CondorcetFuse

Condoret voting for run fusion

Evalutation and comparison of an implementation with other fusion strategies

Anna Bonaldo - mat. 1154780
Emanuele Carraro - mat. 1155105
Vassilikì Menarin - mat. 1156200

Information Retrieval course
2017-2018
University of Padova

Contents

1 Introduction

Fusion in information retrieval is combination of retrieval results computed with multiple different retrieval systems into one single "fused" result. This process aim to improve basic systems performances.

Some previous results show that this technique can greatly improve retrieval effectiveness over that of the individual results.

The aim of this work is to present a possible implementation of some basic fusion strategies and compare them to an advanced one: Condorcet fusion [3].

In this project the input documents were taken from the TREC TIPSTER Collection using 50 topics (topics 351-400). We used Terrier for retrieval and indexing and MATTERS for evaluation.

We organized our work as follows:

- **Indexing:** we created four different indexes:
  - Without both stemmer and stop list;
  - only using the Porter stemmer;
  - only using the stop list;
  - using both the Porter stemmer and the stop list;
- **Retrieval:** 10 different retrieval models listed in Table 1;
- **Normalization:** min/max normalization as presented by Lee [1]
- **Fusion strategy:** we compared seven different fusion strategies:
  - 6 basic strategies proposed by Fox and Shawn [2] and listed in Table 2;
  - Condorcet fusion (advanced strategy) as proposed by Montague and Aslam [3]

Retrieval models
BB2
BM25
DLH13
Hiemstra_LM
IFB2
TF_IDF
DFIC
DFIZ
DirichletLM
InL2

Table 1: Retrieval models used

Basic fusion methods	New score formula
CombMNZ	$(\sum_i^N s_i) * (Num\ s_i \neq 0)$
CombSUM	$\sum_i^N s_i$
CombMIN	$min_i^N(s_i)$
CombMAX	$max_i^N(s_i)$
CombMED	$median_i^N(s_i)$
CombANZ	$(\sum_i^N s_i)/(Num\ s_i \neq 0)$

Table 2: Basic fusion methods used

2 Condorcet Fusion

Condorcet-fuse strategy merges single systems results considering the document ranking problem like a voting problem between different systems.

The output of each input system is seen as a list of preferences, where the higher ranked documents beat the lower ranked ones.

The result of Condorcet Fusion is computed like a majoritarian voting between each system preference for each document-query pair. A system preference is its output ranking list for a given query.

Having the lists of different preferences, Condorcet-fuse orders the documents using the Condorcet voting algorithm, estimating the final ordering for a given document-query pair considering the most voted voted rank between systems' preferences on that pair.

2.1 The Condorcet Graph

A Condorcet Graph can be used to represent the majoritarian preference for each document on a given query.

Given 10 models of retrieval with n documents, the corresponding Condorcet graph $G = (V, E)$ has one vertex for each of the n documents.

For each document pair (x, y), there exists an edge from x to y (denoted by $x \to y$) if x would receive at least as many votes as y in a head-to-head contest.

Cycles can simply be viewed as ties. Montague and Aslam [3] suggest a Condorcet fusion implementation that allow graph nodes ordering in a efficient manner (time $O(n * \log(n))$) .

3 Implementation

We follow the Condorcet Fuse implementation suggested by Montague and Aslam [3].

We choose to assign some dummy relevance score to Condorcet's output, the score is assigned based on the rank.

The implementation of Condorcet uses Quicksort, with the following algorithm as comparing function:

```
count = 0
for each of the k search systems do:

        if (sys_i ranks d1 above d2)  count = count+1
        if (sys_i ranks d2 above d1)  count = count-1

        if (count > 0)  rank d1 better than d2
        else  rank d2 better than d2
```

For more information on the code, see README in our code repository

4 Evaluation

The evaluation criteria based on the given **pool uses a binary relevance score**: Relevant and Non-Relevant. The documents left out from the pool are considered to be Non-Relevant. We decided to use **Average Precision** to evaluate a run on the set of topics and **Mean Average Precision** to evaluate a run between different retrieval systems. We always work on normalized data.

We first did an analysis considering the 10 models presented in Table 1 running retrieval and indexing with Terrier's default settings (that is, using both the Porter Stemmer and a stop list) and the fusion's given these 10 input systems.

Then, since the paper on Condorcet Fuse performed some analysis to understand the relation between CondFuse's MAP and the number of input systems, we also tried something similar.

4.1 First Analysis

At first, we wanted to see if using a fusion strategy actually improves the best performance of a simple model run with both the stemmer (we used Porter Stemmer) and a stop list. We proceeded as follows:

1. We computed the AP and the MAP for the 10 input models;
2. We computed the AP and the MAP for the 7 fusion models (6 basic and Condorcet);
3. We selected the 5 systems with the best MAP values.

Figure 1 shows the Average Precision for the top 5 best systems, while Table 3 shows the MAP for all of them.

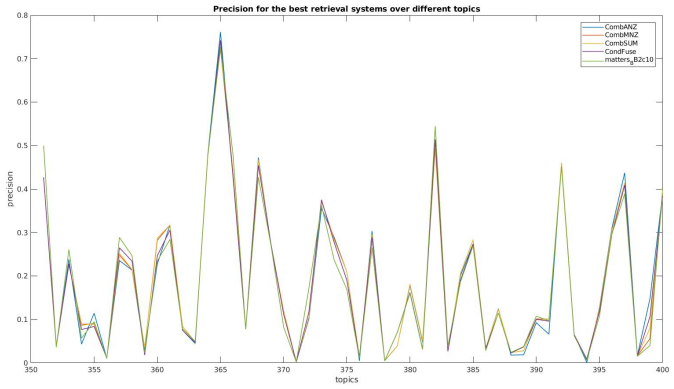

Figure 1: AP over 50 topics for the 5 best systems

Systems	MAP
CombSUM	0.1909
CombMNZ	0.1902
CombANZ	0.1891
CondFuse	0.1883
BB2	0.1881
IFB2	0.1880
DirichletLM	0.1862
InL2	0.1853
CombMED	0.1840
DLH13	0.1829
BM25	0.1827
TF_IDF	0.1821
CombMAX	0.1817
DFIZ	0.1783
DFIC	0.1758
Hiemstra_LM	0.1733
CombMIN	0.1515

: Retrieval systems sorted by descreasing MAP

4.1.1 Results Analysis

The results of these first tests show that it is generally convenient to use a fusion method. Four out of the top five best systems where fusion systems, with Condorcet Fuse beign the fourth best overall.

But, we also noted that the performances of the methods are all very close, and the best and worse topics are the same regardless of the system used.

4.2 Further Analysis

We decided we wanted to test how the performances of the 10 input systems affected the performance of the fusion methods. Furthermore, since Montague and Aslan [3] analyzed the algorithm's performance over the number of input systems, we also tried fusions starting with 2 input systems up to 10.

4.2.1 Analysis over increasing number of inputs

Figure 2 shows a comparison between the MAP for the 7 fusion methods over the number of input systems and with different indexing and retrieval settings.

We noted that the number of input system doesn't much affect the performances of the fusion methods. The main difference was given by the indexing and retrieval settings. In particular, having neither the stemmer or the stoplist was similar to having just the stop list, while having both stemmer and stoplist was more similar to just having the stemmer.

We expected the stop list to be more defining than the stemmer, so we performed some further analysis focusing on what affected more the fusions.

4.2.2 Analysis on the indexing and retrieval settings

We wanted to see how the indexing and retrieval settings affect the fusions' perfomances. Furthermore, we wanted to understand the relationship between the models' and the fusions' performances. We chose to focus on the fusions performed over 10 input systems.

Figure 3 shows the MAP's values for the 10 models over different indexing and retrieval settings, while Figure 4 shows the performance of the fusion method of over the same 10 models with the same settings. CombMIN is not in the graph because of its poor performances and to have a better comparison between the 2 Figures.

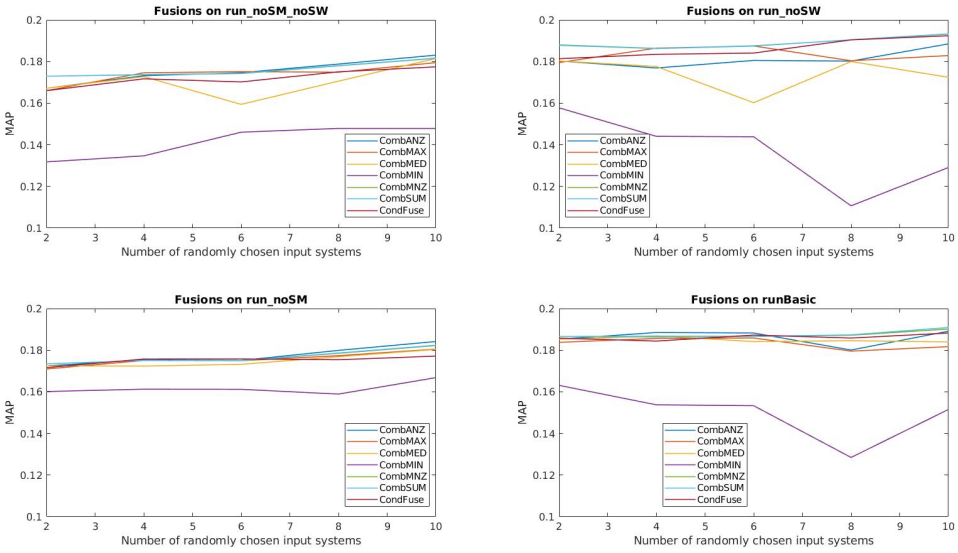

Figure 2: MAP depending in the number of input system and indexing and retrieval settings

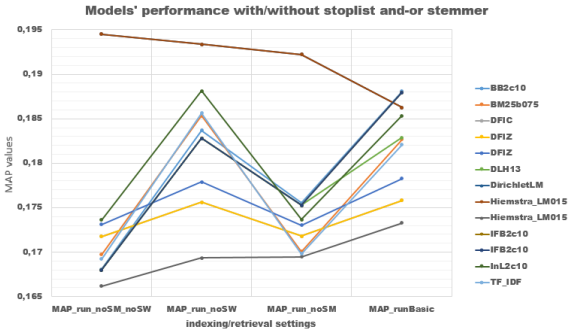

Figure 3: MAP for the retrieval models over different indexing and retrieval settings

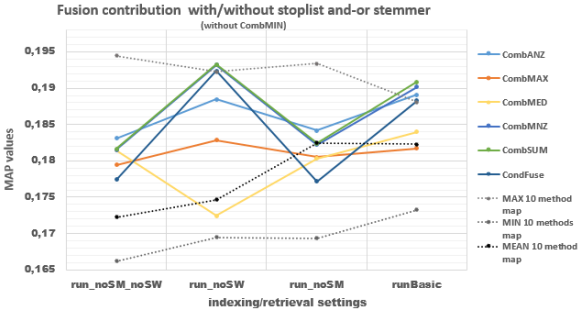

Figure 4: MAP for the fusions over different indexing and retrieval settings

The fusions' performances strongly depend on the input systems' performances. Both in the 10 models and in the fusions the best settings were achieved without the stop list and with both stop list and stemmer. Adding the stop list to the stemmer doesn't change significantly the performance, and neither does using just the stop list compared to using an empty pipeline. This lead us to conclude that, in our collection and with the models selected, the stop list doesn't really affect the performance.

Still, even though the fusions' performance may seem very similar to the models' one, the maximum and minimum MAP values are higher, and we noted a smaller variance between methods.

4.2.3 Analysis on the performance's improvement using fusions

Finally, we focused on the improvement a fusion method can bring given a set of input models.

Figures 5, 6 and 7 show how, over different retrieval and indexing settings, the fusion methods improve compared to the respective MIN, MEAN and MAX models' MAP.

These results show how there is always at least one fusion method that outperforms the "basic" models, even if the improvement is usually quite small. Condorcet ranks above the majority of fusion methods, bringing the biggest improvement when the input models are obtained with just the stemmer or with both the stemmer and the stop list.

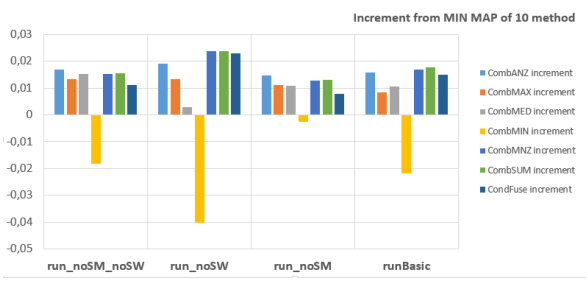

Figure 5: Increment of the MIN MAP value compared to the MIN of the input models over different indexing and retrieval settings

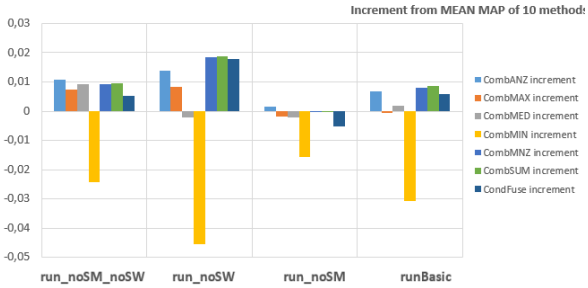

Figure 6: Increment of the MEAN MAP value compared to the MEAN of the input models over different indexing and retrieval settings

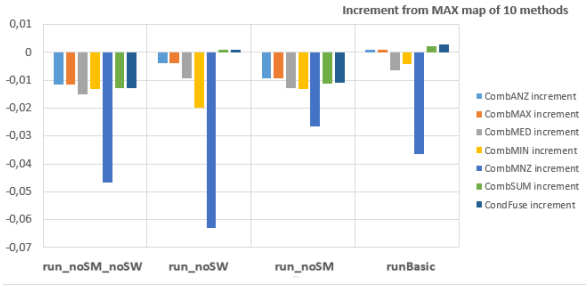

Figure 7: Increment of the MAX MAP value compared to the MAX of the input models over different indexing and retrieval settings

5 Conclusions

We implemented and tested different fusion methods. Our evaluation shows that, even if small, fusion methods actually improve the performance of the single models.

But, the improvement isn't very big and it strongly depends on the performance of the input models, that is if the input models already have a good performance, the fusion will be better. Instead, the number of input system didn't affect the overall performances.

Some fusion methods are clearly worse than others (like CombMIN), being always outperformed, while CondFuse is usually quite good, being one of the top fusion methods tested.

References

[1] Lee, Joon Ho. "Combining multiple evidence from different properties of weighting schemes." Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 1995.

[2] Fox, Edward A., and Joseph A. Shaw. "Combination of multiple searches." NIST special publication SP 243 (1994).

[3] Montague, Mark, and Javed A. Aslam. "Condorcet fusion for improved retrieval." Proceedings of the eleventh international conference on Information and knowledge management. ACM, 2002.