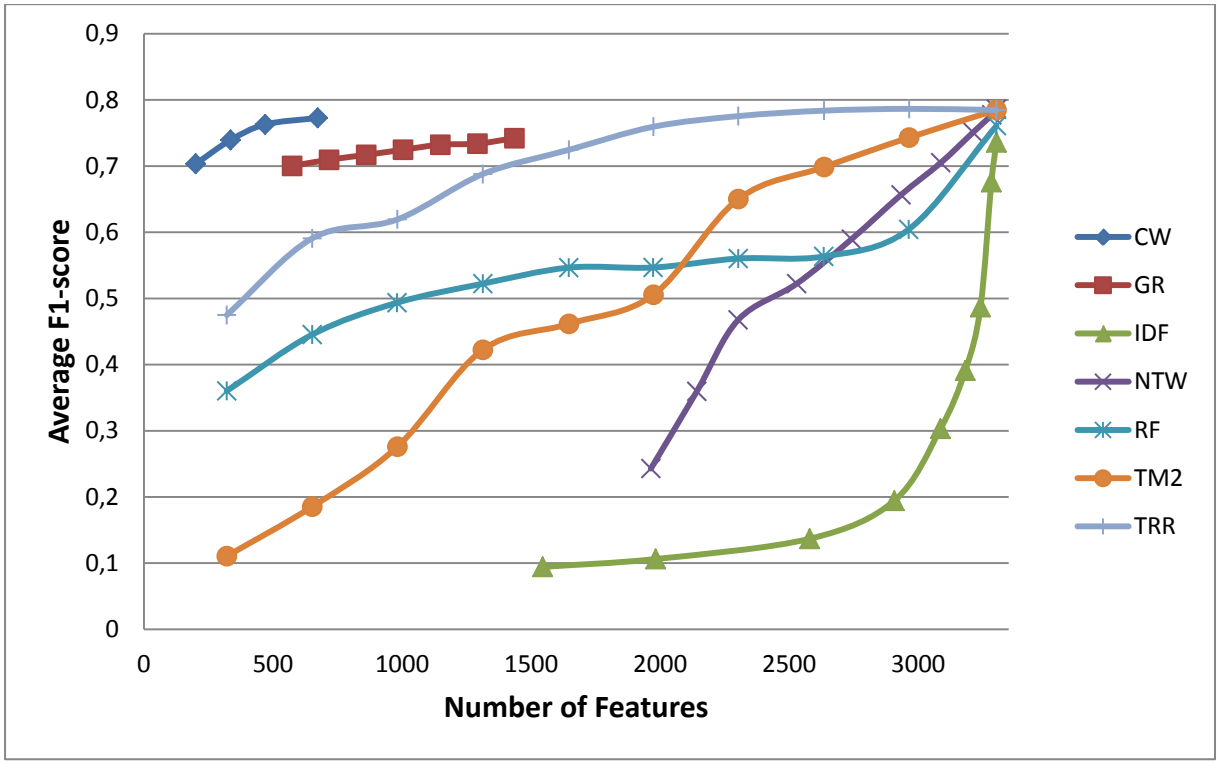

Figure A.2 - Feature selection with *k*-NN for data configuration 2

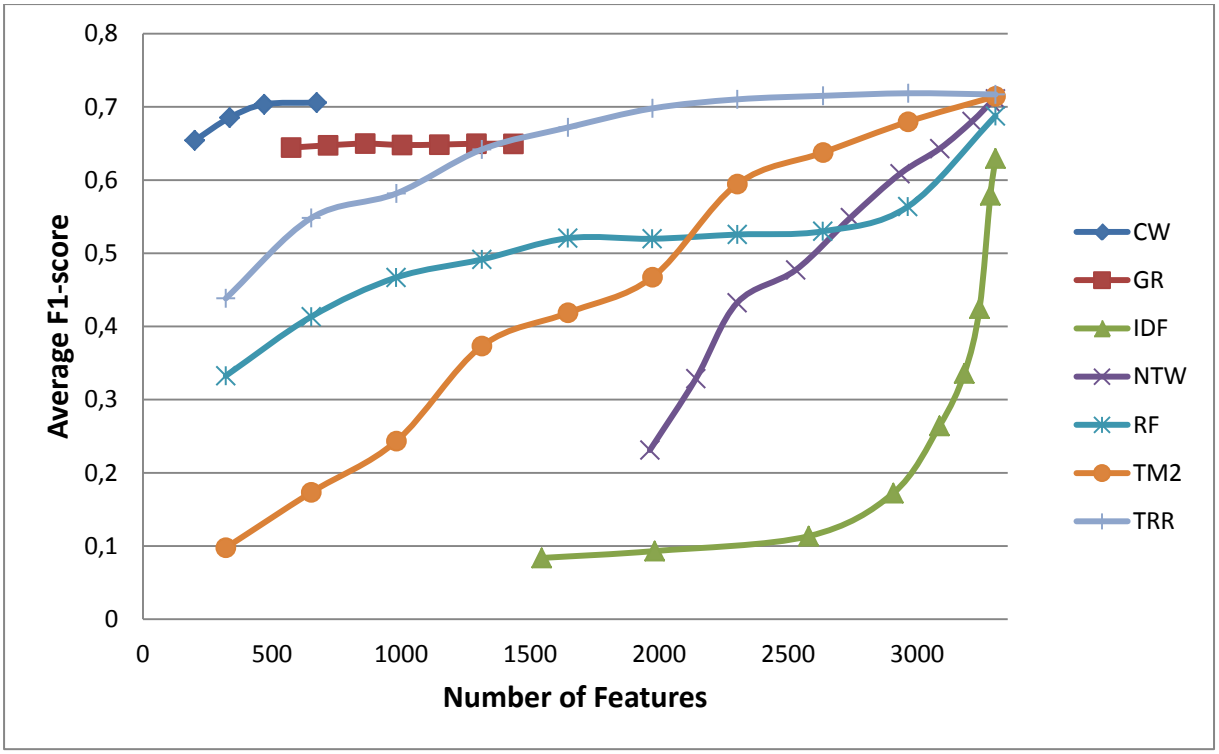

Figure A.3 - Feature selection with *k*-NN for data configuration 3

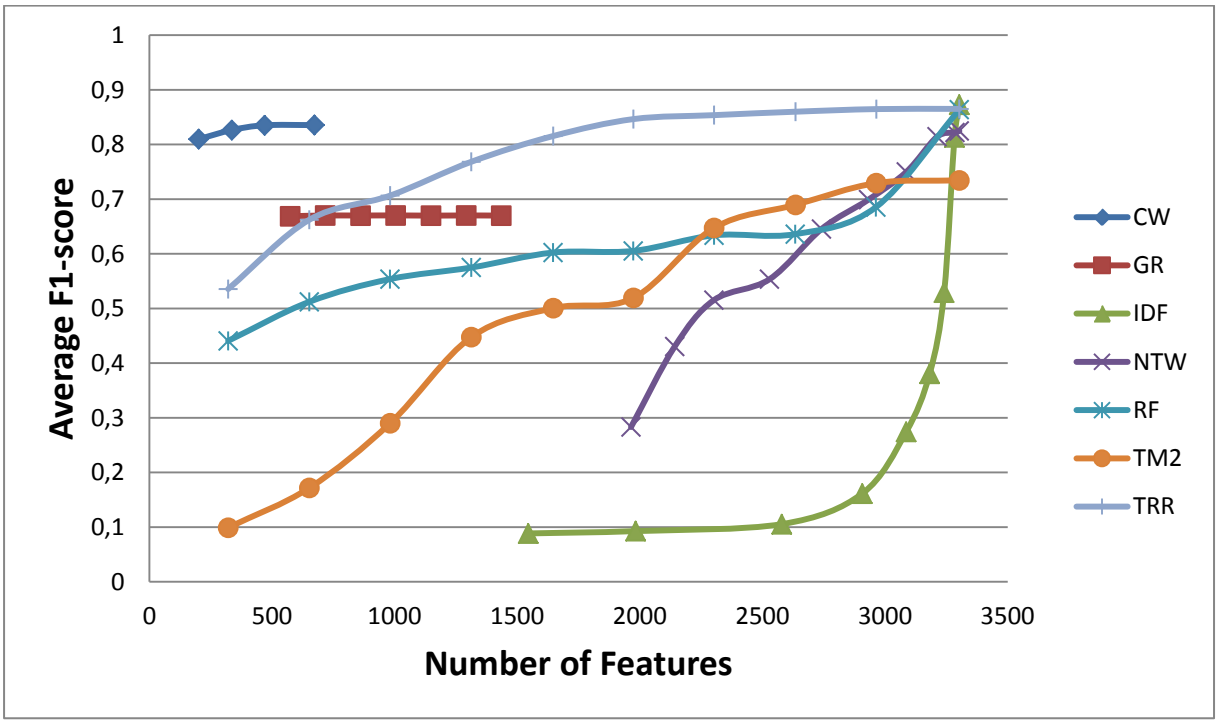

Figure A.4 - Feature selection with SVM-FLM for data configuration 1

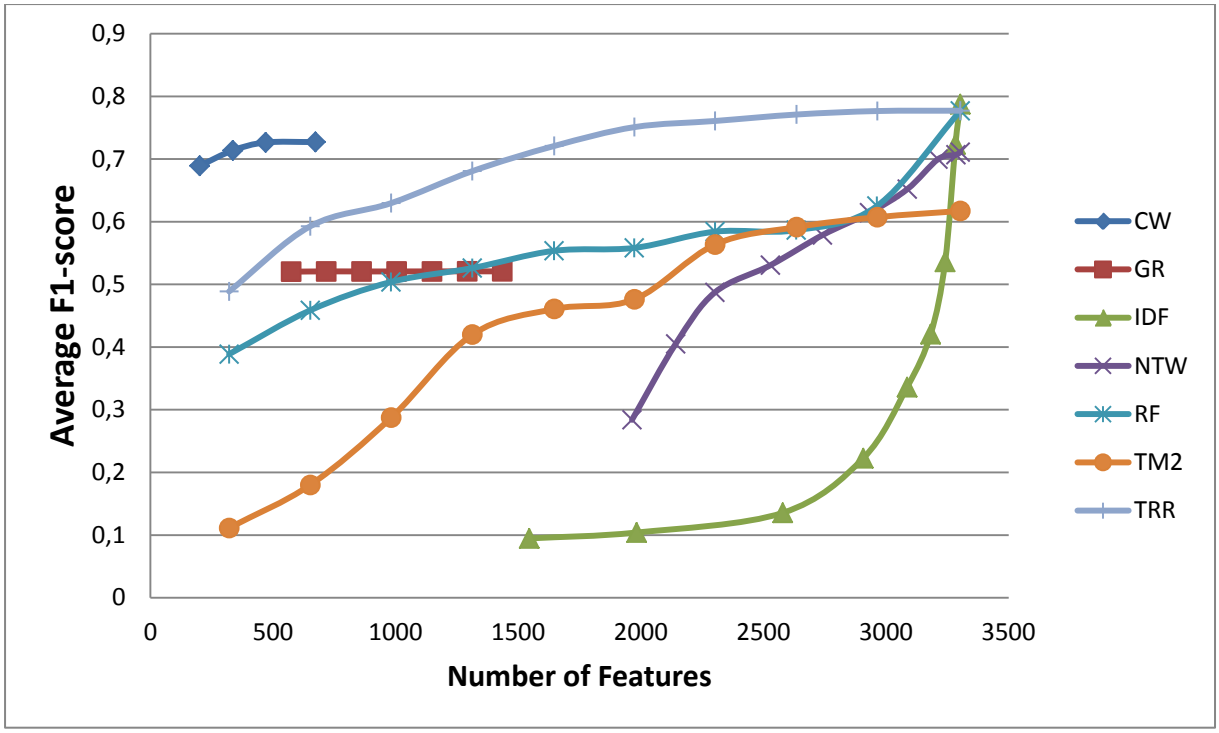

Figure A.5 - Feature selection with SVM-FLM for data configuration 2

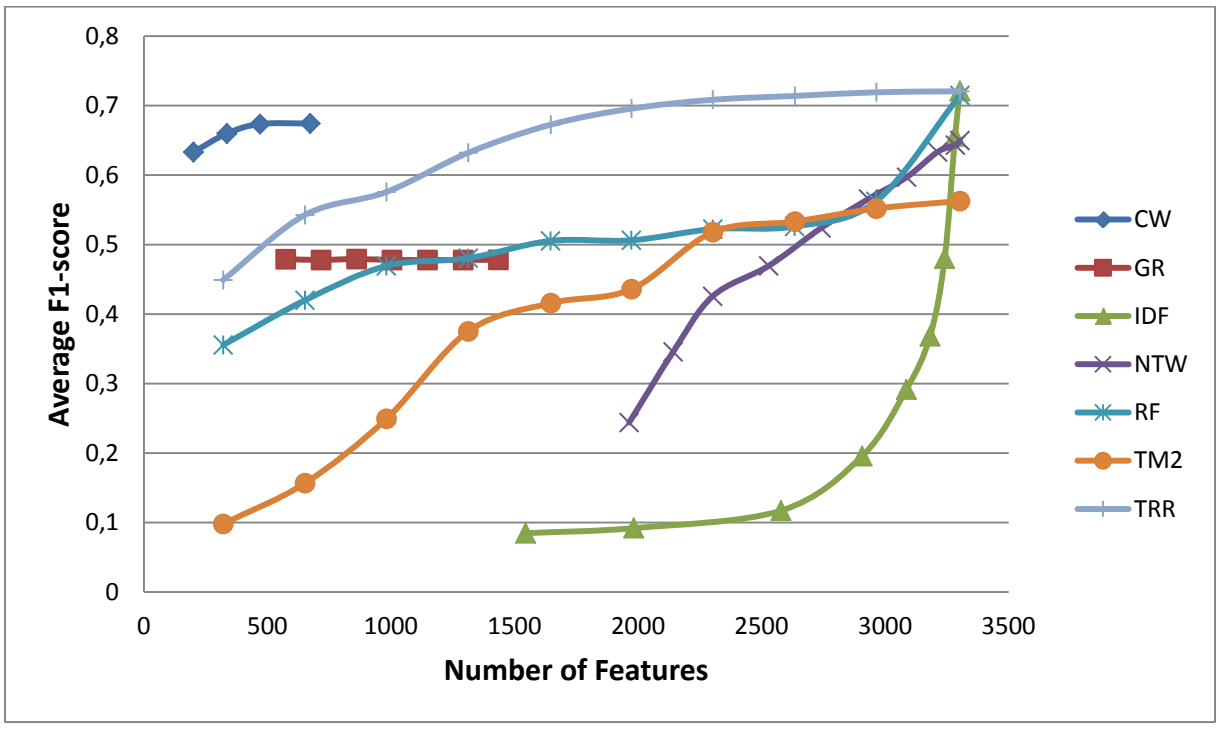

Figure A.6 - Feature selection with SVM-FLM for data configuration 3

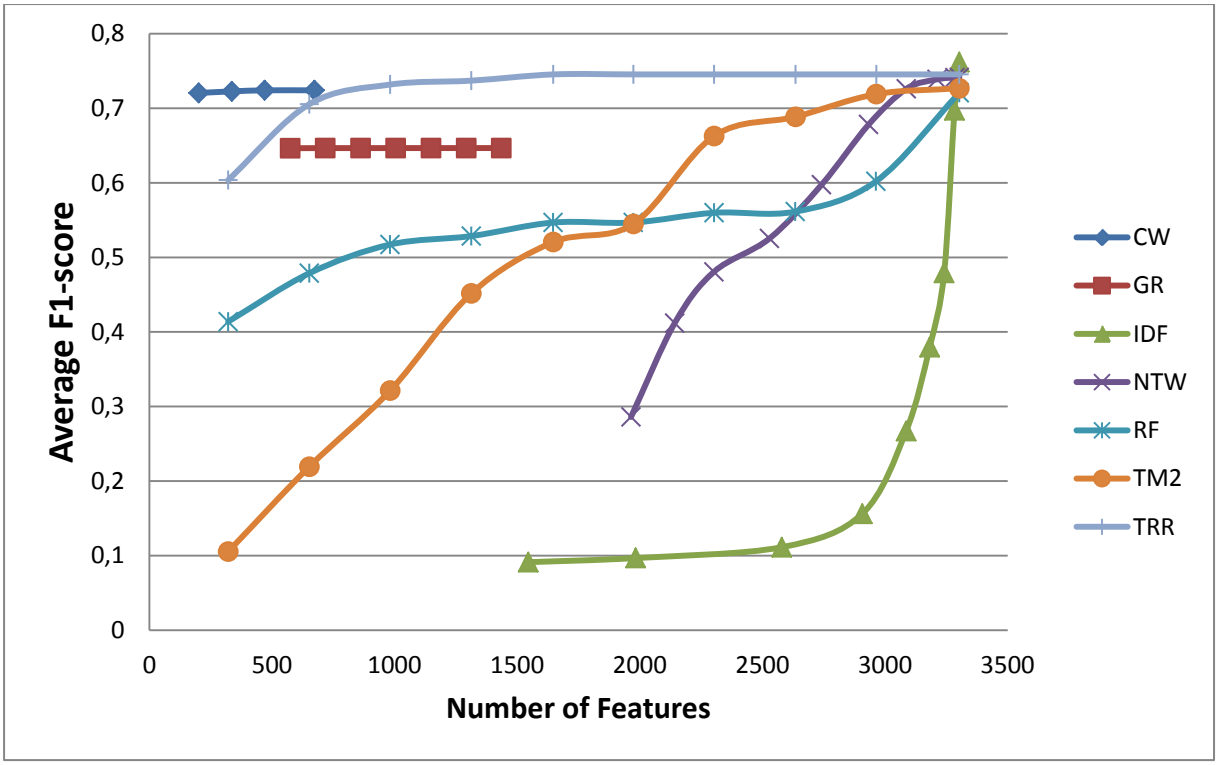

Figure A.7 - Feature selection with the Rocchio classifier for data configuration 1

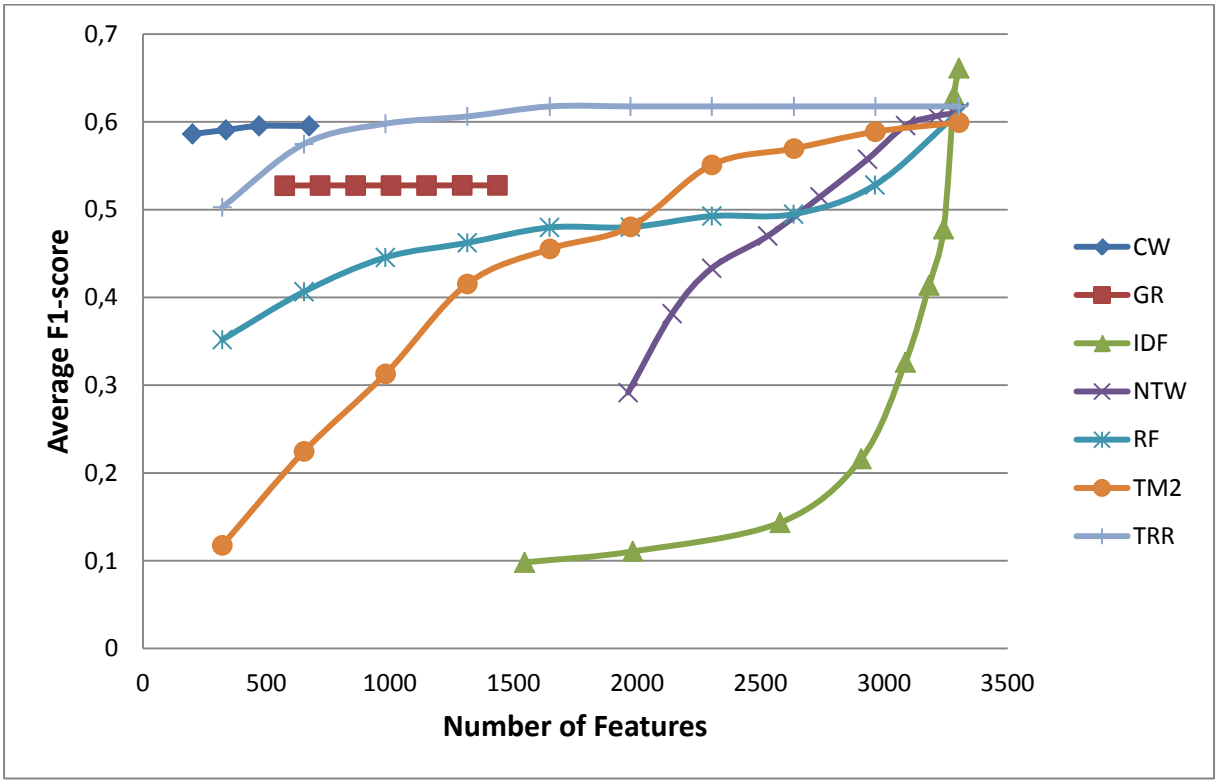

Figure A.8 - Feature selection with the Rocchio classifier for data configuration 2

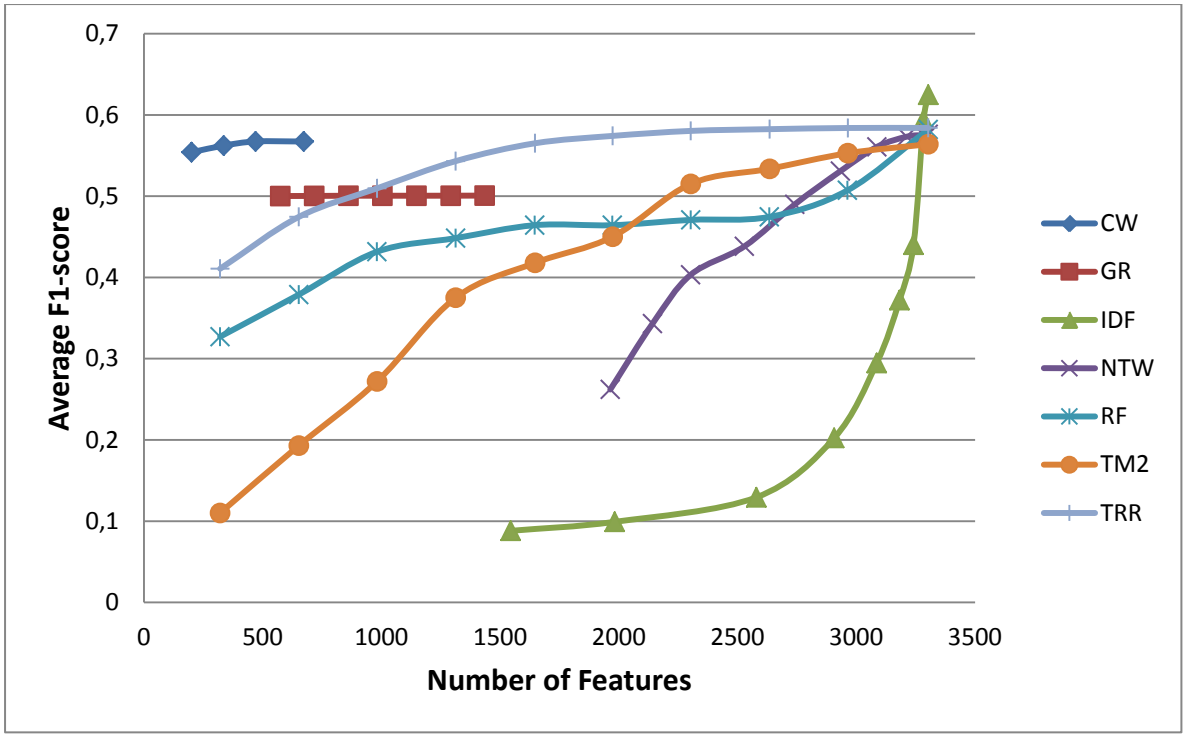

Figure A.9 - Feature selection with the Rocchio classifier for data configuration 3

A.3. Feature Transformation Based on Term Clustering on the „Speech Cycle“ corpus

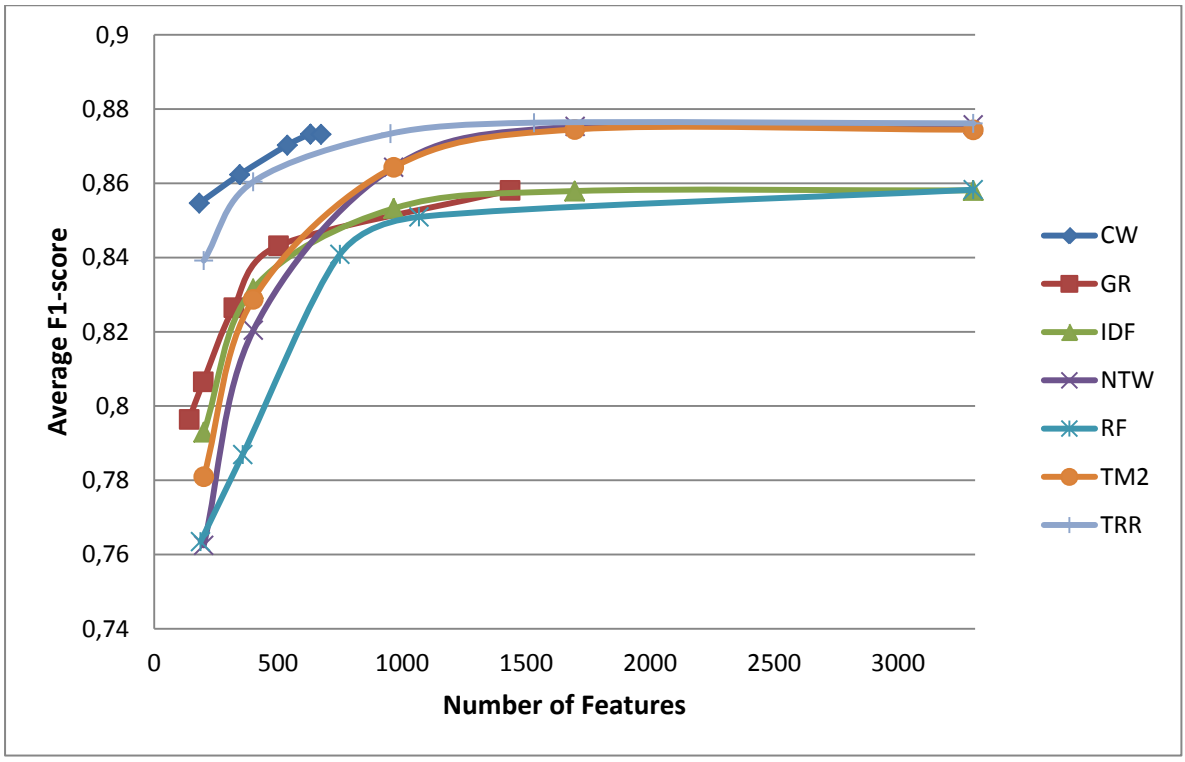

Figure A.10 - Feature transformation based on clustering with *k*-NN for data configuration 1

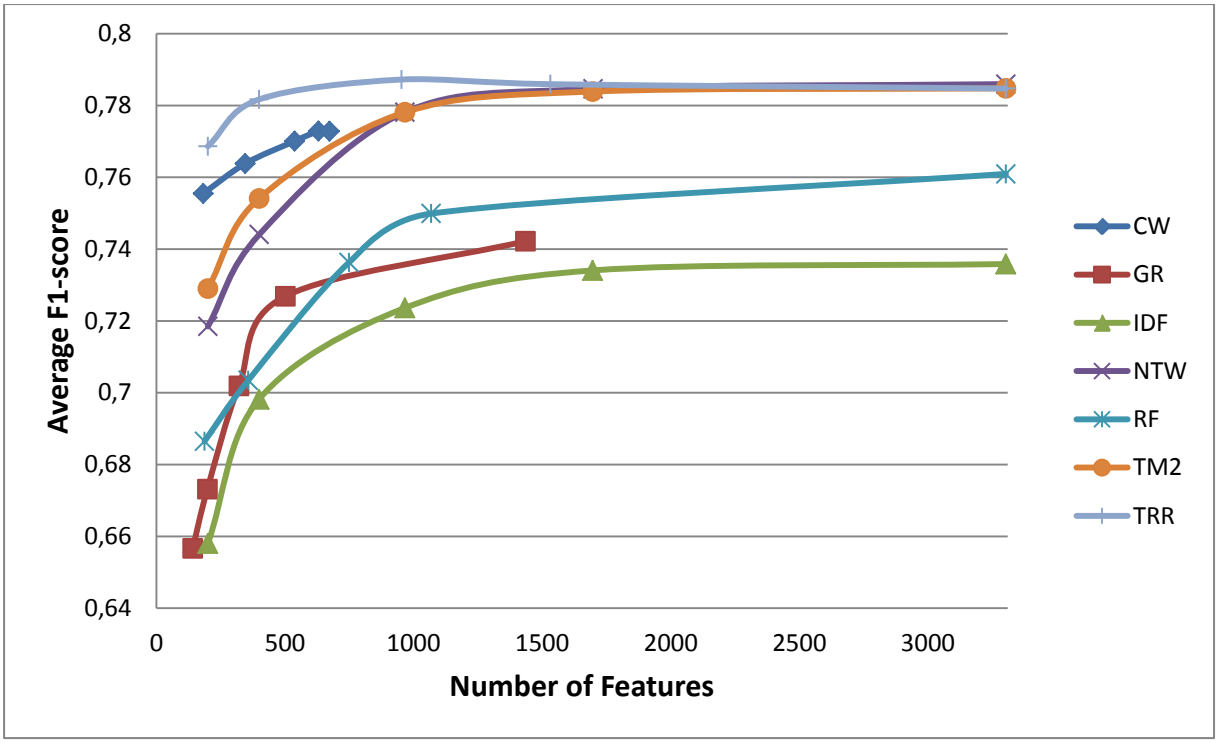

Figure A.11 - Feature transformation based on clustering with *k*-NN for data configuration 2

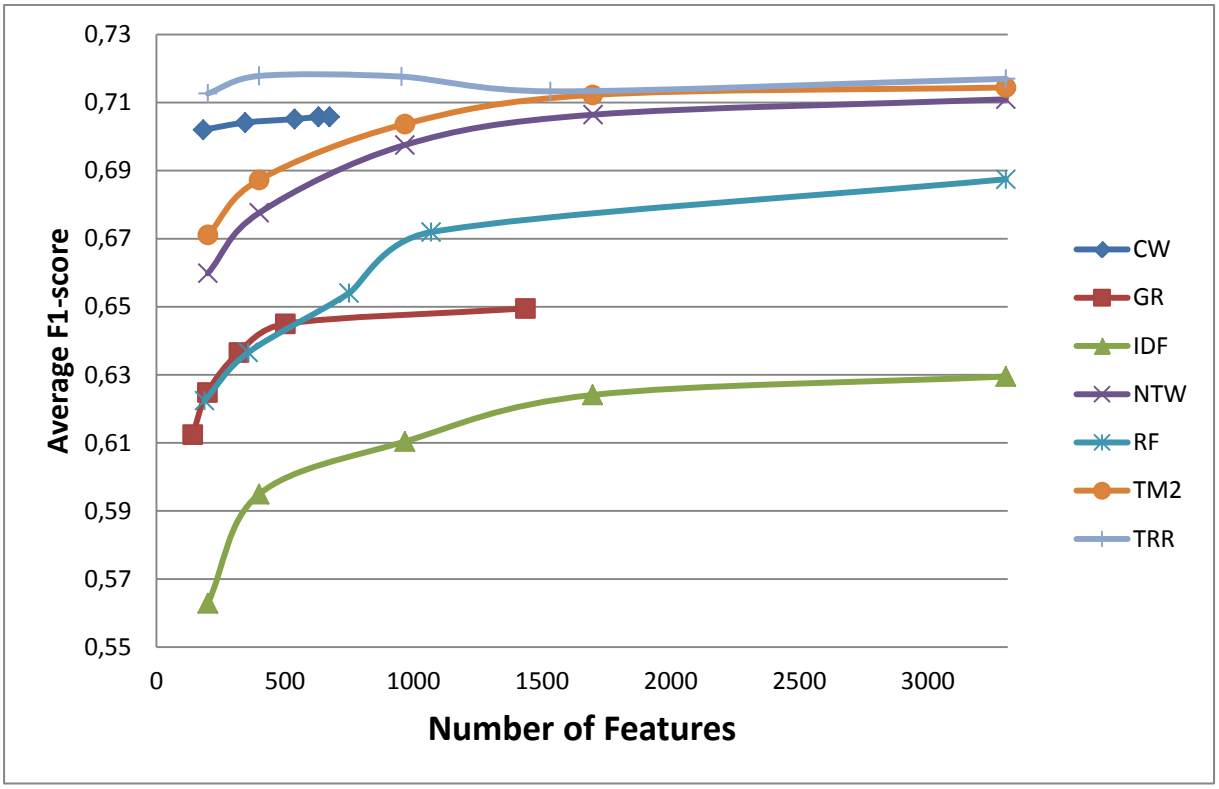

Figure A.12 - Feature transformation based on clustering with *k*-NN for data configuration 3

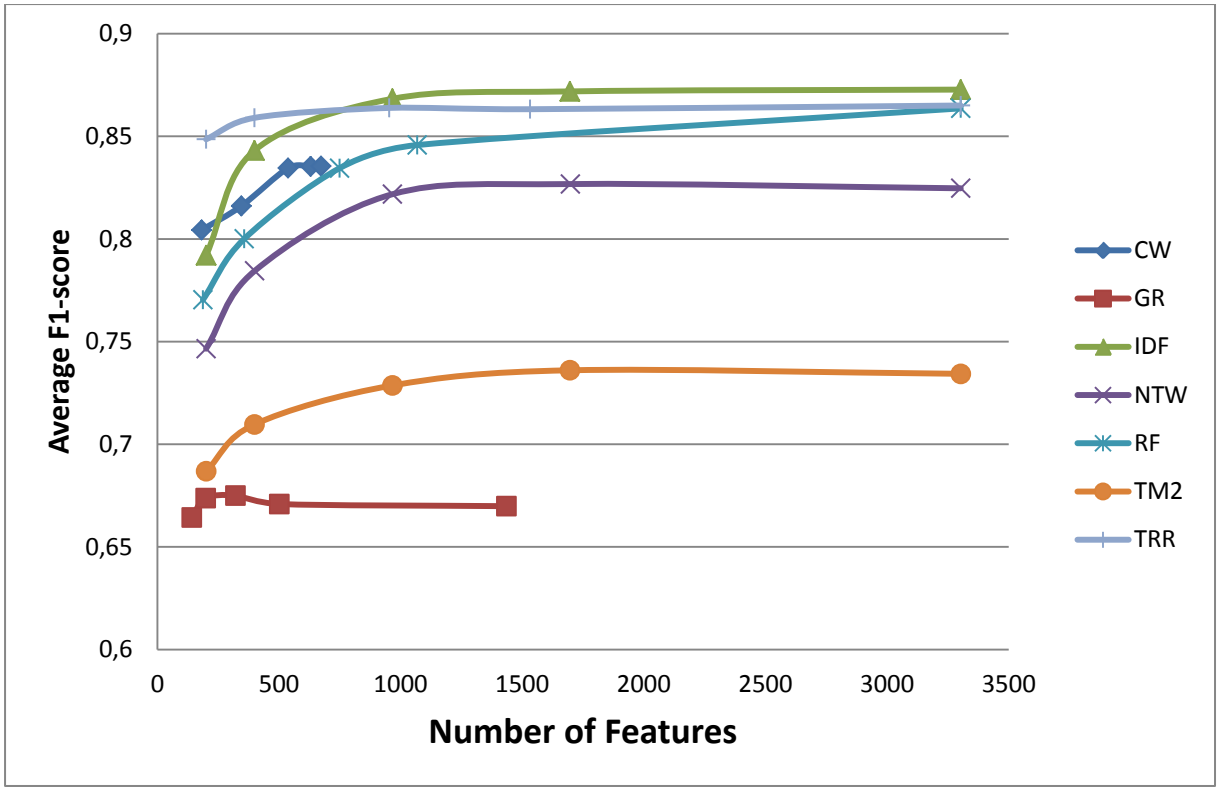

Figure A.13 - Feature transformation based on clustering with SVM-FLM for data configuration 1

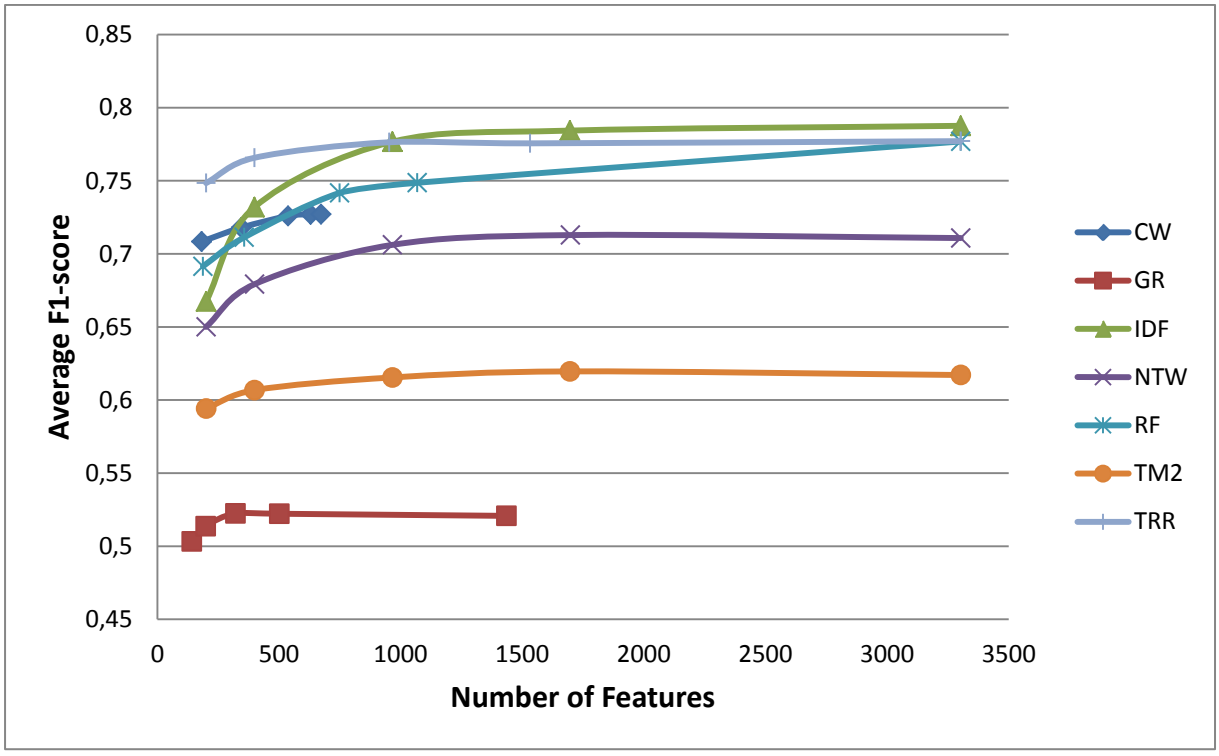

Figure A.14 - Feature transformation based on clustering with SVM-FLM for data configuration 2

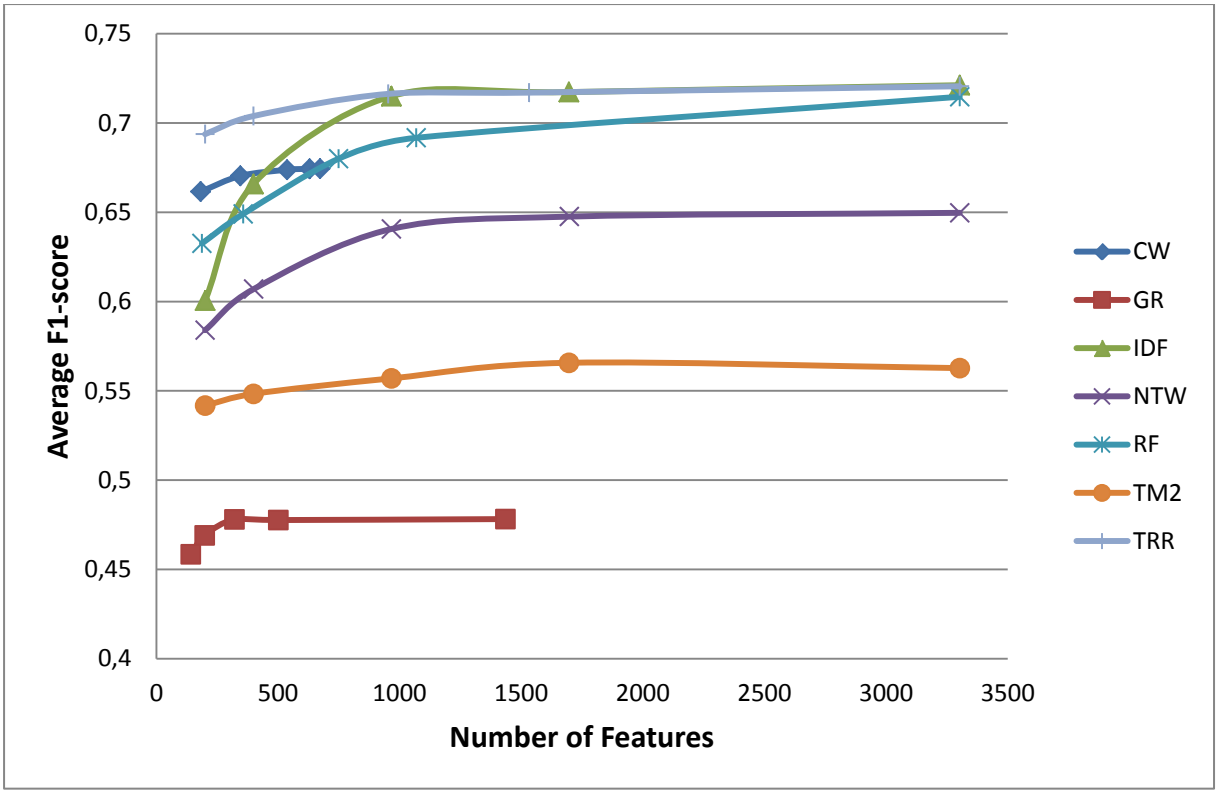

Figure A.15 - Feature transformation based on clustering with SVM-FLM for data configuration 3

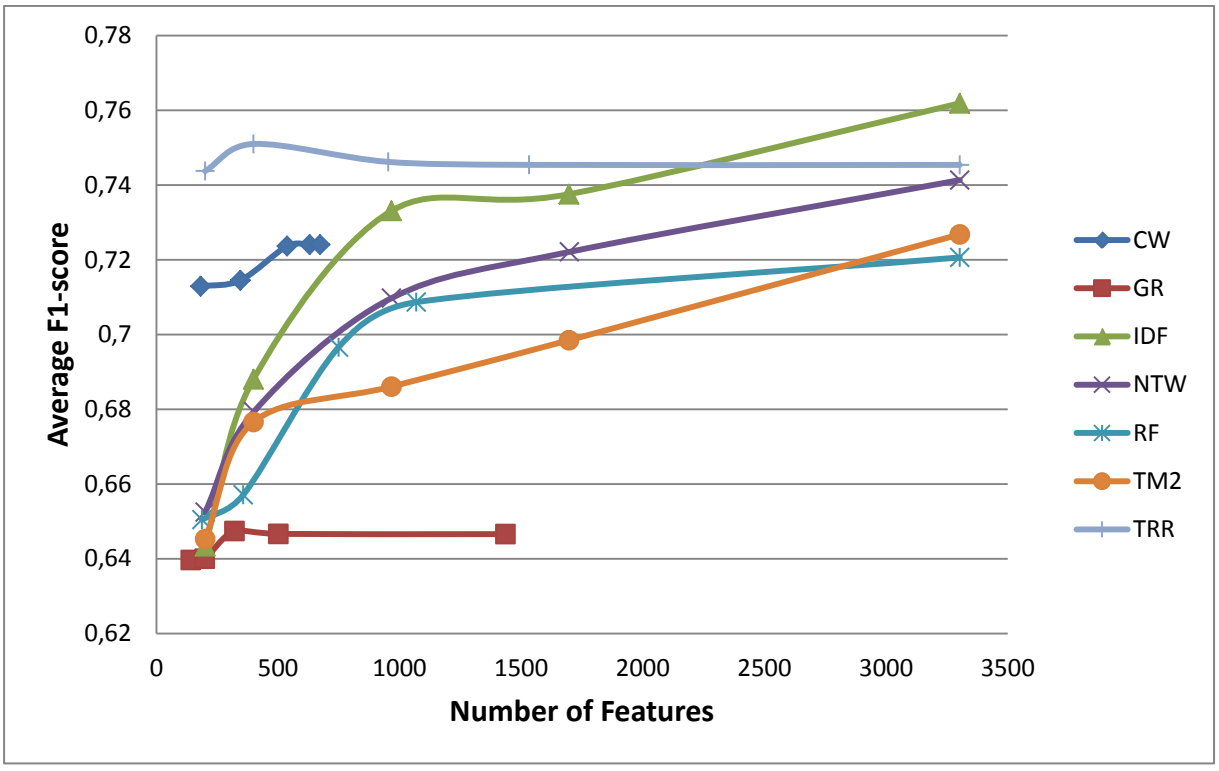

Figure A.16 - Feature transformation based on clustering with Rocchio classifier for data configuration 1

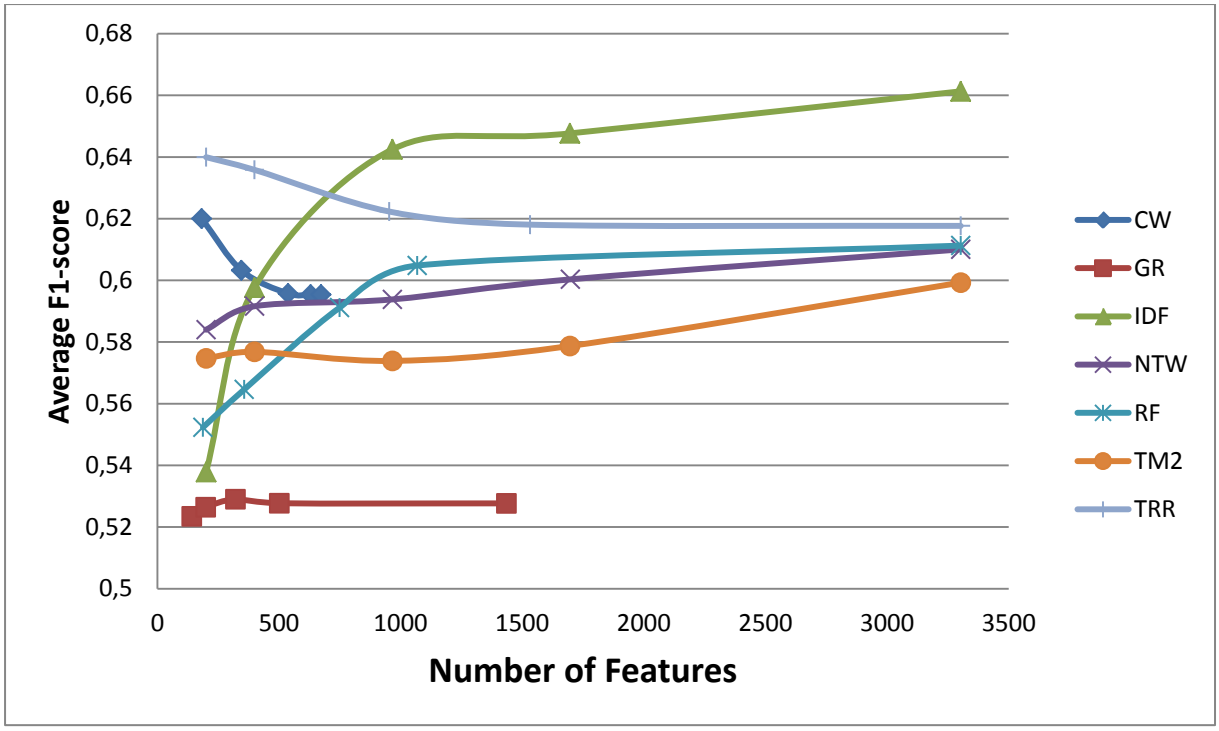

Figure A.17 - Feature transformation based on clustering with Rocchio classifier for data configuration 2

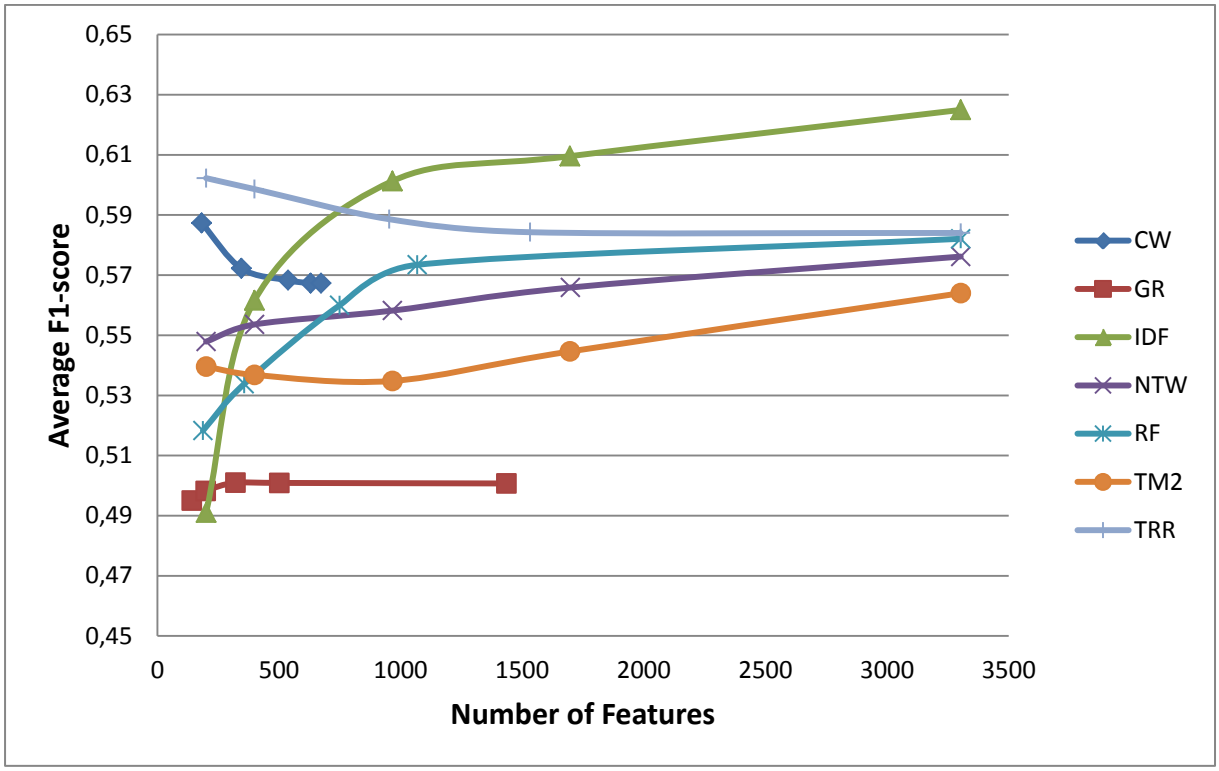

Figure A.18 - Feature transformation based on clustering with Rocchio classifier for data configuration 3

A.4. “Stop-word” filtering + Stemming on the „Speech Cycle“ corpus

The numerical results with “stop-word” filtering + stemming on the “Speech Cycle” corpus are presented in Tables A.6 – A.9. The structure of the tables is the same as for Tables A.1 – A.4.

Table A.6

Numerical results with *k*-NN (*F1*-score)

Term weighting method	Data configuration 1		Data configuration 2		Data configuration 3	
	***F1*-score**	***k***	***F1*-score**	***k***	***F1*-score**	***k***
IDF	0.777 (6)	12.2	0.689 (6-7)	8.8	0.636 (7)	8.9
GR	0.766 (7)	10.0	0.716 (6-7)	6.4	0.667 (6)	8.1
CW	0.784 (3-5)	11.2	0.734 (4-5)	6.7	0.695 (4-5)	7.0
TM2	0.784 (3-5)	11.5	**0.741 (1-2)**	5.9	**0.701 (1-3)**	6.4
RF	0.783 (3-5)	12.3	0.728 (4-5)	8.4	0.688 (4-5)	7.1
TRR	**0.793 (1)**	11.6	**0.743 (1-2)**	6.6	**0.704 (1-3)**	7.6
NTW	0.789 (2)	12.3	0.739 (3)	6.1	**0.702 (1-3)**	6.5

Table A.7

Numerical results with *k*-NN (accuracy)

Term weighting method	Data configuration 1	Data configuration 2	Data configuration 3
IDF	**0.917(1)**	0.733 (7)	0.691 (7)
GR	0.803 (7)	0.772 (6)	0.744 (6)
CW	0.860 (5-6)	0.784 (4-5)	0.766 (3-5)
TM2	0.861 (5-6)	**0.795 (1-2)**	0.773 (2)
RF	0.886 (2-4)	0.782 (4-5)	0.763 (3-5)
TRR	0.888 (2-4)	**0.796 (1-2)**	**0.774 (1)**
NTW	0.888 (2-4)	0.791 (3)	0.769 (3-5)

Table A.8

Numerical results with SVM-FLM (*F1*-score)

Term weighting method	Data configuration 1	Data configuration 2	Data configuration 3
IDF	**0.836 (1)**	**0.756 (1)**	**0.704 (1-2)**
GR	0.680 (7)	0.556 (7)	0.521 (7)
CW	0.749 (5)	0.717 (4)	0.675 (4)
TM2	0.720 (6)	0.631 (6)	0.586 (6)
RF	0.819 (3)	0.746 (3)	0.700 (3)
TRR	0.823 (2)	0.749 (2)	**0.701 (1-2)**
NTW	0.797 (4)	0.708 (5)	0.660 (5)

Table A.9

Numerical results with SVM-FLM (accuracy)

Term weighting method	Data configuration 1	Data configuration 2	Data configuration 3
IDF	**0.954 (1)**	**0.817 (1)**	0.778 (2-3)
GR	0.909 (7)	0.683 (7)	0.664 (7)
CW	0.943 (4-5)	0.789 (4)	0.760 (4)
TM2	0.934 (6)	0.762 (6)	0.730 (6)
RF	0.950 (3)	0.815 (2-3)	**0.781 (1)**
TRR	0.950 (2)	0.814 (2-3)	0.779 (2-3)
NTW	0.942 (4-5)	0.788 (5)	0.755 (5)

A.5. Novel Feature Transformation on the „Speech Cycle“ corpus

The numerical results with the novel feature transformation method on the “Speech Cycle” corpus are presented in Tables A.6 – A.9. The structure of the tables is the same as for Tables A.1 – A.4.

Table A.10

Numerical results with *k*-NN (*F1*-score)

Term weighting method	Data configuration 1		Data configuration 2		Data configuration 3	
	***F1*-score**	***k***	***F1*-score**	***k***	***F1*-score**	***k***
IDF	0.819 (7)	1.1	0.656 (7)	1.4	0.477 (7)	10.9
GR	0.841 (6)	1.1	0.703 (6)	1.8	0.596 (5)	10.1
CW	0.851 (2-3)	2.2	0.733 (2-3)	3.9	**0.653 (1-2)**	9.8
TM2	0.853 (2-3)	1.0	0.732 (2-3)	1.6	0.621 (4)	11.7
RF	0.849 (4)	1.1	0.710 (4-5)	7.2	0.636 (3)	12.0
TRR	**0.862 (1)**	2.5	**0.754 (4)**	3.3	**0.657 (1-2)**	8.9
NTW	0.844 (5)	1.1	0.716 (4-5)	2.4	0.584 (6)	12.6

Table A.11

Numerical results with *k*-NN (accuracy)

Term weighting method	Data configuration 1	Data configuration 2	Data configuration 3
IDF	0.947 (7)	0.724 (7)	0.599 (7)
GR	0.951 (6)	0.751 (6)	0.677 (6)
CW	0.957 (3)	0.792 (2-3)	**0.734 (1-2)**
TM2	0.959 (2)	0.795 (2-3)	0.718 (3-4)
RF	0.957 (4-5)	0.781 (4-5)	0.719 (3-4)
TRR	**0.960 (1)**	**0.808 (1)**	**0.735 (1-2)**
NTW	0.957 (4-5)	0.781 (4-5)	0.687 (5)

Table A.12

Numerical results with SVM-FLM (*F1*-score)

Term weighting method	Data configuration 1	Data configuration 2	Data configuration 3
IDF	0.544 (7)	0.452 (7)	0.396 (7)
GR	0.621 (4-6)	0.487 (6)	0.458 (6)
CW	0.747 (2-3)	0.645 (2)	0.597 (2)
TM2	0.618 (4-6)	0.549 (4-5)	0.501 (4)
RF	0.744 (2-3)	0.635 (3)	0.570 (3)
TRR	**0.792 (1)**	**0.668 (1)**	**0.607 (1)**
NTW	0.621 (4-6)	0.547 (4-5)	0.490 (5)

Table A.13

Numerical results with SVM-FLM (accuracy)

Term weighting method	Data configuration 1	Data configuration 2	Data configuration 3
IDF	0.869 (7)	0.604 (7)	0.560 (7)
GR	0.877 (6)	0.627 (6)	0.611 (6)
CW	0.924 (2-3)	0.727 (2)	0.699 (2)
TM2	0.900 (4)	0.700 (4)	0.665 (4)
RF	0.925 (2-3)	0.719 (3)	0.676 (3)
TRR	**0.937 (1)**	**0.734 (1)**	**0.696 (1)**
NTW	0.896 (5)	0.677 (5)	0.644 (5)

A.6. Results on the „Rafaeli“ corpus

Table A.14 contains results in terms of *F1*-score and accuracy with *k*-NN for three situations: without dimensionality reduction, with „stop-word“ filtering + stemming, with the novel feature transformation. The columns “*k*” contain the average optimal value of *k* after validation. The ranking of term weighting methods in terms of *F1*-score is illustrated in brackets. The lowest raw demonstrates results with *k*-NN and with collectives of term weighting methods based on majority vote. Table A.15 demonstrate the corresponding results with SVM-FLM. The best results are in bold.

Table A.14

Numerical results with *k*-NN

TWM	Without dimensionality reduction			"Stop-word" filtering + stemming			Novel feature transformation		
	***F1*-score**	**Accur.**	***k***	***F1*-score**	**Accur.**	***k***	***F1*-score**	**Accur.**	***k***
IDF	0.481(4)	0.490	2.4	0.461(4)	0.463	5.2	0.490(4)	0.525	3.8
GR	0.416(7)	0.442	3.6	0.395(7)	0.392	3.7	0.182(7)	0.273	5.8
CW	0.466(6)	0.487	9.7	0.399(6)	0.380	7.5	0.345(6)	0.356	7.4
TM2	0.475(5)	0.484	4.8	0.458(5)	0.481	4.3	0.480(3)	0.501	3.9
RF	**0.499(1)**	**0.507**	4.9	0.496(2)	**0.513**	5.7	**0.547(1)**	**0.576**	7.2
TRR	0.489(3)	0.487	4.7	**0.510(1)**	0.496	7.2	0.527(2)	0.549	6.6
NTW	0.497(2)	0.487	2.7	0.483(3)	0.493	5.2	0.471(5)	0.496	5.6
Coll.	0.534	**0.543**	-	0.510	0.519	-	0.513	0.549	-

Table A.15

Numerical results with SVM-FLM without collectives of term weighting methods

TWM	Without dimensionality reduction		"Stop-word" filtering + stemming		Novel feature transformation	
	***F1*-score**	**Accuracy**	***F1*-score**	**Accuracy**	***F1*-score**	**Accuracy**
IDF	**0.584 (1)**	**0.614**	**0.531 (1)**	0.546	0.517 (2)	0.546
GR	0.478 (4)	0.555	0.378 (6)	0.374	0.211 (7)	0.395
CW	0.346 (7)	0.386	0.300 (7)	0.350	0.224 (6)	0.323
TM2	0.404 (6)	0.424	0.452 (5)	0.463	0.487 (4)	0.496
RF	0.575 (2)	**0.614**	0.531 (2)	**0.570**	**0.555 (1)**	**0.582**
TRR	0.548 (3)	0.593	0.512 (3)	0.540	0.506 (3)	0.549
NTW	0.464 (5)	0.478	0.471 (4)	0.499	0.481 (5)	0.499
Coll.	0.554	0.561	0.489	0.516	0.535	0.558