# Search Engine Competition - Strategy

Subscribe for email updates.

No tags yet. + Add Tag

Sort replies by: Oldest first Newest first Most popular

Alwyn J George Signature Track · 7 days ago

Since competition is over, would like to impress upon Top 50 rankers to share their strategy to understand how they had higher MAP scores.

My strategy yielded me good result but could not implement adding synonyms to queries which may have resulted in my low scores.
Though i stand somewhere in 30s for MAP i was surprised my precision at 5 was in top 3.

I implemented my ranker in the following way -

implemented mpln2 optimization as discussed in the white paper referred in the Assignment.
Then weighed combined it with okapi/bm25 with 0.8 weight to mpln2 and 0.2 to bm25.

Would like to know about others.

Also i could not get anything meaningful out of Rocchio feedback method.

Thanks

5 · flag

+ Comment

Anonymous · 7 days ago

I coded up Rocchio, didn't get an entry on the leaderboard. Commented it out and ran vanilla bm25 (k1, b, k3 as per PA1), jumped right into the list. Tried combining bm25 and pl2 (from PA1) in the new_ranker, didn't improve.

Used Jelinek-Mercer with lambda 0.7, suddenly rank was in double digits. Then added a few most common words in moocs.dat to the stopwords list (those that were not already in it). And got a best rank of 19 (before it dropped due to better submissions coming in)!

Further improvements did not help - adding some more stopwords increased the training set MAP, but not on the leaderboard. Have read in papers that Dirichlet-Prior is better than Jelinek-Mercer, but that didn't work at least for this problem, with recommended Dirichlet-Prior mu of 800 (and even with other values).

I believe extending the query with synonyms from a thesaurus is the way to go, but due to some other involvements, was not able to implement it.

0 · flag

Alwyn J George Signature Track · 7 days ago

Nice, could you let me know which stop words did you add.
I added few but it didn't help me.

0 · flag

 Virgil Palanciuc Signature Track · 7 days ago

Cool thanks, I tried Rocchio, but it didn't really seem to improve anything, thought I made a mistake, but it seems lots of people tried it and it didn't work.

Best thing that I did apparently was a parameter auto-tuner that I ran with the builtin algorithms, with it I got best results on my test set with JM, using a lambda value of 0.72. Was ranked 3rd when I submitted it, but I had no time to do further improvements and apparently I dropped to 126 in the meantime :)

0 · flag

+ Comment

Renaud Dufour · 7 days ago

Hi, here is the strategy I followed

```
## First I spent a few days learning basics of C++ and understanding the competition.cc fil
e...

## As a first step I tuned the different built-in rankers

* bm25
* absolute-discount
* dirichlet-prior
* jelinek-mercer
* language-model
* pivoted-length

I tuned each ranker by varying all their parameters over a wide range of values, then refined.
I saved the results to text files and plotted the MAP curve to have better idea of each ranker
performance.
From this first analysis, bm25 and jelinek-mercer appeared as the most promising (I reached MA
P on the Test data of about 0.622 with bm25).
```

```
## Then I played with the analyzer filter chain

The only improvement was obtained by changing the 'min' parameter of length_filter from 2 to
1.
This allowed a slight improvement of the Test MAP, reaching about 0.627.
(This may be due to some query related to programming language like R or C, although I didn't
check in more detail)

## Then I implemented and tuned few other rankers

* pl2
* MPtf2ln
* MDtf2ln

The latter two are described in the article "Diagnostic Evaluation of Information Retrieval Mo
dels".
pl2 appeared not very efficient (Train MAP only up to 0.58).
MPtf2ln and MDtf2ln gave good results and I achieved my best score with MPtf2ln reaching a MAP
on the test data of 0.6317. This ranker also appeared to be quite robust (the MAP curves are r
elatively flat around the maximum). So I decided to go on with this ranker for the rest of my
investigations.

## Other unsuccessful strategies

* Implementation of Rocchio Feedback (I just added all terms of the top 10 documents to the qu
ery, varying weights alpha/beta) - no improvement
* Adding new stopwords. I analyzed the words distribution in both query and mooc datasets, the
n added most common words such as "learn", "course" or "data" to the stopwords list. - no impr
ovement
```

So basically I essentially focused on tuning different rankers, and did not spend much time on the rocchio feedback.
It was already very time consuming to learn some basics of C++ and figure out how to implements a feedback loop.
With this I am ranking #22, with MAP of 0.6317 and precision at 5 of 0.553.
I stopped working on the assignment more than one week ago, there has been a lot of improvements recently and I'm curious to learn about the different strategies !

↑ 3 ↓ · flag

+ Comment

 Toh Zhiqiang Signature Track · 6 days ago

Hi,

My strategy is described in the following article:

https://drive.google.com/open?id=0BzoEQjwojB9fS1IxV3FoM3hlVFE&authuser=0

8 · flag

Alwyn J George Signature Track · 6 days ago

Nice, how did you narrow down on p1, p2, .. p5 values.
I too tried weighing combining 3 of the rankers but my score dipped so thought 2 should be good enough.

0 · flag

Toh Zhiqiang Signature Track · 5 days ago

I tried permutation of values for a subset of the tunable parameters, e.g. for $p_1 \in \{0.1, 0.2, \cdots, 1.0\}, \lambda_1 \in \{0.5, 0.6, \cdots, 2.0\}, k_1 \in \{1.0, 1.1, \cdots, 2.0\}$, run the search engine, and submit the output file if the training MAP is above a certain threshold.

1 · flag

Gavin Conran · 2 days ago

Nice work Toh.
My solution was a sub-set of yours. After doing a crash course in C++ I created a new ranking class with the score_one function returning "(alpha_ * bm25) + ((1 - alpha_) * jm)".
I manually tuned alpha to 0.29 for the optimum result.
Rank 31.

+gavin

1 · flag

+ Comment

Hema Tanikella Signature Track · 5 days ago

I am not in the top 50 but wanted to ask if anyone tried the BM25+. I tried it as one of my rankers but it did not improve the performance. I am curious because I remember the prof saying it has been proven analytically to have improved the performance. Another comment is that in BM25, I experimented with various formulas for the IDF and they had the biggest impact for me in terms of the MAP.

0 · flag

+ Comment

Hema Tanikella Signature Track · 5 days ago

Also, couldn't make much of Rocchio and pl2 performance was quite poor and so was Dirichlet-Prior.

0 · flag

+ Comment

Anonymous · 5 days ago

Waiting for some more toppers to enlighten us. :)

0 · flag

+ Comment

Jiacheng Pan Signature Track · 5 days ago

I tried following two ways: (two days before the deadline...

- I modified the stop list and used a combination of BM25 and PL2 as my new ranker.
  For stop list, I added following terms: `<s>, </s>, an`.
  For the new ranker, I merely put both of the formulas into the score_on function, with an extra parameter alpha to tune the weights.
  I then wrote a python script to try different permutations of the parameters -- big intervals first, and did some local searches...
  I finally got a ~0.655 MAP on the 100 test set, while got 0.6322 MAP for the submission.
  I ranked 9 at that time. *(very much to my surprise...* (Finally I ranked 18...)
- I modified the stop list and implemented MPtf2ln described in the paper provided in the instructions. (of course, I did it step-by-step following the section 6, and it indeed shows that ranker with a combination of TF and LN yields better MAP than individual ones...
  For stop list, I did exactly the same as 1.
  For the new ranker tuning, I managed to get a ~0.662 MAP on the test set. ( And I thought I would have a better rank then...
  To my surprise, this time, the submission only returns 0.6263 MAP.....
  I thought I overfit the test data... But using a combination of para that produced a worse MAP did not improve the submission MAP...
  And that concludes my trials, due to limited time... (busy days...

Besides, following attempts were not so successful:

- Rocchio feedback
  I tried different counts and different weights, it turned out that the smaller the counts and weights are, the better MAP I could get... (I used all the terms in the searched document as the feedback, just as is implemented in this thread.
  Also I tried to apply different weights to terms that occur in the query and those not in the query.
  Still no luck...
- bi-gram analyser
  I tried to use both uni-gram and bi-gram analyser, but the result shows much worse MAP.
  I then tried to combine feedback into it, no luck... (using BM25, the MAP value was always below 60...
  Then I tried to apply different weights to uni-gram terms and bi-gram terms, for the feedback, no luck still...
  Then I gave up.....

- synonym expander
  I plan to try it later...

**Also, looking forward to more toppers for enlightenment!!**

Cheers!
Jiacheng

1 · flag

+ Comment

Alwyn J George Signature Track · 5 days ago

Bottom line is implement the optimized ranking functions from the paper and weighed combine them with other rankers.
Any other relevations especially from Professor may be appreciated.

0 · flag

+ Comment

Robert McAnany · a day ago

I ended up in 5th place, MAP 0.63639. A copy of my description.txt is pasted in below.

==========
The following shows the approaches I took during the competition. The format is:
Approach: Result. MAP. Rank.
Description.
The entries are in approximate chronological order.

Ran competition with all defaults: Helped. MAP 0.61422. Rank 14.

NOT learn C++: Helped. MAP 0.61422. Rank 14.
There was no way I was going to learn C++ from scratch in 2 weeks. Instead, I modified competition.cpp just enough to write out text files for processing. I used R for all tasks that were not already implemented in default MeTa.

Manually tuned ranker parameters: Helped. MAP 0.61961. Rank 13.

Scoring ensemble rankers: Helped. MAP 0.63052. Rank 1.
After tuning, I ran all the rankers, normalized their individual scores, and added them up. The summed scores were sorted for final selection. Scores for each query and each ranker were normalized by

(score - score_min)/score_max,
which gave values between 0 and 1.

```
method = "absolute-discount"
gamma = 0.814 #manual optimization 0.630835

method = "bm25"
k1 = 1.95 #manual optimization 0.655785
b = 0.8
k3 = 500 #k3 had no effect in MAP scores

method = "dirichlet-prior"
mu = 228.75 #manual optimization 0.622947

method = "jelinek-mercer"
lambda = 0.71875 #manual optimization 0.655282

method = "pivoted-length"
s = 0.1328125 #manual optimization 0.645449
```

Global normalization: Didn't help.
The above normalization is "local." Each query has a range of 0-1. Tried a "global" approach for each ranker,
(score - score_min_all_queries)/score_max_all_queries.

Ensemble machine learning: Didn't help.
Used the normalized (local and global) ranker scores as input to an ensemble of machine learning algorithms. Included Random Forest, glmnet, logistic regression, neural networks, and self-organizing maps. My suspicion is that the raw scores were not informative enough by themselves to help.

Synonyms: Didn't help.
Got query word synonyms from the R package "qdap". Some words, like "play," had hundreds of synonyms. I kept only those that had a small number, e.g. "distributed." To avoid undue influence of the added words, I increased the number of times the raw query terms were included in the final query. This was a parameter, old_query_rep, that I selected by trial and error.

Auto tuning ranker parameters: Helped. MAP 0.63381. Rank 1.
Every time a new approach was tried, the ranker parameters needed to be adjusted. This was tedious since I was using 5 rankers in the scoring ensemble.

```
method = "absolute-discount"
gamma = 0.8134115 #auto optimization 0.630845

method = "bm25"
k1 = 1.748047 #auto optimization 0.655904
b = 0.8136719
```

k3 = 500

method = "dirichlet-prior"
mu = 283.5286 #auto optimization 0.625888

method = "jelinek-mercer"
lambda = 0.7191406 #auto optimization 0.655326

method = "pivoted-length"
s = 0.1308594 #auto optimization 0.647612

Modified Rochio: Didn't help.
The query result Title seemed to contain the most succinct information. If N rankers agreed on the first query result, I added those words to the query. Did tuning on N and old_query_rep, followed by ranker auto tuning.

Wiki API synonyms: Helped. MAP 0.63638. Rank 4.
The R package synonyms were too general for this set of queries. For each query, I used the Wiki API to grab the first 50 unique words from the wiki page if it existed. Did tuning on old_query_rep, followed by ranker auto tuning.

method = "absolute-discount"
gamma = 0.7891602 #auto optimization wiki50 0.651135

method = "bm25"
k1 = 1.846680 #auto optimization wiki50 0.659421
b = 0.8861545
k3 = 500

method = "dirichlet-prior"
mu = 309.93924 #auto optimization wiki50 0.65437

method = "jelinek-mercer"
lambda = 0.6491862 #auto optimization wiki50 0.648343

method = "pivoted-length"
s = 0.1280056 #auto optimization wiki50 0.649754

Use raw queries for "jm" and Wiki synonyms for all others: Helped. MAP 0.63639. Rank 4.

method = "absolute-discount"
gamma = 0.7891602 #auto optimization wiki50 0.651135

method = "bm25"
k1 = 1.846680 #auto optimization wiki50 0.659421
b = 0.8861545
k3 = 500

method = "dirichlet-prior"
mu = 309.93924 #auto optimization wiki50  0.65437

method = "jelinek-mercer"
lambda = 0.7191406 #auto optimization 0.655326

method = "pivoted-length"
s = 0.1280056 #auto optimization wiki50 0.649754

Quit working on the competition:  Helped.  MAP 0.63639.  Rank 5.
It was fun and I learned a lot.  Now it's time to get back to my real life.

5 · flag

Sergii Signature Track · a day ago

thank you, now I know where to look for synonyms in case I need them in the future :)

1 · flag

Chedi Bechikh Ali · 8 hours ago

Thank for sharing that, can you please explain more how do you use the machin elearning algorithm to optimise the paramters ( where do you take these algorithms implementation, and how do integrate them with meta, i think you used R)

Your approach is very interesting and if possible you can explain it with more detail

best regards

0 · flag

Robert McAnany · 6 hours ago

Actually, I was trying to use the machine learning ensemble to improve the final score prediction.

For each query I assembled a table like
doc1, bm_score, jm_score, pl_score, ad_score, pl_score
doc2, bm_score, jm_score, pl_score, ad_score, pl_score
...

My first approach was to simply add up all the scores and sort based on that.  It worked pretty well, but I thought an ensemble of machine learning algorithms might do a better job.  I used the new R package caretEnsemble to do it.

Here's a snippet showing how easy it is to do...

```
    inTrain <- createDataPartition(y=training$relevance, p=0.7, list=FALSE)
    validation <- training[-inTrain, ]
    training <- training[inTrain, ]

    cvCtrl <- trainControl(method="boot", number=25, savePredictions=TRUE,
                           classProbs=TRUE, index=createResample(training$relevance,
25),
                           summaryFunction=twoClassSummary)
    x_cols <- which(grepl("score", names(training), fixed=TRUE))

    model_list <- caretList(x=training[, x_cols], y=training$relevance,
                            trControl=cvCtrl,
                            methodList=c("glm", "rf", "glmnet", "svmRadial", "gbm",
"xyf", "nnet"))

    greedy_ensemble <- caretEnsemble(model_list)
```

Unfortunately, it didn't improve my MAP, so I quit working on it.

In terms of integration with MeTA, well that was pretty lame.  I modified competition.cpp to write out doc_ids and scores for processing.  I called MeTA from R using

```
    system("./competition config.toml | tail -n 1 > outfile.txt")
```

As for ranker parameter optimization, I simply did a sweep across the parameters and picked the value that gave me the highest MAP on the training set.

0 · flag

+ Comment

Eduardo Rodriguez · a day ago

I implemented Rochio. No improvement.

I combined bm25 with pl2. No improvement.

I combined bm25 with Jelinek-Mercer, tuning the parameters of both rankers and the weight of each one (automatically). I ranked #19.

I was not able to implement synonyms. I did not know any dictionary I could use. It was the last day, so I discarded it. Maybe I will try it in the future.

0 · flag

+ Comment

Cristi Dumitrescu · a day ago

I ended up #4, I stopped trying to improve when I decided I had spent too much time on the project. Instead of boring you with all that I tried, I'll tell you the one thing that I did that had the most significant effect: I wrote a PHP script that simply duplicated the document titles in the database, the idea being that everyone would look at the document title - not at the document contents - when submitting their queries, hence the document titles should have an increased weight. Also, another thing that greatly helped was modifying MeTA to work with floating point weights for the query and document terms.

The other thing that I was going to try and that I'm sure would have had a big effect on the MAP would've been to build a word distance index for each document (for each word in a document, save the top-k closest words together with their average distance and count) and rank the documents based on that. I was planning to use the proximity of each query term to each other query term in each document as a weight multiplication factor for that term for that document, the idea being that a "**political news**" query should rank higher in "...today's **political news**..." than in "The **political** scene is dominated by buffoons. In other **news**, dolphins ate Hawaii..." despite their having an identical term count. However, I deemed that this would be way too much effort for the purpose especially given the fact that MeTA is written in C++ and that I hadn't bothered to install a development environment and was using a simple text editor to edit files.

For what it's worth, Rocchio resulted in a small boost after I modified it so much that it no longer even looked like Rocchio (I used Jelinek-Mercer - not BM25 or anything else, haven't even tried those - so I had to adapt it quite a bit); I used it with words exclusively from the topmost ranked result (!), anything else resulting in a performance decrease. I also tried re-weighting the query terms based on word association analysis, which was a rather extensive pain in the ass and which didn't help at all.

My $0.02: MeTA is too immature for this course and C++ is not such a good choice nowadays. I realize this course is not geared towards IT beginners, but perhaps a more stable framework should've been chosen in conjunction with a "less hardcore" programming language. Not to mention that I'm pretty sure the majority of the people taking this course are running Windows. It's probably too late to veer off the MeTA course now, though.

3 · flag

+ Comment

Kurt Kuppens Signature Track · a day ago

Ranked #43

From the paper Diagnostic Evaluation of Information Retrieval Models section 6.2 (p34)
I used MPtf1 function with the following parameters

s = 0.3
alpha = 0.03

mu = 1007

I obtained these values modifying the l2_tune function from the first Assignment and evaluating MPtf1 for different values for s,alpha and mu.

I experimented with Rochio -> no improvement. (I have to comment that I am not sure about my implementation)
I experimented with changing stopwords -> no improvement. (To be precise: some improvement on the training set adding some stopwords however no improvement on the final test)

1 · flag

+ Comment

 Sergii Signature Track · a day ago

Hi Everyone,

My Rank is #8 with MAP 0.634494655936.

BM25 and Jelinek-Mercer were linearly combined, Each of the rankers was tuned to given the best MAP. Then $\alpha$ was tuned too.
Rocchio feedback didn't help.
Synonyms were not used. Like Eduardo in the previous comment experienced, I couldn't find quickly the dictionary of synonyms. I tried to manually use several synonyms for technical terms (such as programming - computer - algorithms), but that didn't help to improve score for the selected queries.

The general thought on why even the maximum MAP in a competition table is low (as I can subjectively judge, less than 0.65 is low). The relevant documents were evaluated by participants of this course (remember couple of exercises in programming assignments). As I reviewed the queries and the resulting documents by relevance, many documents appeared to me relevant to the query, but were marked as non-relevant. As a most trivial example, one of the queries was 'chinese'. And most if not all of the first 10 retrieved documents contained 'chinese' in its header, but non of them was marked as relevant. And you can find many other examples, where precision at 10 was 0-0.2. That said, it is hard even for a person to discover the logic of which documents are relevant in this setting.

The competition helped to learn new techniques, understand META library better, and obtain practical experience in Text Retrieval. I thank to Prof. Zhai, all TA's and everyone who helped to prepare and supervise this course.
Looking forward for new challenges in subsequent courses!

Sergii

2 · flag

+ Comment

Birjodh Tiwana Signature Track · a day ago

I am currently leading the board
MAP 0.643920577821
Precision at 5 0.559851301115

I used a combination of BM25 and dirichlet-prior.
Besides that I did the following:

1. I stayed away from pseudo relevance. The data seemed to little and too noisy to be useful for pseudo relevance and I learnt by reading other people's experience on forums. Thanks for being so helpful.

2. I scanned the documents using a python script to get a list of the most popular words inside the documents. I changed the lemur-stopwords.txt to add all the words which occurred more than 20,000 times in those documents and also seemed useless for queries. That helped push my ranking up.

3. The most important thing which pushed me to the top was the realization that when users were marking the documents as relevant or non-relevant, they only looked at titles. Therefore I wanted to give a term match in title a higher weight than a term match in the rest of the document . To achieve that I first tried to modify the meta code but soon I realized an easier way. I simply changed the documents, I copied over the title 4 times in every document. So a term match in the title was given 4 times more weight than the term match in the document.

4 · flag

Sergii Signature Track · a day ago

Good job, Birjodh!
Didn't think about #2, mostly because my understanding was that TF and IDF should handle the frequently occurred words.
I had the same idea as you have in #3, that people ranked documents based on the title only. Unlike you, I removed all document bodies from the index and kept titles only (by copying moocs.dat.names to moocs.dat). That didn't help to improve ranking, however. So your idea to replicate the title several times and in such simple manner is great!

Sergii

1 · flag

Anonymous · 19 hours ago

Congrats Birjodh!

Even I tried adding stopwords with count > 20k, 25k, 30k, 40k. Only the words 'course', 'courses' improved grade on the leaderboard for me. When I added too many stopwords,

say all above 20k like you said, I actually got lower scores on the leaderboard, even though the MAP on training set increased.

Of course, I only used vanilla JM ranker with lambda 0.7. As you said, your copying the title 4 times did the actual job.

Actually, adding too many stopwords makes the search engine biased with the topic - trying with different combinations of stop words is like trying to fit the test set. I want to make one observation related to this: In the leaderboard, all scores are within 0.03 MAP. The improvements mostly came by trying to fit the test set. The only takeaway from this competition, for building an unbiased search engine, would be the approach of combining multiple rankers with their param values, and giving weightage to specific areas of the doc like you did.

And yes, most people would give feedback on the docs based on the title. I did too at first, but I submitted 2 more feedbacks where I actually checked the links. For one query I even watched a youtube video to learn about acoustic and electric guitars :)

0 · flag

+ Comment

Birjodh Tiwana Signature Track · a day ago

Sergii. I also tried removing all the document bodies first. That was my first instinct.

0 · flag

+ Comment

Jennifer Evans Signature Track · a day ago

You guys are all geniuses! Awesome! Thanks for sharing!!

I wonder if this "multiplying or heavier weighting of titles" causing results to be better than state of the art (BM25) is an artifact of the grading/judging process used? I read through the above and this was used at least twice in top ten results.

0 · flag

Sergii Signature Track · a day ago

As Birjohd found, giving higher weights to the headlines helped to improve the score. Makes sense, since the judgements were given by headlines only. I also read in one article for text analytics for predictions of news popularity that headlines were given the higher weight. That also makes sense to me, since a headline plays a big role: a person will read the news article more likely if the headline is interesting.

1 · flag

Birjodh Tiwana Signature Track · a day ago

I think Sergii covered it pretty well. To add to it think about it from a user perspective. Even if we give the whole document to the user they will not read through all of it to figure out if it is relevant. They will first look at title and if title sounds promising only then read the rest of it. A very relevant example is google search. We first look at the title (and a litle bit of intro) and the click it. The click on the title is used by google as a signal of feedback.

Looking at it from another angle, a nicely written title always tells us what the document should be about. Usually (not always) most important information about the document should be in the title.

0 · flag

+ Comment

 Corey Auger Signature Track · 6 hours ago

I was laserKitty with a score of 0.625611509264

I too made the most progress with a linear combination of BM25 and jelinek-mercer.
It seems that some people were able to tune this to slightly better then my own parameters.

The one things that I did different was to use "Word Net". On linux you can use wordnet database
http://linux.die.net/man/1/wn

I originally tried expanding terms that I found a hit on... but this did not improve the score. So I then tried to help "small" query's only. I think what you really want to do is expand on well known words.. to ones that may be more specific. This of course is extremely difficult as it turn into a bit of an NLP problem with you trying to establish context and disambiguate terms from small to text query's.

I also noticed that if you could expand acronyms (for example "asap") and correct spelling mistakes .. this would further increase the MAP.

In the end I could not get my wordnet solution to turn over higher MAP :(

Thanks to everyone for sharing.

0 · flag

+ Comment

New post

To ensure a positive and productive discussion, please read our forum posting policies before posting.

B | I | Link | <code> | Pic | Math | Edit: Rich | Preview

Make this post anonymous to other students

Subscribe to this thread at the same time

Add post