Sebastian Schmidt, Steffen Schnitzer, Christoph Rensing: *Domain-Independent Sentence Type Classification: Examining the Scnearios of Scientific Abstracts and Scrum Protocols*. In: Stefanie Lindstaedt, Michael Granitzer, Harald Sack: Proceedings of 14th International Conference on Knowledge Technologies and Data-driven Business (i-KNOW '14), September 2014. ISBN 978-1-4503-2769-5.

Domain-Independent Sentence Type Classification: Examining the Scenarios of Scientific Abstracts and Scrum Protocols

Sebastian Schmidt
Multimedia Communications Lab
Technische Universität Darmstadt
Germany
schmidt@kom.tu-darmstadt.de

Steffen Schnitzer
Multimedia Communications Lab
Technische Universität Darmstadt
Germany
steffen.schnitzer@kom.tu-darmstadt.de

Christoph Rensing
Multimedia Communications Lab
Technische Universität Darmstadt
Germany
rensing@kom.tu-darmstadt.de

ABSTRACT

The amount of available textual information in everybody's daily environment is increasing steadily. To satisfy a user's information needs, the user has to examine numerous documents until the required information has been found. Additionally, the relevant information is often contained in only short sections of the considered documents. This leads to a high amount of irrelevant text the user has to read what could be solved by filtering relevant information within textual documents automatically. In this article we present our findings on the classification of sentences according to the type of information contained. Our evaluation has been conducted on documents from the field of abstracts of scientific publications and protocols of Scrum retrospective meetings. The results show the feasibility of our approach for finding a higher percentage of relevant information within textual documents and hence reducing the information overload for the users.

1. INTRODUCTION

Nowadays, almost everybody is flooded with information in form of textual documents. All these documents can comprise potentially important information for the user but without a good organization of the documents, the user can easily be overburdened with the task of finding relevant information. While this problem is often described as "information overload" others have termed it as "organisation underload". According to this understanding the main problem is not the sheer amount of information but the lack of tools and structures to cope with it.

Text retrieval methods can help identifying potentially relevant documents according to an information need but the user is still required to search manually for the relevant parts of the documents that comprise the required information. Further, a pure string-based search might yield wrong results since the relevancy of information is defined by its context. Pre-filtering relevant sections of a retrieved document reduces the amount of information to be processed by the user. We apply this pre-filtering based on a classification of sentences classifying the kind of information contained. We can then filter out irrelevant information by taking the type of sentence into account. In the following we will illustrate that this approach is applicable in a number of different settings:

Professional settings In most professions, work is too complex nowadays to be handled by a single workers without exchanging knowledge with other individuals or groups. This communication can be found in several different manifestations. Hence, most workers are confronted with various kinds of documents in their working environments. These can be formal documents like documentations, regulations, legal documents or documents of a more informal nature like emails. In most of these documents only certain parts are of relevance for the workers. For example, with a given number of meeting protocols, a worker might only be interested in the decisions which have been documented within these protocols. Another example can be the problems which come up when an individual searches for a new professional position. Besides other information, job offers often contain a company description, a number of prerequisites for filling the position and a list of tasks which describe the position to be filled. All these sections might contain similar descriptions

but the actual information varies depending on the context in which the information can be found. A simple example sentence classification for similar job offer descriptions could be:

- "Our company employs Java developers." (company description)
- "The applicant has to be a Java developer." (prerequisite)
- "You will develop Java applications." (task description)

Another critical example for this scenario could be a medical company placing a job offer for a software developer. Searching the whole document for medical terms would find matches in the company description (but not the task description) and hence yield a false positive for somebody searching a job in the medical sector.

Research settings Both in industrial and academic research settings, researchers have to cope with an enormous number of textual documents in order to acquire knowledge. Besides the textual work as described in the previous paragraph, a huge amount of text comes with the research articles the researcher has to handle for acquiring knowledge and comparing the own work with the work of other researchers. The two main challenges are the identification of relevant research articles and the extraction of the information required. Applying a pure string-based search on the complete text of the documents might yield unintentional results. For example when searching for other research articles that apply a certain method, one will also find these that refer to articles which apply this method because they mention them in a "related work" context.

Educational settings Besides traditional text books for learning, the amount of on-line learning resources is increasing as well. The abstracts of such learning resources often state educational objectives but also pre-requirements for the particular resource. Depending on the learners, they might search for similar phrases but only within one of those two different contexts. While both context sections might contain similar phrases (e.g. "Java Hello World" or "Simple MySQL statements"), a learner is only interested in resources which include such phrases in either the educational objectives or the pre-requirements (depending if they want to learn about this or already know about it and want to continue learning on an advanced topic).

In all these settings it would be beneficial to apply a tool that filters the relevant parts of a document instead of searching in a complete document. Within this paper we examine the usage of sentence classification for the identification of types of sentences in order to provide such a pre-filter for the users. To this aim, we take a look into two different concrete scenarios and evaluate the feasibility of our approach which is designed in a domain-independent fashion.

1.1 Outline

After the given introduction, which motivates our work, the following section gives a brief overview on related work. Section 3 presents the application scenarios of focus and the acquired corpora for these scenarios which are used for evaluation purpose. Section 4 presents our solution and gives insight into the machine learning features we use for sentence classification. The results of our evaluation are presented in Section 5 before the paper is concluded and an outlook on future work is given (Section 6).

2. RELATED WORK

Most existing work tackling the problem of information overload, try to solve this issue on document level by either either filtering irrelevant documents totally out or by improving the document retrieval process of relevant documents by semantic annotations of the complete documents.

Text analysis is an established field of research for quite a while now with applications in almost any domain. Noticeable is the wide range in terms of scope of the analysis. On the one end, there is the coarse grained classification of textual documents or even collections of textual documents handled with techniques such as *text classification* or *text retrieval*. The goal of these approaches is to assign to a complete collection of documents or to a single document one or multiple label(s) or to find the document(s) which might help the user to fulfil his/her information need. Finding the required information within these documents is to be done manually by the user. The precision for these approaches is rather low. On the other end, there are fields such as automated *question answering* systems which aim at finding an answer to a question composed in natural language or *named entity recognition* which aims to identify all named entities (of a restricted type) within a text. These approaches cater mainly to the identification of a single word/token or a short word/token sequence. These approaches often struggle in terms of recall since a very precise formulation of the query is required.

The work we present in this paper aims at the classification of sub-units of complete texts which are in our case sentences. Existing work on sentence classification mainly focuses on single scenarios. One of the fields examined frequently are scientific articles. Daniel et al. [2] identified word patterns for the classification of sentences into information type categories, such as "could be used in" for application related sentences or "remains unknown" for problem related sentences. An SVM-based approach for sentence classification in the bio-medical domain has been presented in [5]. Liakata et al. [9] show the feasibility of the classification for automated text summarization. Besides the focus on classification, investigations on sentence annotation schemes is focus of several work, where some focus on abstracts of scientific articles (e.g. [5]) while others focus on complete papers (e.g. [15]). Remotely related work considers discourse and argumentation structures to identify sentiments on sub-sentence level in small documents [10] which could be used as features for scientific abstracts. However, the scenarios examined in this article and many other scenarios do not contain discourse characteristics. Other examined scenarios for sentence classification are e.g. e-mail data ([8], [17]), bio-medical data bases [16], legal texts [3] and event classification in newspaper articles [11]. To conclude, we did not see work which examines the usage of a single approach within different scenarios.

3. EXAMINED SCENARIOS

We examined two scenarios that are relevant for our work in detail. These scenarios have been selected to have con-

trary properties. While the first scenario on *Abstracts of Scientific Articles* is characterised by complete sentences of a very formal and precise language, the second scenario on *Protocols of Scrum Retrospective Meetings* is characterized by very short sentences (often not even of correct grammar) or bullet point lists. Further, the first scenario originates in the research setting and the second scenario originates in the professional setting.

3.1 Abstracts of Scientific Articles

As mentioned in the introduction, researchers are coping almost daily with scientific articles. To get familiar with a topic it requires a lot of time and effort to find relevant literature and in particular, find the relevant information within these articles. Typical information needs of a researcher are:

- Which other articles face a particular problem or have the same motivation as my work?
- Which other articles use a particular approach/method?
- Which authors work in related fields as I do?
- Which approach performs best for a specific problem?

To find an answer to these questions the examination of the abstracts of the article can be quite helpful since the abstracts contain a condensed version of the content of the whole paper. Further, only particular information within the abstracts is relevant to answer a particular question. To answer the first question it is only required to examine sentences which present the motivation of a research work whereas for the last question only sentences that present results are required to be examined. Based on this observation we aim at the classification of sentences in scientific articles according to the information they contain.

3.1.1 Corpora and Annotation Schemes

For the work described in this article we use two different corpora of scientific abstracts. The corpus *MM* has been created by collecting 81 abstracts of scientific articles out of the field of multimedia research publications. In total, the corpus consists of 628 sentences. We developed an annotation scheme consisting of eight different class labels which allow to capture the multiplicity of information in scientific abstracts. The annotation scheme together with the description of the single classes can be seen in Table 1. Each sentence has been annotated by three annotators. The annotators were asked to assign each sentence a label from the provided set. In case of ambiguities where a single sentence belongs to multiple categories, the annotators were allowed to assign multiple categories to a single sentence. E.g. the sentence *"While there is evidence, both scientific and anecdotal, that olfactory cues help users in information recall tasks, there is a lack of work when the targeted information is one contained in a multimedia presentation, which is precisely the focus of this article."* can be both assigned to the classes *Related Work* and *Motivation*. The annotators were explicitly asked to use a single class label whenever possible. On average, the annotators used 1.11 labels per sentence. Table 2 shows the detailed result of our annotation study. While columns 1 to 3 present the usage frequency of the single class labels per person, column 4 shows the number of sentences with a total agreement and the last column shows the number of sentences with a $\frac{2}{3}$-majority. One can see that there are different understandings of the single labels. While a total agreement was achieved for 40.76%, a $\frac{2}{3}$-majority was achieved for 86.94% of the sentences . This high result of the majority-voting made us decide to use the majority-based class labels. Hence, our cleaned corpus consists of 546 sentences with one single class label.

Label	Annotator			Agreement	Majority
	1	2	3		
Summary	74	69	141	51	73
Motivation	163	191	140	100	165
Goals	22	11	22	2	11
Method	95	34	125	0	25
Related Work	82	41	82	20	64
Solution	203	239	48	27	162
Results	99	96	76	56	87
Conclusion	25	9	17	0	8
In total (628)	763	690	651	256	595

Table 2: Frequency of the Class Labels for the Corpus *MM*

As a second corpus we use the data set by Guo et al. [5]. The data set consists of 1,000 abstracts with 8,633 sentences in total. The abstracts are coming from the biomedical domain focusing on cancer risk assessment. Each sentence is annotated with one out of seven class label. The set is defined as follows: {*Background*, *Objective*, *Method*, *Result*, *Conclusion*, *Related Work*, *Future Work*}. In a pre-experiment with three annotators Guo et al. have discovered an inter-annotator agreement of $\kappa = 0.85$, due to this high value, they decided to rely on the annotations of a single annotator for the final corpus.

3.2 Protocols of Scrum Retrospective Meetings

In the previous decade, software development has seen a change towards agile development processes. Scrum is the probably most common agile software development process in the industry. While agile development has been initially mainly successful in small and medium-sized companies, nowadays also many big software companies apply this principle. One of the important artifacts of Scrum is the regular retrospective meeting. The goal of this meeting is to understand the process during the last Sprint[1] by identifying good, bad and improvable aspects of the process during the passed Sprint. This is often narrowed down by asking the three questions *"What went well?"*, *"What went wrong?"* and *"What could be improved?"* to the Scrum team members. The answers are commonly written down into a protocol, so that the process can be improved on the long run.

3.2.1 Corpora and Annotation Scheme

We got a corpus consisting of 139 Scrum retrospective protocols provided from a major software company. Each protocol is divided into several varying paragraphs but in most cases answers to the above mentioned questions can be

[1] A *Sprint* is the cycle in which the actual software development is happening. Sprints are of fixed length.

Label	Description
Summary	A complete summary of the work in one sentence
Motivation	What is the motivation for the described work? Why is it relevant? What is the challenge?
Goals	What is/was the goal of the work?
Method	Which approach has been chosen for facing the problem? (Technology, Steps during the design process, etc.)
Related Work	What has been done in this field before? Where did it succeed? Where did it fail?
Solution	How does the presented approach work? What is the main idea?
Results	How well does the approach perform?
Conclusion	Is the result satisfying? In which ways can it be used?

Table 1: Class Labels used for the Corpus *MM*

Label	*Scrum*	*Scrum_Subset*
What went well?	264	191
What went bad?	155	136
What could be improved?	234	115
In total	653	442

Table 3: Frequency of the Class Labels for the Corpora *Scrum* and *Scrum_Subset*

found. The questions are followed by a list of bullet points where each bullet point is an independent statement/answer for the respective question. In total, there were 653 sentences in the corpus. Within the corpus we realized some variations in the terminology of the three typical questions mentioned above. This happened in particular across the different Scrum teams where each team uses their own terminology. Based on this observation we clustered all answers/notes for the following sets of questions/headlines:

- **"What went well?"**
- **"What went wrong?"**, "What went bad?", "What did not work so well?", "What I did not like", "What did not go well?"
- **"What could be improved?"**, "What should be improved?", "What can we improve?", "What should we do differently?", "What should we change?", "Areas of Improvement", "Suggestions", "What should we start doing?"

This clustering resulted in the corpus *Scrum* with the class distribution as presented in the first columns of Table 3.

A further analysis of the data revealed that some of the sentences/notes cannot even by humans be classified into one of the categories. In the data we found examples like *"Timing"* or *"Collaboration with Peter Smith"*. These examples could be classified in any of these categories, depending if *Timing* or *Collaboration with Peter Smith* was considered as positive, negative or something to be improved. In order to understand how much this influences the results we created another corpus which is a subset of *Scrum* where we manually removed all sentences where an annotator was not able to decide for a class label. We refer to this corpus by *Scrum_Subset*. This corpus consists of 442 sentences in total. The distribution among the class labels is also presented in Table 3.

4. SOLUTION

We aim to solve the task of sentence classification with a supervised machine learning solution applying three classifiers with differing technologies. We apply a Bayesian classifier, a Decision Tree Classifier and a Support Vector Machine (SVM). Naive Bayes stands out by its training and classification speed, SVM has shown to be very successful when handling with textual data and tree-based classifiers can easily be interpreted by humans. In the following we present the groups of features used for classification. These features are designed to be domain-independent being not only restricted for the usage in the domains investigated in this article.

4.1 Feature Groups

We use a set of different features which can be applied to the single sentences. The features have been chosen to be generic and well suited for the application scenarios but also for scenarios other than the examined ones. The groups of features are explained in the following.

4.1.1 Content

Each word in the corpus is used as feature. In order to allow for different importance of the single words, we apply the *term frequency-inverse document frequency (tf-idf)* weighting scheme [14].

4.1.2 Sentiment

In order to incorporate the varying sentiment among different types of sentences, we check each word of a sentence for bearing an emotional connotation. For this, we use SentiWordNet [1], which provides a mapping from single words to positive and negative sentiment scores (within the interval [-1.0,1.0]). Three features are derived from these scores: one feature summing up all the positive scores for the words in a sentence, one feature summing up all the negative scores for the words in a sentence and one feature the total sum.

4.1.3 Negation

In addition to the sentiment features described above, the existence of negation words is checked. For this, each sentence is checked for the existence of the terms "not", "never" and "no". The feature described here contains the total number of these words in the sentence.

4.1.4 Tense

Since the type of information contained in a sentence seems to vary in terms of tense, we are apply the *Stanford Lexicalized Parser* [4] in order to determine the tense of the

sentences. One feature counts the past tense verbs and one feature counts the present tense verbs.

4.1.5 Tense Indicator

As in some cases, automated grammar parsers and taggers particularly fail in situations with incomplete sentences we use two additional features that count the existence of past tense indicators. On the one hand, this is the ending "ed" and on the other hand, these are the modal verbs "had", "was", "were", "been" and "got".

4.1.6 Adjectives

Adjectives are central to the meaning of a sentence and we therefore consider them as well. In order to detect them, we use again the tags created by the *Stanford Lexicalized Parser*.

4.1.7 Indicative Indicator

Certain types of information are characterized as a prompt to perform an action. We identify these sentences by the search for the terms "need", "should" and "must". The total frequency of these terms is hold by a single feature.

4.1.8 Personal Pronouns

Since the point of view from which a sentence is composed can give important insights on the type of information, we use personal pronouns as additional features.

4.1.9 Position

While some information tends to be at the beginning of a document, other tends to be placed in the middle or in the end. Since the length across the documents is usually not normalized, we make use of the relative position of a sentence within the document. We therefore divide the position of the sentence within the document by the total number of sentences in the respective documents resulting in a floating point value between 0.0 and 1.0.

4.1.10 Counts

The number of numeric tokens is also consulted as a feature. We assume that e.g. sentences from scientific abstracts that are of type "Result" contain a higher number of these tokens compared to other sentences. Based on the intuition that some type of information require more text to be described than others, we include the number of words within a sentence as another feature.

5. EVALUATION

We implemented the solution described in the previous section in Java, making use of the Weka framework [6]. After importing the corpora described in Section 3, we extracted the features out of the textual representation without any data-cleaning and stored the values in arff[2] files. In order to evaluate the feasibility of the single feature groups, we applied a single or all except one of the feature groups and for each of the resulting document representations we executed a separate evaluation on all data sets. For the scenario on *Abstracts of Scientific Publications*, we made use of all feature groups outlined in Section 4. For the scenario on *Protocols of Scrum Retrospective Meetings* we excluded the *Sentence Position* feature since the sentences in the provided corpus are ordered by class labels, which cannot be assumed in a realistic setting and hence the result would be biased.

We used three different classifiers from the Weka library. As a representative for the Support Vector Machines, we used the *SMO* class [12], as a Bayes classifier we used the *NaiveBayes* class [7] and finally as a decision tree classifier we used the *J48* classifier [13]. Each of these classifiers was used with its standard settings. In order to avoid a biased split into training and testing data the evaluation was carried out applying a 10-fold cross validation. For all evaluations we measured the average by class sized weighted F-Measure across all class labels for each corpus separately.

[2]arff: "Attribute relation format", the internal data representation used by Weka

	MM			*Guo*		
	SVM	**NB**	**J48**	**SVM**	**NB**	**J48**
All	.692	.690	.640	.798	.731	.739
All Except						
Content	.555	.492	.510	.666	.605	.648
Sentiment	.694	.710	.644	.800	.732	.739
Negation	.688	.693	.641	.799	.734	.741
Tense	.686	.699	.641	.798	.730	.737
Tense Indic.	.692	.697	.641	.799	.736	.738
Adjectives	.699	.692	.641	.799	.735	.738
Indic. Indic.	.689	.690	.641	.799	.731	.742
Pers. Pron.	.690	.690	.641	.798	.731	.739
Position	.634	.656	.576	.750	.670	.675
Counts	.692	.694	.639	.799	.741	.741

Table 4: Weighted F-Measures for the Scenario on Scientific Abstracts for all Features and for all Features except a single Feature Group

5.1 Evaluation Results

5.1.1 Abstracts of Scientific Articles

The results of our evaluation within the scenario of *Scientific Abstracts* can be obtained from Table 4 and Table 5. While the first one gives an overview on using all features and all features except one single group of features, the second table presents results for using only a single feature group. For both corpora, *MM* and *Guo*, it can be seen that the SVM outperforms the other classifiers. The results for the *Guo* corpus are more than 10% better compared to the *MM* corpus. Reasons for this will be discussed later in this article. Examining Table 4 in detail, one can see that the removal of the *Content* feature results in the largest drop in terms of F-Measure for all the classifiers and both corpora. Further, leaving out the *Position* of a sentence in an abstract also results in a major decrease of the F-Measure. The removal of the other feature groups has only a minor impact on the F-Measure. The best results across the corpora using *SVM* or *Naives Bayes* were obtained when including all features except for the feature group of *Adjectives* or *Sentiment*.

Examining the results for the usage of single features (see Table 5), one the same tendency can be observed: the best results for a single feature group are obtained using the *Content* features (up to 0.748), while the position information alone yields an F-Measure of up to 0.557. Further, using solely *Tense*, *Tense Indicator* or *Personal Pronouns* with the *MM* corpus the results are clearly better compared to

	MM			*Guo*		
Only	**SVM**	**NB**	**J48**	**SVM**	**NB**	**J48**
Content	.634	.668	.575	.748	.683	.668
Sentiment	.141	.224	.223	.247	.262	.338
Negation	.148	.162	.154	.247	.247	.247
Tense	.248	.243	.242	.323	.326	.358
Tense Indic.	.278	.279	.265	.254	.319	.319
Adjectives	.166	.168	.193	.247	.250	.247
Indic. Indic.	.155	.147	.143	.254	.250	.254
Pers. Pron.	.274	.269	.268	.279	.280	.280
Position	.489	.487	.492	.557	.540	.554
Counts	.143	.241	.257	.248	.276	.300

Table 5: Weighted F-Measures for the Scenario on Scientific Abstracts for Single Feature Groups

the remaining features.

The provided tables only present the weighted average F-Measure for all labels. When we analyzed the results for the single labels, we observed a spread of the F-Measures. For the corpus *Guo*, we observed the maximum F-Measure for the class *Result* with a value of 0.85 and the minimum value was observed for the class *Related Work* with a value of 0.26. For the corpus *MM*, the lowest F-Measure was observed for the class *Conclusion*. This value is 0.0 since the class label appeared very rarely in the corpus and the respective sentences were wrongly classified (due to the lack of training data for this class label). The best F-Measure was achieved for the class *Summary* with a value of 0.85.

We have seen that the results for the corpus *Guo* are substantially better compared to the results for the corpus *MM*. In order to understand this behaviour we ran another experiment where we used only (randomly selected) subsets of *Guo* instead of the complete corpus. The results for the different subset sizes can be obtained from Figure 1. For comparison the result for the usage of the complete corpus *MM* is plotted in the figure. One can see that the F-Measure for *Guo* is increasing steadily with increasing corpus size resulting in a saturation at around 2,500 instances. Comparing the result for the corpus *MM* with the result for *Guo* using the same number of instances shows that the outcome is similar. Hence, it can be assumed that the different results for the complete corpora are a consequence of the different corpora sizes. This also delivers the insight, that the different annotation schemes used for the corpora *MM* and *Guo* do not have a major impact on the result.

5.1.2 Protocols of Scrum Retrospective Meetings

Tables 6 and 7 show the results for the scenario on *Protocols of Scrum Retrospective Meetings*. Similar to the evaluation outlined above, we first evaluated the usage of all features and all features except a single feature group and afterwards analyzed the results for the usage of single feature groups.

Overall, we see that *SVM* and *Naive Bayes* classifier perform similarly (depending on the setting) while the *J48* classifier provides almost continuously worse results. The results for the corpus *Scrum_Subset* are better for all evaluations compared to *Scrum*. We consider this to be the result of the filtering of instances where humans were not to perform a class decision.

Examining Table 6 in detail, one can observe that the removal of the *Content* feature has the largest impact on the overall result. Removing any of the other feature groups results only in minor changes of the F-measure. Interestingly, the *Sentiment* feature group seems to have a major impact when using *J48*. The same trend can be observed when examining Table 7. *Content* is the feature group performing best when used alone while it is followed by *Sentiment* when using *J48*.

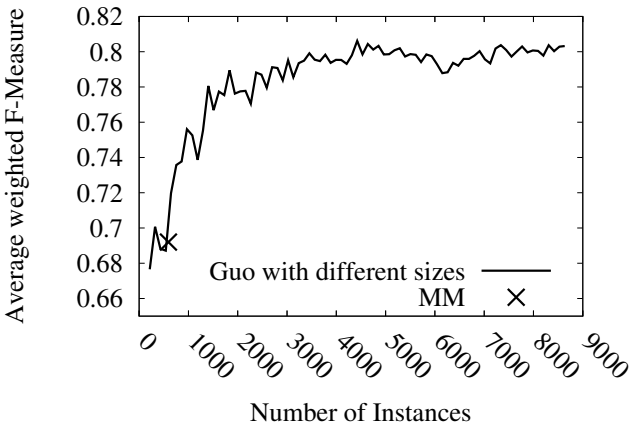

Figure 1: F-Measures when using only Parts of *Guo* for 10-Fold Cross Validation compared to the Result for *MM*

	Scrum			*Scrum_Subset*		
	SVM	**NB**	**J48**	**SVM**	**NB**	**J48**
All	.572	.562	.513	.661	.669	.592
All Except						
Content	.467	.484	.466	.550	.570	.548
Sentiment	.558	.550	.495	.656	.650	.565
Negation	.574	.560	.506	.661	.688	.600
Tense	.565	.567	.522	.673	.669	.603
Tense Indic.	.563	.556	.503	.655	.666	.588
Adjectives	.572	.560	.520	.664	.685	.606
Indic. Indic.	.569	.565	.511	.657	.671	.590
Pers. Pron.	.574	.565	.515	.663	.673	.593
Counts	.567	.560	.510	.653	.680	.605

Table 6: Weighted F-Measures for the Scenario on Scrum Protocols for all Features and for all Features except a single Feature Group

5.2 Evaluation Discussion

As presented, we have evaluated the feasibility of our approach in two different scenarios. While the scenario on *Scientific Abstracts* was evaluated using two different corpora with two different class sets, the scenario on *Protocols of Scrum Retrospective Meetings* was evaluated using one corpus and a filtered subset of the same corpus, both with the same class set. In the first scenario, we have seen a similar result for the two corpora and have shown that the size of the training corpus is a key aspect for a successful classification of sentence types. Using not enough training data yields losses in terms of F-Measure. Although the number of classes was smaller for the scenario on *Protocols of Scrum Retrospective Meetings*, the overall results are worse. Com-

	Scrum			*Scrum_Subset*		
Only	**SVM**	**NB**	**J48**	**SVM**	**NB**	**J48**
Content	.552	.533	.485	.647	.644	.546
Sentiment	.323	.379	.425	.415	.464	.458
Negation	.293	.309	.309	.364	.364	.364
Tense	.382	.392	.399	.260	.346	.462
Tense Indic.	.357	.339	.410	.366	.366	.315
Adjectives	.310	.336	.340	.346	.362	.341
Indic. Indic.	.341	.341	.341	.391	.392	.392
Pers. Pron.	.332	.322	.325	.285	.349	.332
Counts	.324	.375	.410	.280	.376	.363

Table 7: Weighted F-Measures for the Scenario on Scrum Protocols for Single Feature Groups

paring the F-Measure for *Scrum* (0.57) with the results for a subset of *MM* with the same number of instances (0.72), one can see that not only the size of the training corpus is important for a successful classification. Reducing the classification problem for *Scrum* to a problem solvable by humans (filtering out instances that humans are not able to classify), the results become better although the training corpus is shrinking but are still worse than the results for the first scenario. We see the following aspects that could be considered as a reason for this outcome:

- The instances in the corpus *Scrum* often consist of incomplete sentences and have an incorrect grammatical structure. This results in misleading values for the features that rely on part-of-speech taggers. These taggers typically assume complete sentences with correct grammar.
- Many typographical errors can be found in this corpus, probably because of the informal nature of the protocols. This increases the size of the feature space unnecessarily and similar words are not mapped on each other although they should. This could at least partly be resolved by applying automated spelling corrections before feature extraction.
- The sentences are shorter and hence instances have less non-zero values in the feature space (on average 11.32 non-zero values for each instance) compared to the scenario on *Scientific Abstracts* (on average 23.05 non-zero values for each instance) that can be used by the classifier for classification.

6. CONCLUSIONS AND FUTURE WORK

In this article we have examined the usage of machine learning based sentence classification in order to detect the type of a sentence without the usage of domain-specific features and a design without restriction on the set of class. To gather insights on this, we analyzed two different scenarios with two corpora each. We have shown, that the classification is applicable in both scenarios, while the obtained results in the scenario on *Abstracts of Scientific Articles* are better. We further identified some factors for a successful sentence classification and showed limitations of the approach.

As mentioned in the beginning of this paper, the sentence classification can be used for the purpose of information filtering. This approach could be transferred into applications such as tools for efficient search where users can enter a search query together with the type of information they want to search in. This would result in a higher precision of the retrieval process and a reduced search space.

The work presented focuses on single sentences without incorporating contextual information (except for the relative position of the sentence). Taking this information into account could improve the overall results. In particular in scenarios where sentences of a single type appear in textual proximity to each other, such as the job search scenario outlined in the introduction, considering information on neighbouring sentences might improve the overall result.

Finally, the work presented should be evaluated in additional application scenarios.

Acknowledgments

The work presented in this paper was performed within the context of the Software Campus project *SADoku*; it was funded by the German Federal Ministry of Education and Research (BMBF) under grant no. 01IS12054. The authors assume responsibility for the content.

7. REFERENCES

[1] S. Baccianella, A. Esuli, and F. Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*, volume 10, pages 2200–2204, 2010.

[2] R. Daniel. Domain-independent mining of abstracts using indicator phrases. *D-Lib Magazine*, 18(7/8), July 2012.

[3] E. de Maat, K. Krabben, and R. Winkels. Machine learning versus knowledge based classification of legal texts. In *Proceedings of the 2010 conference on Legal Knowledge and Information Systems: JURIX 2010: The Twenty-Third Annual Conference*, pages 87–96. IOS Press, 2010.

[4] M.-C. De Marneffe, B. MacCartney, and C. D. Manning. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454, 2006.

[5] Y. Guo, A. Korhonen, M. Liakata, I. S. Karolinska, L. Sun, and U. Stenius. Identifying the information structure of scientific abstracts: An investigation of three different schemes. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*, BioNLP '10, page 99–107, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[6] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[7] G. H. John and P. Langley. Estimating continuous distributions in bayesian classifiers. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Mateo, 1995. Morgan Kaufmann.

[8] A. Khoo, Y. Marom, and D. Albrecht. Experiments with sentence classification. In *Proceedings of the 2006 Australasian language technology workshop*, pages 18–25, 2006.

[9] M. Liakata, S. Saha, S. Dobnik, C. Batchelor, and D. Rebholz-Schuhmann. Automatic recognition of

conceptualization zones in scientific articles and two life science applications. *Bioinformatics*, 28(7):991–1000, 2012.

[10] S. Mukherjee and P. Bhattacharyya. Sentiment analysis in twitter with lightweight discourse analysis. In *COLING*, pages 1847–1864, 2012.

[11] M. Naughton, N. Stokes, and J. Carthy. Sentence-level event classification in unstructured texts. *Information retrieval*, 13(2):132–156, 2010.

[12] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schoelkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998.

[13] R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[14] G. Salton and C. Buckley. Term Weighting Approaches in Automatic Text Retrieval. *Information Processing Management*, 24(5):513–523, 1988.

[15] S. Teufel, A. Siddharthan, and C. Batchelor. Towards discipline-independent argumentative zoning: evidence from chemistry and computational linguistics. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, page 1493–1502. Association for Computational Linguistics, 2009.

[16] P. Warnier and C. Nédellec. Sentence filtering for bionlp: Searching for renaming acts. In *Proceedings of the BioNLP Shared Task 2011 Workshop*, BioNLP Shared Task '11, pages 121–129, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[17] S. Yelati and R. Sangal. Novel approach for tagging of discourse segments in help-desk e-mails. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 03*, pages 369–372. IEEE Computer Society, 2011.