Merging Classifiers for Improved Information Retrieval

Anette Hulth, Lars Asker
Dept. of Computer and Systems Sciences
Stockholm University
[hulth|asker]@dsv.su.se

Jussi Karlgren
Swedish Institute of Computer Science
jussi@sics.se

Abstract

One prospective way to improve information retrieval is to use several indexing methods to retrieve different sets of documents, and then to merge (or combine) these results into one single result. The merging should be done in a way that produces a final result that is more accurate than the output of any of the individual classifiers. A merging algorithm called *SEQUEL* has been applied for this task to data in the field of information retrieval. This article describes the results of these experiments, as well as conceivable future directions.

1 Introduction

Training several different classifiers and combining their predictions into a single one is a common method for creating a classifier from a set of training data. This approach generally yields a more accurate result than that from the constituent classifiers, which has been shown by a number of researchers. (For an overview of research and results in this area see for example Merz (1999) and Dietterich (1997).) Similar results have also been shown in the document retrieval domain by Bartell *et al.* (1994): using different retrieval algorithms and then combining them may significantly improve retrieval performance.

When applying a merging strategy, the first thing to decide is what different classifiers to use. In the information retrieval domain the source of information is often documents. As documents consist of words, a feasible approach could be to use linguistic methods for retrieval. If each of the methods captures a different aspect of the documents' content, we could possibly retrieve a larger amount of relevant documents on the whole. When merging the different results, the aim is to produce a final result that is more accurate, i.e., has a higher average precision, than the output of any of the individual results. Obtaining this is, however, not trivial, as we only have recourse to weak clues about text relevance, and since results vary between queries, domains and reader preferences, all of which is based on knowledge that is difficult to model reliably.

2 Background

In this section we will describe the Text REtrieval Conference—the framework within which most of the work was done. We will also describe how the relevance is judged, as well as give a motivation for using several information retrieval methods.

2.1 Text REtrieval Conference

The Text REtrieval Conference (TREC) is an annual workshop on information retrieval organised in the form of a competition by National Institute of Standards and Technology (NIST). Several aspects of information retrieval are dealt with within different tracks: *interactive search* (using users' feedback); *spoken document retrieval*; *question answering* (the answer to a stated question is returned to the user); *cross-language retrieval* (the database is multi-lingual), just to name a few. The track that attracted the largest number of participants at TREC-8 was the *ad hoc* track. This task investigates the performance of systems that search a static set of documents using new topics. This corresponds to how we, for example, search the World Wide Web using Altavista: the user states a number of words that describe the wanted document, and a list of documents ranked after presumed relevance is returned by the system.

Each group participating in the ad hoc task is given a large set of documents (approximately 2 GB) plus 50 topics. First, the participants have to produce a new query set from the topics given, thereafter these queries are run against the given collection. For each topic, a ranked list of 1 000 documents retrieved from the collection should be returned.

The document collection used for TREC consists mainly of articles from, for example: Wall Street Journal; Foreign Broadcast Information Service (FBIS); The Financial Times; Federal Register; and The LA Times. (For more on TREC, see Voorhees and Harman (1998b).)

2.2 The Relevance Judgement

In information retrieval, one way to measure how relevant a set of retrieved documents is, is by looking at *precision* and *recall*. By precision we mean the proportion of relevant documents in the retrieved set. Recall is the proportion of relevant documents in the whole collection which the system has retrieved. Both measures take a value between 1.0 and 0.0, where 1.0 corresponds to the best performance.

The measure used in the experiments described below, and one of the measures used in TREC is *mean average precision*. For a single query, the average precision is the mean of the precision obtained after each relevant document that is retrieved. By averaging this value for several queries, we obtain the mean average precision. This measure is sensible to the rank of the relevant documents: having the relevant documents at the top will result in a higher value, i.e., such documents will be given higher weights.

As the data collection is fairly large, as is the amount of runs submitted by the participating teams, it is not possible to calculate the exact precision or recall. Instead, a number of human judges take the top 100 for each query and mark the documents for relevance. These judgements are used as a relevance pool for the rest of the retrieved documents. Hence, a rather large amount of documents will not be judged at all (the documents that are ranked as 101–1 000 and that have not been retrieved by anybody else on the first 100 places). However, if none of the participating systems have retrieved a document in the top 100, sampling tests seem to indicate that it is not very likely to be relevant.

2.3 Natural Language Information Retrieval

The GE/Rutgers/SICS/Helsinki team (further described in Strzalkowski *et al.* (1998)) has for several years participated in TREC with the aim to show that linguistic features can be useful for classifying documents as relevant or irrelevant to a query. To accomplish this, different indexing approaches, term extracting, and weighting strategies have been used to build a number of *streams*. Each stream represents an alternative text indexing method; some require complex linguistic processing, while others are based on simple quantitative techniques. The results obtained from the different streams are lists of documents ranked in order of relevance: the higher the rank of a retrieved document, the more relevant it is presumed to be. The ordering is based on the relevance score—a figure produced by the stream, reflecting the document's accuracy as judged by the system. The streams perform in parallel and the results from the different streams should be merged to produce one final result.

3 The Data Set

The experiments described in this article were performed with TREC-7 data, processed with methods developed for TREC-8. We merged four different streams; these will be described below. For more details on the indexing methods see Strzalkowski *et al.* (1999).

3.1 The Indexing Methods

As mentioned above, the different streams correspond to different indexing methods. In this section, we will describe these streams. For simplicity we will refer to the streams as *one*, *two*, *three* and *four* respectively.

Stream one Automatic expansion, i.e., passages extracted from retrieved documents are added to the query. Uses proximity phrases—that is two (or more) terms must be adjacent (or within a distance) of each other. Runs on *stem* stream (stemmed words and stop words removed).

Stream two Before expansion the terms from stream *one* were added to the topics using a feature in InQuery called #phrase operator (giving higher weight to co-occurring phrases) in case it is a phrase. Automatic expansion. Runs on *stem* stream.

Stream three Automatic expansion. Runs on *stem* stream.

Stream four Single words and head-modifier word pairs. The word pairs contain noun phrases but also other syntactic constructions which have similar meaning.

The words from the title of the topic description are repeated three times before the query is processed for all four streams, thus giving different weights to the different fields in the topic.

3.2 The Data

As stated in the previous section, we performed the experiments using four different streams. Each stream produced a list of 1 000 documents for each query, and as we had a total of 50 queries, the whole set added up to 200 000 documents. The relevance pool (i.e., all documents that were reviewed by the human judges) for TREC-7 consisted of 4 674 relevant and 75 671 non-relevant documents (in total 80 345).

In table 1 below, we will give some statistics about the streams: the number of relevant documents retrieved for each of the stream (rel.); average precision (av. pr.); the number of queries for which the stream had the highest number of relevant documents retrieved as compared to the other streams (best); and the highest and the lowest relevance score for each stream over the 50 queries (max. and min.).

Table 1: Statistics about the data. (For explanations of the abbreviations, see above.)

	one	*two*	*three*	*four*
rel.	2 779	2 604	2 918	1 435
av. pr.	0.2442	0.2266	0.2442	0.0918
best	15	11	9	3
max.	0.4729	0.4721	0.4765	0.4593
min.	0.4062	0.4052	0.4062	0.4036

It is interesting to note that although stream *three* retrieved about 5% more relevant documents than stream *one*, the average precision is the same for the two streams. One should also note that although stream *one* was the best stream on many more occasions than stream *three*, they had the same average precision, thus showing that the ordering of relevant versus non-relevant documents is important.

4 Experiments

In this section, we will first describe the merging algorithm used, and then look at the results we got when applying this algorithm to the task at hand. As mentioned previously, the data used was the TREC-7 data, and we also used the relevance judgements from TREC-7 to measure the average precision.

4.1 SEQUEL

SEQUEL is a merging algorithm developed by Asker and Maclin (1997) and has been used successfully for image classification tasks. The rationale behind SEQUEL is to find the most confident classifier down to a certain threshold. It requires that the list be sorted by—in this case—relevance score. The confidence is calculated by finding the classifier with the highest proportion of correct classifications (i.e., relevant documents) at the top, down to the first non-relevant one. The threshold will be the lowest relevance score within this interval. The items covered by the span are removed from all classifiers.

The training was performed on 40 out of 50 queries—setting aside 10 for testing. Two different implementations of the algorithm were made: one where all queries were sorted by the judgement of the system; and one where the program examined the confidence for each query at a time, taking the average as the result. All non-judged documents were removed, leaving only the judged documents for consideration. In addition, one small set of experiments was performed where the ranking scores were normalised to fall between 1 and 0 (only for the first implementation).

4.2 Results and Discussion

Although the algorithm performed well, no merged list had higher average precision than the best individual stream (which is our criteria for a successful merging). For this reason, no diagrams or curves of these runs will be presented here, but we will conclude with some reflections of possible reasons why the algorithm did not work for this type of data. We will also suggest an alternative method, which still has to be realised before we can draw any conclusions about its ability.

The algorithm tended to favour the best performing classifier (it being the most confident stream) and discarded the additional information that the weaker streams may have contributed with. This could possibly be because the algorithm was not very forgiving: immediately upon finding an irrelevant document the stream was discarded. SEQUEL also tended to work with chunks of documents, covering too many at the time. This could be due to the fact that the relevance scores given by the retrieval systems range over a quite limited span. (The above mentioned normalisation of the scores did not, however, yield a better result.) A shortcoming of the algorithm is that it does not take a possible overlap of the retrieved documents in the different streams into account.

At a certain value the best performing classifier may be considered the default classifier, i.e., it can be used as the single classifier. While experimenting with different threshold values it turned out that the more we let the best performing stream influence the result, the better it got. That was simply because we had fewer of the less well-performing streams lowering the result.

This automatic classification scheme is seemingly tuned for a different type of task. Hence, we need to find more forgiving methods, which do not discard a stream immediately upon finding an irrelevant document: document relevance is a debatable issue in itself, and cannot easily be compared to other classification tasks where the errors are of a more clear-cut nature.

5 Future Work

SEQUEL was implemented to consider the top 1 000 for each query and classifier. This means that for the majority of the documents, we will only know the relevance scores from one or two streams. It would be more appealing to apply a method where every document to a larger extent could benefit from the fact that we use several retrieval methods. Something that points in the direction that this could be a feasible approach is the fact that the ranking scores continue to be more or less at the same level even at end of the list of retrieved documents.

We will conduct a set of new experiments that will take into account the judgement of every stream: all documents that have been ranked among the first 1 000 documents by at least one stream out of N streams will constitute a "document pool" of at least 1 000 documents and not more than N*1 000 documents. We will let every stream score all documents in this pool. As a first experiment, we will implement a simple linear combination of the judgements from the four streams. By a weighted sum of all the scores from each of the streams we will get a total score that includes the knowledge of all streams in question. As mentioned before, the span for the ranking scores is not that large, and therefore even a very small score can alter the ordering of the documents. For these tests we will retain the non-judged documents, and we will experiment with both non-normalised and normalised scores.

A document pool for the four streams would constitute of the 113 135 unique documents (out of 200 000 in the retrieved set, meaning that the overlap is about 43%). Of these 3 408 are relevant. If comparing this figure to the 2 918 documents that stream *three* retrieved the other three streams could—at least in theory—contribute with 490 relevant documents. Below, in table 2, we will give an example from one query. Here we can see that there is a difference of 55 documents between the best performing stream (stream *one*) and the document pool. If including these additional documents in the best retrieved set, the number of relevant documents would increase with 32%.

Table 2: Number of relevant documents found to query no. 354.

one	*two*	*three*	*four*	*doc. pool*
172	142	170	115	227

There is a possibility that some documents could get a better total score by having been given four low scores (too low to be among the best 1 000 documents for any stream) than one with a high score on one and no rating on the other streams. However, if none of our streams ranks a document among the top 1 000 we will discard it. If the experiments with simple linear combinations turn out satisfactory—i.e., better than the non-adapted learning algorithms—we will continue with more sophisticated methods, for example by weighting the different streams. As a baseline, we will use the average precision we get when combining the scores, not looking beyond the top 1 000.

6 Related Work

An approach similar to the one described in the previous section was used by Mayfield *et al.* (1999) at TREC-8. The team used a linear combination with manually set weights for merging three different runs. The ranking scores from the runs were normalised to fall between 0 and 1. An improvement was obtained with the merging compared to the single runs. However, only the top 1 000 was considered for each run, and when having conducted the above described experiments, we will be able to conclude whether our approach is more favourable.

References

Asker, L. and Maclin, R. 1997. Ensembles as a Sequence of Classifiers. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI 97)*, Nagoya, Japan.

Bartell, B. T., Cottrell, G. W., and Belew, R. K. 1994. Automatic Combination of Multiple Ranked Retrieval Systems. In *Proceedings of the seventeenth annual international ACM-SIGIR conference on Research and development in information retrieval (SIGIR'94)*, Dublin, Ireland.

Dietterich, T. G. 1997. Machine Learning Research: Four Current Directions. *AI Magazine*, **18(4)**:97–136.

Mayfield, J., McNamee, P., and Piatko, C. 1999. The JHU/APL HAIRCUT System at TREC-8. Preliminary TREC-8 Report. Gaithersburg: NIST.

Merz, C. J. 1999. Using Correspondence Analysis to Combine Classifiers. *Machine Learning*, **36(1/2)**:33–58.

Strzalkowski, T., Stein, G., Wise, G. B., Perez-Carballo, J., Tapanainen, P., Jarvinen, T., Voutilainen, A., and Karlgren, J. 1998. Natural Language Information Retrieval: TREC-7 Report. In Voorhees and Harman (eds.) (Voorhees and Harman, 1998a).

Strzalkowski, T., Perez-Carballo, J., Hulth, A., Karlgren, J., and Tapanainen, P. 1999. Adhoc experiments performed by the GE/Rutgers/SICS/Helsinki team in the context of TREC-8. Preliminary TREC-8 Report. Gaithersburg: NIST.

Voorhees, E. M. and Harman, D. K., editors. 1998a. *Proceedings of the Seventh Text REtrieval Conference (TREC-7)*. NIST Special Publication 500-242.

Voorhees, E. M. and Harman, D. 1998b. Overview of the Seventh Text REtrieval Conference (TREC-7). In Voorhees and Harman (eds.) (Voorhees and Harman, 1998a).