# Monolingual Retrieval Experiments with a Domain-Specific Document Corpus at the Chemnitz Technical University

Jens Kürsten, Maximilian Eibl
Chemnitz University of Technology
Faculty of Computer Science, Media Informatics
Strasse der Nationen 62, 09107 Chemnitz, Germany
jens.kuersten@s2000.tu-chemnitz.de, eibl@informatik.tu-chemnitz.de

**Abstract**

This article describes the first participation of the Media Informatics Section of the Chemnitz Technical University at the Cross Language Evaluation Forum. A first experimental prototype is described which implements several different methods of optimizing search results. The configuration of the prototype is tested with the GIRT corpus. The results of the Domain-Specific Monolingual German task suggest that combining the suffix stripping stemming and the decompounding approach is very useful. Also, a local document clustering approach used to improve pseudo relevance feedback seems to be quite beneficial. Nevertheless, the evaluation of the English task using the same configuration suggests that the qualities of the results are highly speech dependent.

## Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval
I.5 [**Pattern Recognition**]: I.5.3 Clustering

## General Terms

Measurement, Performance, Experimentation, Domain Specific Retrieval, GIRT-Corpus

## Keywords

Pseudo relevance feedback, Local clustering, Data fusion

## 1 Introduction

In our first participation in the *Cross Language Evaluation Forum* we focused on the development of a retrieval system which provides a good baseline for the *Monolingual GIRT* tasks. Furthermore, we tried to improve the baseline results with several optimisation approaches. The outline of this paper is organized as follows. Section 2 introduces the concepts used to achieve good baseline results and section 3 describes the concepts used to optimise the baseline results. Section 4 deals with the configuration of our submitted runs and section 5 recapitulates the results. In section 6 the results are discussed with respect to our expectations.

## 2 Improving baseline results using the CLEF training data

At first, we had to design a system which implements the well known algorithms of information retrieval. We used the A*pache Lucene API* [1] as core to simplify matters. The developed system consists of the following components: (a) filters for indexing: lowercase filter, stop word filter and stem filter, (b) several algorithms for query construction and (c) an evaluation tool, which allows comparison of different configurations based on the training data. The weighting scheme is embedded in the Lucene core and it is defined in its documentation.

In our primary experiments with the training data provided by the *CLEF campaign* we concentrated on the *Domain-Specific Monolingual German* task. Unfortunately, the results were quite bad results in those experiments. Thus, we searched the respective components for possible improvements. We decided to improve the query construction process and to try different German stemmers.

## 2.1 Constructing improved queries

In a first step only the topic terms (*title* and *description*) in all fields of the *GIRT* corpus were searched. After evaluating our results we analysed the corpus and we observed that the *title*, *abstract*, *controlled-term* and *classification-text* fields should be used for searching, because the other fields do not contain information which can successfully be used for indexing and searching. A slight improvement was achieved by adapting the query construction process to search the topic terms only within the fields above-mentioned. As we wanted to participate with *automatic* runs, we have to consider the *CLEF* guideline [2]. It states that all fields of the *GIRT* corpus can be used for *automatic retrieval*, which is still satisfied.

Also, the query construction had to be improved. There are many ways to combine the search terms within a query in the correct manner. For example, one can build phrases, i.e. that two or more terms have to appear besides each other and in the correct order. But phrases are not the only way to specify or generalise a search for two or more terms which could have a semantic connection. Other methods to combine terms in a query formulation procedure are span queries, fuzzy queries and range queries. We experimented with phrase queries without a significant success and with span queries with a slight improvement of precision. The results of our experiments are shown in table 1.

| *query construction* | *mean average precision* | *average recall* |
|---|---|---|
| standard | 0.3373 | 0.6731 |
| spans | 0.2880 | 0.3632 |
| standard + spans | 0.3515 | 0.6523 |
| standard + spans + topic analyzer | 0.3822 | 0.6920 |

Table 1 – optimised query construction tested with the *CLEF 2003* training data

In our initial experiments we also observed that the topics should be analysed before they are used for query construction. The problem with the topics is that they contain certain terms that are not beneficial for searching the GIRT collection. Thus, we implemented a topic analyzer to remove those terms automatically. We had to be careful by realizing this, since we did not want to eliminate terms that are important for a specific topic. As the described topic analyzer works without any kind of human intervention, we are still constructing our queries automatically according to the *CLEF* guideline. As shown in table 1, we achieved a significant improvement of both precision and recall.

## 2.2 Different stemming algorithms for German

We compared two different stemming approaches for German. The first one is a part of the *Snowball project* [3] and the second one was implemented at *Chemnitz University of Technology* by Stefan Wagner [4]. With the development of the second stemmer, we aimed at the improvement of the effectiveness of our retrieval results. Therefore, we used a stemming approach, which is quite different from well known suffix stripping algorithms. In our approach every term is divided into a number of syllables. These syllables are treated by the system as tokens. Each token then can consist of syllables again. Different implementations of the decompounding approach were tested to detect which one would achieve the best retrieval results. Our decompounding algorithm has two options: (1) remove flexion and (2) avoid overstemming. With the first option we try to remove flexions from the original term and with the second one we try to remove some letters to avoid the problem of overstemming.

Tables 2, 3 and 4 compare the results of the two stemming approaches and their algorithms on the basis of the *CLEF* training data from 2003 to 2005. As one can see in the result tables both stemming approaches outperform the non-stemmed approach. There is also a major improvement with our decompounding approach, which is not very significant at precision. But concerning the recall values we achieve an increase of 11.9% (2004), 14.9% (2003) and 18.7% (2005) compared to the snowball algorithm.

| *stemming algorithm* | *mean average precision* | *average recall* |
|---|---|---|
| none | 0.3114 | 0.6122 |
| plain german decompounder | 0.3751 (+20.5%) | 0.7789 (+27.2%) |
| plain german decompounder + overstemming | 0.4173 (+34.0%) | 0.7950 (+29.9%) |
| plain german decompounder + remove flexion + overstemming | 0.4089 (+31.3%) | 0.7794 (+27.3%) |
| snowball german2 | 0.3822 (+22.7%) | 0.6920 (+13.0%) |

Table 2 – stemming algorithms tested with the *CLEF 2003* training data

| *stemming algorithm* | *mean average precision* | *average recall* |
|---|---|---|
| none | 0.2741 | 0.5526 |
| plain german decompounder + overstemming | 0.3440 (+25.5%) | 0.7054 (+27.7%) |
| snowball german2 | 0.3444 (+25.6%) | 0.6302 (+14.0%) |

Table 3 – stemming algorithms tested with the *CLEF 2004* training data

| *stemming algorithm* | *mean average precision* | *average recall* |
|---|---|---|
| none | 0.2971 | 0.5966 |
| plain german decompounder + overstemming | 0.4048 (+36.3%) | 0.7923 (+32.8%) |
| snowball german2 | 0.3811 (+28.3%) | 0.6674 (+11.9%) |

Table 4 – stemming algorithms tested with the *CLEF 2005* training data

## 3 Concepts to optimise the monolingual baseline results

In this section we will introduce and describe the concepts we used to optimise our baseline results. Three different optimisation approaches and their possible combinations were tested. In the first subsection we take a look at the well known query expansion approach. Thereafter, we describe how we tried to use clustering to get further improvements. Finally, we show which fusion algorithms were used to combine the results of the two stemming algorithms from section 2.2.

### 3.1 Query expansion

Query expansion is one of the most widely used techniques to improve precision and/or recall of information retrieval systems. There are many different approaches in the query expansion field. The main distinction of them is whether manual (user) intervention is needed or not. In our experiments for automatic retrieval we used the pseudo relevance feedback approach. Pseudo relevance feedback is a form of unsupervised learning, where terms from the *top k* documents of an initial retrieval are used to reformulate the original query. Unfortunately, the application of pseudo relevance feedback is not useful in all cases. Many researchers tried to figure out how the reformulation of the query has to be done and which of the available features have to be used for it. Thus, various approaches exist in this field. Again, we tried to keep things simple and implemented a pseudo relevance feedback strategy as follows.

We also used the *top k* documents of an initial retrieval run. From those *k* documents, where $k = 10$ turned out to be a good choice, we only took terms that appeared at least *n* times in our *k* documents. Thereby, it can be ensured that only terms from those documents are used that are term correlated and related to the topic anyway. In our experiments we found out that $n = 3$ seems to be a good value for our query expansion strategy.

In further experiments with our feedback algorithm we tried to improve the performance by using the feedback algorithm repeatedly, but we observed that only using it twice could be useful. Higher numbers of iterations did not improve retrieval performance; in fact they even deteriorated performance. The results of our successful experiments with query expansion are summarized in table 5 and compared to a retrieval run without query expansion (first row). We only show the results of the *CLEF* training data from 2005 to keep clearness. As one can see, we got a strong performance increase with our feedback strategy. The results of the tests with the training data from the other years as well as those with the stemming algorithm from section 2.2 were similar.

| *# docs for PRF* | *min term occurrences* | *iterations* | *mean average precision* | *average recall* |
|---|---|---|---|---|
| 0 | 0 | 0 | 0.3811 | 0.6674 |
| 10 | 3 | 1 | 0.4556 (+19.5%) | 0.8091 (+21.2%) |
| 10 | 3 | 2 | 0.4603 (+20.8%) | 0.8154 (+22.2%) |

Table 5 – pseudo relevance feedback (PRF) tested with the *CLEF 2005* training data

## 3.2 Local clustering

This section describes how we applied clustering in the query expansion procedure. We also mention the algorithms implemented for clustering. The goal of the appliance of clustering was again the improvement of precision and/or recall.

At first, a short review of the different clustering techniques is given. Principally, one has to distinguish between non-hierarchical and hierarchical algorithms for clustering. We tested both concepts to find out what advantages and disadvantages they have in an IR application. The non-hierarchical methods can be further subdivided into single pass partitioning and reallocation algorithms. The non-hierarchical algorithms are all heuristic in nature because they require a priori decisions, e.g. about cluster size and cluster number. For that reason, these algorithms are only approximating clusters. Nevertheless, the most commonly used non-hierarchical clustering algorithm called *k-means* was also implemented.

Although many researchers have been concentrating their forces on hierarchical clustering, some papers on *k-means* like algorithms have appeared in the last few years. For example Steinbach et al. compared some of the most widely used approaches with an algorithm they developed [5]. A short overview on non-hierarchical clustering is also given in [6], but the main part of the paper is on hierarchical clustering. In [7] Peter Willett provided a review of research on hierarchical document clustering. As above-mentioned, we also implemented hierarchical clustering. We applied the *Lance-Williams update formula* [6] to get the possibility to compare different hierarchical algorithms. To avoid confusion we do not explain in further detail how our clustering algorithms were implemented.

As mentioned in the beginning of this section we tried to apply clustering in the query expansion procedure. Therefore, we tried the following configurations: (a) *document clustering*: using the *top k* documents returned from clustering the *top n* documents of an initial retrieval, (b) *term clustering*: using the terms returned from clustering the terms of the *top n* documents of an initial retrieval, (c) *document clustering + PRF*: combining (a) with our query expansion from section 3.1 and (d) *term clustering + PRF*: combining (b) with our query expansion from section 3.1. For document clustering our experiments showed, that clustering the *top n = 50* documents of an initial retrieval provides the best results.

The term clustering approach is quite different from the traditional document clustering, because the objects to be clustered are not the *top k* documents themselves, but the terms of those *top k* documents. To get more reliable term-correlations, we used the top *k = 20* documents for term clustering. A brief description of some methods for term clustering is given in [8].

Table 6 lists the results of our four clustering configurations by testing each of them with the *CLEF* training data from 2005. Clustering is compared with our baseline results (first row) and the simple PRF procedure (second row). The value "10 + x" in the first column of the table means, that the *top k = 10* documents from the initial retrieval were taken and we added those documents returned from clustering, which were not already in this set of 10 documents, i.e. that the number of documents for PRF can vary between 10 and 20.

| *# docs for PRF* | *# terms for PRF* | *clustered obj.* | *mean average precision* | *average recall* |
|---|---|---|---|---|
| 0 | 0 | - | 0.3811 | 0.6674 |
| 10 | 0 | - | 0.4603 (+20.8%) | 0.8154 (+22.2%) |
| 0 | 20 | terms (b) | 0.4408 (+15.7%) | 0.8024 (+20.2%) |
| 10 | 20 | terms (d) | 0.4394 (+15.3%) | 0.7968 (+19.4%) |
| 10 | Unlimited | docs (a) | 0.4603 (+20.8%) | 0.7949 (+19.1%) |
| 10 + x | Unlimited | docs (c) | 0.4726 (+24.0%) | 0.8225 (+23.2%) |

Table 6 – comparison of PRF with/without clustering algorithms on the *CLEF 2005* training data

The results show that local document clustering can achieve a small improvement at precision and recall, when it is combined with the pseudo relevance feedback method described in section 3.1. Furthermore, it is obvious that term clustering did not improve the performance.

## 3.3 Merging

This section will give a short review on the result list fusion approaches we implemented to combine the different indexing (stemming) schemes described in section 2.2. The motivation of applying fusion of different result lists is based on the following two effects. First, the relevant documents retrieved by different retrieval systems should have a high rank and a high overlap. Second, the non-relevant documents of the result lists obtained by different systems are supposed to have a low rank and a low overlap. It is assumed that those hypotheses hold on the result lists we want to fuse.

Merging has also been in the interest of many researchers in the IR field. In the report of Fox and Shaw [9] methods for result list fusion have been compared. In their result summarizing the RSV values performed best. Savoy [10] also compared some data fusion operators and he additionally suggested the *z-score* operator, which performs best in his study. Another fusion operator called *norm-by-top-k* was introduced by Lin and Chen in their report on merging mechanisms for multilingual information retrieval [11]. Beside the most widely known *round robin* and *raw score* fusion operators we implemented the fusion mechanisms depicted in table 7.

| *name* | *formula* | *explanation* | |
|---|---|---|---|
| sum RSV | $\sum_1^k \alpha_i \times RSV_k$ | $RSV_k$ | retrieval status value of document k in list i |
| | | $\alpha_i$ | weighting factor for list i |
| Norm max | $\sum_1^k \alpha_i \times (\frac{RSV_k}{Max_i})$ | $Max_i$ | maximum retrieval status value of list i |
| Norm RSV | $\sum_1^k \alpha_i \times \frac{(RSV_k - Min_i)}{(Max_i - Min_i)}$ | $Min_i$ | minimum retrieval status value of list i |
| z-score | $\sum_1^k \alpha_i \times (\frac{(RSV_k - Mean_i)}{StDev_i} + \delta_i)$, with $\delta_i = \frac{Mean_i - Min_i}{StDev_i}$ | $Mean_i$ | mean retrieval status value of list i |
| | | $StDev_i$ | standard deviation of mean in list i |
| Norm by top-k | $\sum_1^k \alpha_i \times (\frac{RSV_k}{RSV_{top-k}})$ | $RSV_{top-k}$ | average retrieval status value of the top-k documents in list i |

Table 7 – fusion operators based on RSV

We tested the merging strategies with the two results lists obtained by the two different stemming algorithms of table 4 from section 2.2. All the tested fusion operators improved the precision of our primary results as can be seen in table 8. Furthermore, one can observe that *z-score* performed best and *norm RSV* performed almost as good as *z-score*. Another interesting observation is that the simple *round robin* approach also achieved very good results. The only fusion operator that performed poorly compared to the others was *raw score*.

| *fusion operator* | *mean average precision* | *average recall* |
|---|---|---|
| round robin | 0.4296 | 0.7875 |
| raw score | 0.4070 | 0.7938 |
| sum RSV | 0.4267 | 0.7938 |
| norm max | 0.4256 | 0.7867 |
| norm RSV | 0.4311 | 0.8057 |
| norm by top-k | 0.4297 | 0.7878 |
| z-score | 0.4324 | 0.8050 |

Table 8 – fusion operators tested with the *CLEF 2005* training data

## 4 Configuration of official runs

This section provides an overview about the configuration of the runs we submitted. We participated in the *Domain-Specific Monolingual* tasks for German and English. Our runs are described in detail in the following subsections.

### 4.1 Domain-specific Monolingual Retrieval – German

We submitted four runs for this task and their individual configurations were as follows:

*(1) TUCMIgirtde1*

| | |
|---|---|
| Stemmer: | Snowball – German2 |
| Feedback: | Yes, 2-step iterative feedback (see section 3.1) |
| Local clustering: | No |
| Query construction: | fully automatic from topic title and topic description |
| Merging: | No |

*(2) TUCMIgirtde2*

| | |
|---|---|
| Stemmer: | Snowball – German2 |
| Feedback: | Yes, 2-step iterative feedback (see section 3.1) |
| Local clustering: | Yes, document clustering for PRF (see section 3.2) |
| Query construction: | fully automatic from topic title and topic description |
| Merging: | No |

*(3) TUCMIgirtde3*

| | |
|---|---|
| Stemmer: | Snowball – German2 |
| Feedback: | Yes, 2-step iterative feedback (see section 3.1) |
| Local clustering: | Yes, term clustering for PRF (see section 3.2) |
| Query construction: | fully automatic from topic title and topic description |
| Merging: | No |

*(4) TUCMIgirtde4*

| | |
|---|---|
| Stemmer: | Snowball – German2, Plain German Decompounder + Overstemming |
| Feedback: | Yes, 2-step iterative feedback (see section 3.1) |
| Local clustering: | No |
| Query construction: | fully automatic from topic title and topic description |
| Merging: | Yes, sum RSV (see section 3.3) |

All of our experiments used the weighting scheme embedded in the Lucene core, which is a tf*idf weighting scheme (see [1]). With our submitted experiments we hoped to confirm our observations from our experiments with the training data. Mainly, we hoped that the query expansion with clustering would be superior to the query expansion approach without clustering. Moreover, we expected quite good results from our last experiment where the result lists of the two different stemming approaches (see section 2.2) were fused.

### 4.2 Domain-specific Monolingual Retrieval – English

We also submitted four runs for this task and their individual configurations were as follows.

*(1) TUCMIgirten1*

| | |
|---|---|
| Stemmer: | Snowball – Porter |
| Feedback: | Yes, 2-step iterative feedback (see section 3.1) |
| Local clustering: | No |
| Query construction: | fully automatic from topic title and topic description |
| Merging: | No |

*(2) TUCMIgirten2*

| | |
|---|---|
| Stemmer: | Snowball – Porter |
| Feedback: | Yes, 2-step iterative feedback (see section 3.1) |
| Local clustering: | Yes, document clustering for PRF (see section 3.2) |
| Query construction: | fully automatic from topic title and topic description |
| Merging: | No |

*(3) TUCMIgirten3*

| | |
|---|---|
| Stemmer: | Snowball – Porter |
| Feedback: | Yes, 2-step iterative feedback (see section 3.1) |
| Local clustering: | Yes, term clustering for PRF (see section 3.2) |
| Query construction: | fully automatic from topic title and topic description |
| Merging: | No |

*(4) TUCMIgirten4*

| | |
|---|---|
| Stemmer: | Snowball – Porter |
| Feedback: | Yes, simple feedback (see section 3.1) |
| Local clustering: | No |
| Query construction: | fully automatic from topic title and topic description |
| Merging: | No |

With our submission for this task we also wanted to confirm that the query expansion with clustering performs better than the other query expansion approaches. The last experiment was submitted to test whether the simple query expansion performs better than the iterative 2-step method.

## 5 Results of submitted runs

In this section we present the results of our official experiments. The results for the *Domain-Specific Monolingual German* task are shown in table 9 and those for the *Domain-Specific Monolingual English* task are illustrated in table 10. In each row of the tables the value of the best performing configuration is highlighted bold and the worst performing configuration italic.

| *topic no.* | *TUCMIgirtde1* | *TUCMIgirtde2* | *TUCMIgirtde3* | *TUCMIgirtde4* |
|---|---|---|---|---|
| 151 | 0.5735 | *0.5696* | 0.5726 | **0.7766** |
| 152 | 0.6139 | *0.4973* | 0.7000 | **0.7657** |
| 153 | 0.7631 | **0.7734** | 0.7631 | *0.7169* |
| 154 | 0.6928 | 0.7061 | *0.6784* | **0.8012** |
| 155 | 0.4345 | 0.4875 | *0.3405* | **0.4926** |
| 156 | *0.6792* | **0.7088** | 0.6901 | 0.6911 |
| 157 | 0.4380 | *0.3876* | 0.4364 | **0.5263** |
| 158 | 0.5201 | **0.5731** | 0.5320 | *0.4022* |
| 159 | **0.6628** | 0.6344 | 0.6516 | *0.6226* |
| 160 | *0.5431* | **0.6343** | 0.5809 | 0.6261 |
| 161 | 0.7532 | **0.7678** | 0.7569 | *0.6954* |
| 162 | 0.5232 | 0.5312 | *0.5082* | **0.6260** |
| 163 | 0.2978 | **0.3061** | 0.2978 | *0.0816* |
| 164 | 0.0067 | *0.0028* | 0.0060 | **0.0551** |
| 165 | *0.8447* | 0.8573 | 0.8771 | **0.8938** |
| 166 | *0.5773* | 0.6009 | 0.5917 | **0.6184** |
| 167 | 0.5025 | **0.5154** | 0.4986 | *0.4845* |
| 168 | *0.4517* | **0.5130** | 0.4636 | 0.4842 |
| 169 | 0.1584 | 0.1539 | *0.0906* | **0.1957** |
| 170 | *0.5882* | 0.6004 | 0.5931 | **0.6154** |
| 171 | *0.4377* | 0.4584 | 0.4383 | **0.5699** |
| 172 | 0.5005 | *0.4388* | 0.5023 | **0.5697** |
| 173 | 0.5981 | 0.5859 | *0.5634* | **0.6630** |
| 174 | **0.4873** | 0.4802 | 0.4861 | *0.4796* |
| 175 | *0.1571* | **0.2212** | 0.2202 | 0.1810 |
| *aver. prec.* | *0.5122* | 0.5202 | 0.5136 | **0.5454** |
| *# best topics* | 2 | 9 | 0 | 14 |
| *# worst topics* | 8 | 5 | 5 | 7 |

Table 9 – topic by topic precision of submitted runs for the *Domain-Specific Monolingual German* task

| *topic no.* | *TUCMIgirten1* | *TUCMIgirten2* | *TUCMIgirten3* | *TUCMIgirten4* |
|---|---|---|---|---|
| 151 | 0.5331 | 0.5438 | *0.5301* | **0.5592** |
| 152 | 0.5817 | 0.5919 | **0.5948** | *0.5760* |
| 153 | 0.4269 | **0.4660** | 0.4269 | *0.4166* |
| 154 | **0.7248** | *0.7148* | *0.7148* | 0.7237 |
| 155 | **0.4936** | 0.4828 | *0.4624* | **0.4936** |
| 156 | 0.5248 | **0.5368** | 0.5254 | *0.5237* |
| 157 | 0.1952 | *0.1361* | 0.2038 | **0.2100** |
| 158 | 0.2468 | 0.1731 | *0.1369* | **0.2944** |
| 159 | **0.7857** | *0.7675* | 0.7851 | 0.7824 |
| 160 | 0.1626 | **0.2807** | 0.2076 | *0.1395* |
| 161 | 0.5146 | *0.5102* | 0.5132 | **0.5182** |
| 162 | **0.2902** | *0.2590* | 0.2816 | 0.2805 |
| 163 | 0.1485 | **0.1634** | 0.1559 | *0.1435* |
| 164 | 0.0273 | *0.0251* | 0.0262 | **0.0276** |
| 165 | 0.5431 | 0.5313 | *0.5158* | **0.5585** |
| 166 | 0.4630 | **0.4757** | *0.4457* | 0.4703 |
| 167 | 0.0634 | *0.0604* | **0.0792** | 0.0714 |
| 168 | 0.1224 | *0.0949* | 0.1075 | **0.1238** |
| 169 | 0.1162 | **0.1719** | *0.1033* | 0.1159 |
| 170 | **0.2547** | 0.2455 | **0.2547** | *0.2454* |
| 171 | *0.4122* | 0.4309 | 0.4140 | **0.4314** |
| 172 | *0.0792* | **0.1002** | 0.0911 | 0.0835 |
| 173 | 0.3773 | **0.4202** | 0.3810 | *0.3770* |
| 174 | **0.5668** | *0.5511* | **0.5668** | 0.5580 |
| 175 | 0.1199 | **0.1499** | *0.1003* | 0.1197 |
| *aver. prec.* | 0.3510 | 0.3553 | 0.3450 | 0.3538 |
| *# best topics* | 6 | 9 | 4 | 9 |
| *# worst topics* | 2 | 9 | 8 | 7 |

Table 10 – topic by topic precision of submitted runs for the *Domain-Specific Monolingual English* task

## 6 Discussion of results and conclusions

Generally speaking, the results of our experiments were as expected. The result list fusion approach for the *Domain-Specific Monolingual German* task clearly outperformed our other experiments. Moreover, the local clustering of the top documents in combination with the PRF approach achieves slight better results than PRF alone or PRF in combination with local term clustering.

In the results of the *Domain-Specific Monolingual English* task one can see that also PRF with document clustering performs best and PRF with term clustering worked worst. Besides, the differences in the results were very small. Additionally, we have to reinvestigate our experiments for the *Monolingual English* task, because the gap between those results and the results of the *Monolingual German* task indicates, that there might be an unknown issue with those experiments, when we keep in mind that the used corpora are pseudo-parallel.

Concluding our experiments one can say, that for German text retrieval, combining the suffix stripping stemming and the decompounding approach is very useful and further experiments with other languages should be done to determine if one can achieve similar results with them. Also, the local document clustering approach for improvement of PRF seems to be quite beneficial, but again further experiments, especially with different corpora, should be carried out to detect, if the improvement works only within this (domain-specific) environment or not.

## 7 References

[1] The Apache Software Foundation (1998-2006). Lucene.
Retrieved August 10, 2006 from the World Wide Web:
http://lucene.apache.org

[2] CLEF (2006). Guidelines for Participation in CLEF 2006 Ad-Hoc and Domain-Specific Tracks. Retrieved August 10, 2006 from the World Wide Web:
http://www.clef-campaign.org/DELOS/CLEF/Protect/guidelines06.htm (access restricted)

[3] Porter, Martin (2001). The Snowball Project.
Retrieved August 10, 2006 from the World Wide Web:
http://www.snowball.tartarus.org

[4] Wagner, Stefan (2005). A German Decompounder.
Retrieved August 10, 2006 from the World Wide Web:
http://www-user.tu-chemnitz.de/~wags/cv/clr.pdf

[5] Steinbach, Michael; Karypis, George; Kumar, Vipin (2000). A Comparison of Document Clustering Techniques. University of Minnesota, Technical Report #00-034

[6] Rasmussen, Edie (1992). Clustering Algorithms. Published in: Frakes, William B.; Baeza-Yates, Ricardo (1992). Information Retrieval – Data Structures and Algorithms, Prentice Hall

[7] Willett, Peter (1988). Recent trends in hierarchic document clustering. Information Processing & Management Vol. 24, No. 5, pp. 577-597

[8] Baeza-Yates, Ricardo; Ribeiro-Neto, Berthier (2005). Modern Information Retrieval, Pearson Addison-Wesley

[9] Fox, Edward A.; Shaw, Joseph A. (1994). Combination of Multiple searches. Proceedings of the 2$^{nd}$ Text Retrieval Conference (TREC2), NIST Special Publication 500-215

[10] Savoy, Jacques (2004). Data Fusion for Effective European Monolingual Information Retrieval. Working Notes for the CLEF 2004 Workshop

[11] Lin, Wen-Cheng; Chen, Hsin-His (2000). Merging Mechanisms in Multilingual Information Retrieval. Working Notes for the CLEF 2002 Workshop