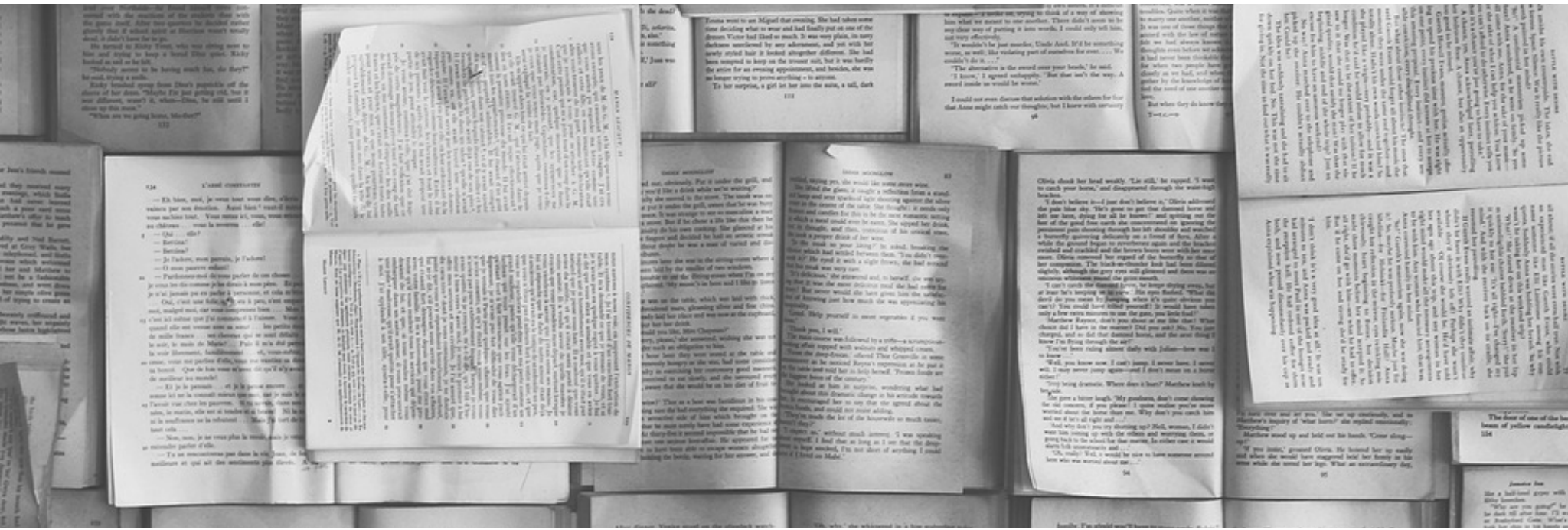
TOPIC 4: Efficient VSM

GROUP 9



Na Gong
Qian Xia
Siying Liu

AGENDA

LEARN UNDER FIRE

1. Data Structure
2. Preprocessing
3. Evaluation Metrics
4. Modeling & Results
5. Comparison Overview

DATA STRUCTURE

1. **NFCopurs**: Medical text data set (English)

2. **doc_dump**: 5371 documents
   - ID, URL, TITLE, ABSTRACT

3. **nfdump**: 3437 queries
   - ID, URL, TITLE, MAINTEXT, COMMENTS, TOPICS_TAGS ...

4. **2-1-0.qrel *3**: 134,295 relevance judgments with three levels
   - QUERY_ID, 0, DOC_ID, RELEVANCE_LEVEL

5. **stopword**: 543 given stopwords

AGENDA

LEARN UNDER FIRE

1. Data Structure
2. Preprocessing
3. Evaluation Metrics
4. Modeling & Results
5. Comparison Overview

PREPROCESSING

1. **Query cleaning**
   - remove queries without relevance judgment
2. **Feature Selection**
3. **Lower Case**
4. **Remove Stopwords**
5. **Tokenization**
6. **Remove punctuation**
7. **Stemming**

RegexpTokenizer

PorterStemmer

5371 document + 3237 queries

AGENDA

LEARN UNDER FIRE

1. Data Structure
2. Preprocessing
3. Evaluation Metrics
4. Modeling & Results
5. Comparison Overview

EVALUATION METRICS

1. MAP (Mean average precision): Average of average precision over the set of queries.

$$\text{MAP} \quad = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} P(R_{jk})$$

2. nDCG (Normalized discounted cumulative gain):

$$\text{nDCG}_\text{p} = \frac{DCG_p}{IDCG_p}$$

DCG with penalty:

$$\text{DCG}_\text{p} = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)}$$

Alternative DCG:

$$\text{DCG(k)} = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$

AGENDA

LEARN UNDER FIRE

1. Data Structure
2. Preprocessing
3. Evaluation Metrics
4. Modeling & Results
5. Comparison Overview

BASIC VSM

1. 6 TFIDF Schemas (vector length: about 18000)

TFIDF	Document term weighting	query term weighting
Schema 0	$\frac{1+\log f_{t,d}}{1+\log(max_t f_{t,d})}.\log\left(\frac{N}{n_t}\right)$	$\frac{1+\log f_{t,d}}{1+\log(max_t f_{t,d})}.\log\left(\frac{N}{n_t}\right)$
Schema 1	$\frac{1+\log f_{t,d}}{1+\log(max_t f_{t,d})}.\log\left(\frac{N}{n_t}\right)$	$f_{t,d}$
Schema 2	$f_{t,d}.\log\left(\frac{N}{n_t}\right)$	$(0.5+0.5\frac{f_{t,d}}{max_t f_{t,d}}).\log\left(\frac{N}{n_t}\right)$
Schema 3	$1+\log f_{t,d}$	$\text{Log}(1+\frac{N}{n_t})$
Schema 4	$(1+\log f_{t,d})\log\left(\frac{N}{n_t}\right)$	$(1+\log f_{t,d})\log\left(\frac{N}{n_t}\right)$
Schema 5	$\frac{1+\log f_{t,d}}{1+\log(max_t f_{t,d})}.\log\left(\frac{N}{n_t}\right)$	0 or 1

2. distance measurement : cosine similarity

3. evaluation methods : MAP and nDCG

BASIC VSM

	S0	S1	S2	S3	S4	S5
MAP	0.13	0.11	0.11	0.13	0.13	0.09
nDCG1	0.49	0.46	0.48	0.49	0.49	0.43
nDCG2	0.49	0.46	0.48	0.46	0.49	0.43
time	518	1158	499	555	507	1092

- MAP & nDCG: 3237 queries + 5371 documents
- time: cosine similarity computation time for all queries
- The TFIDF schema 0 and 4 have similar performances which are better than other schemas.
- The following speed up models and all result comparisons are based on TFIDF schema 0.

RANDOM PROJECTION

1. Generate M random vectors
   Based on the normal distribution of each term in documents(query)
   M : 10, 100, 300, 500, 1000, 2000, 2500, 3000

2. Compute the new documents(query) vector based on original documents(query) TFIDF and the M random vectors

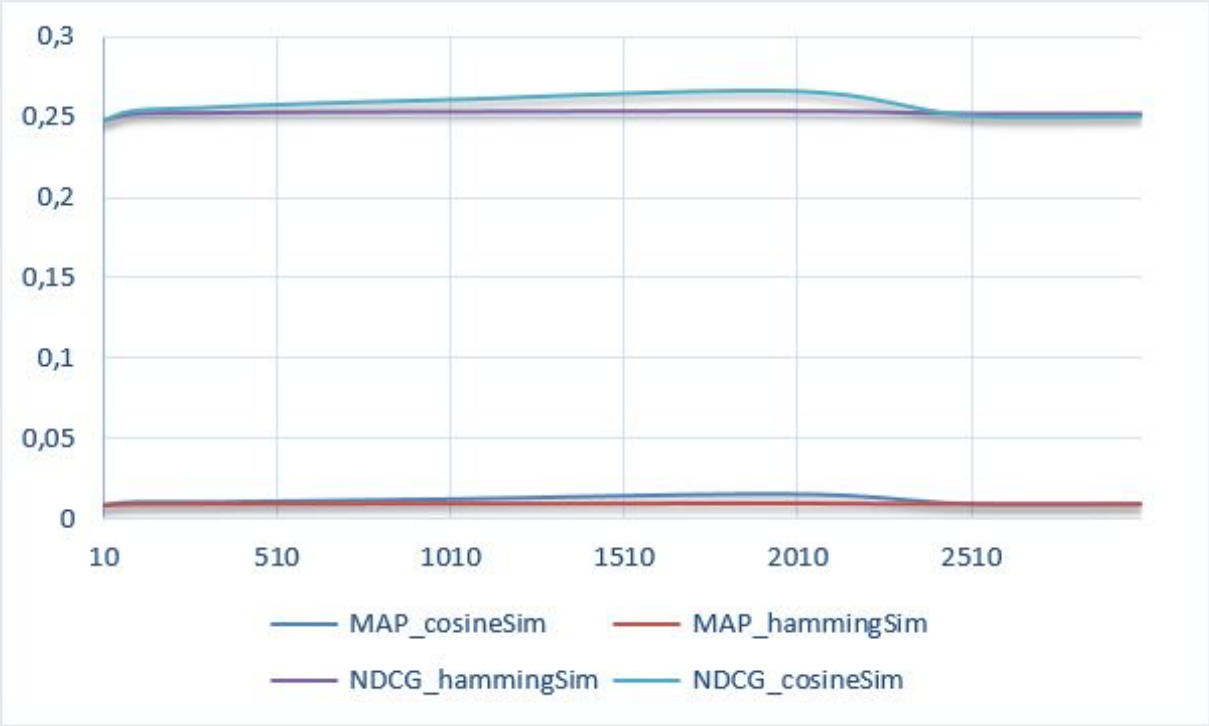
3. Set thresholds to hash the inner product of new documents(query) vectors
   - one global threshold for whole documents(query)
   - different thresholds for each document(query)

4. Similarity computation
   - cosine similarity
   - hamming similarity

RANDOM PROJECTION

Cosine similarity and Hamming similarity

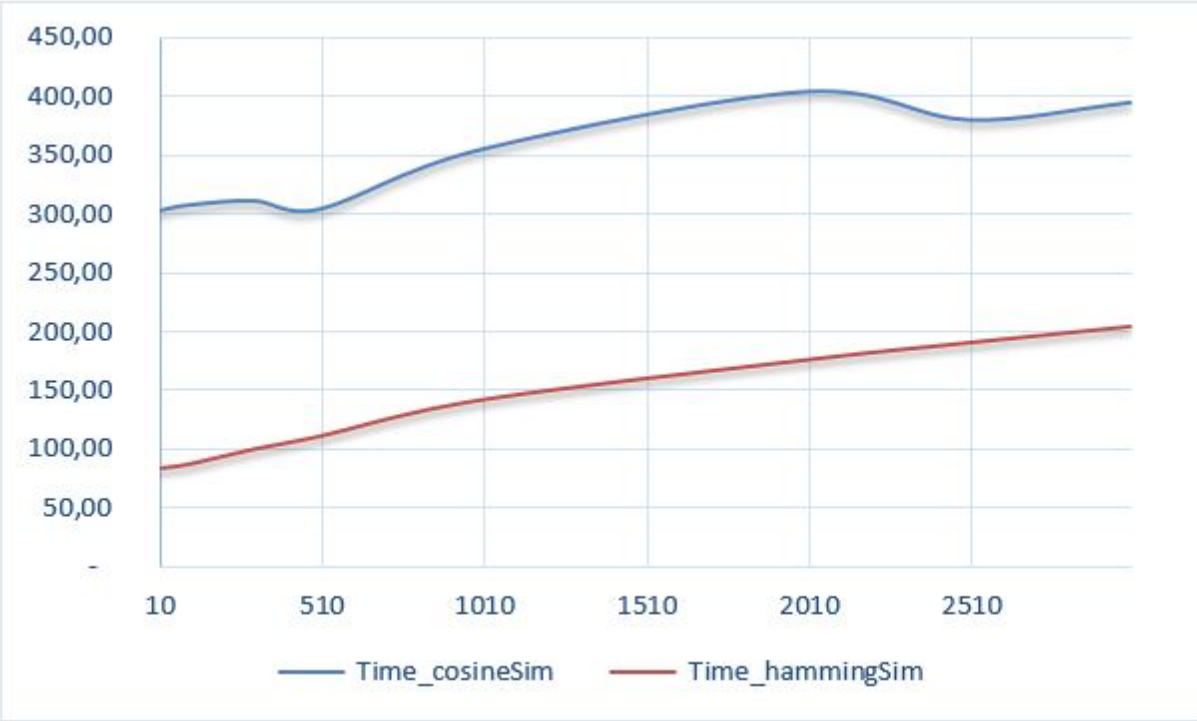
- Performance comparison of cosine similarity and hamming similarity



RANDOM PROJECTION

Cosine similarity and Hamming similarity

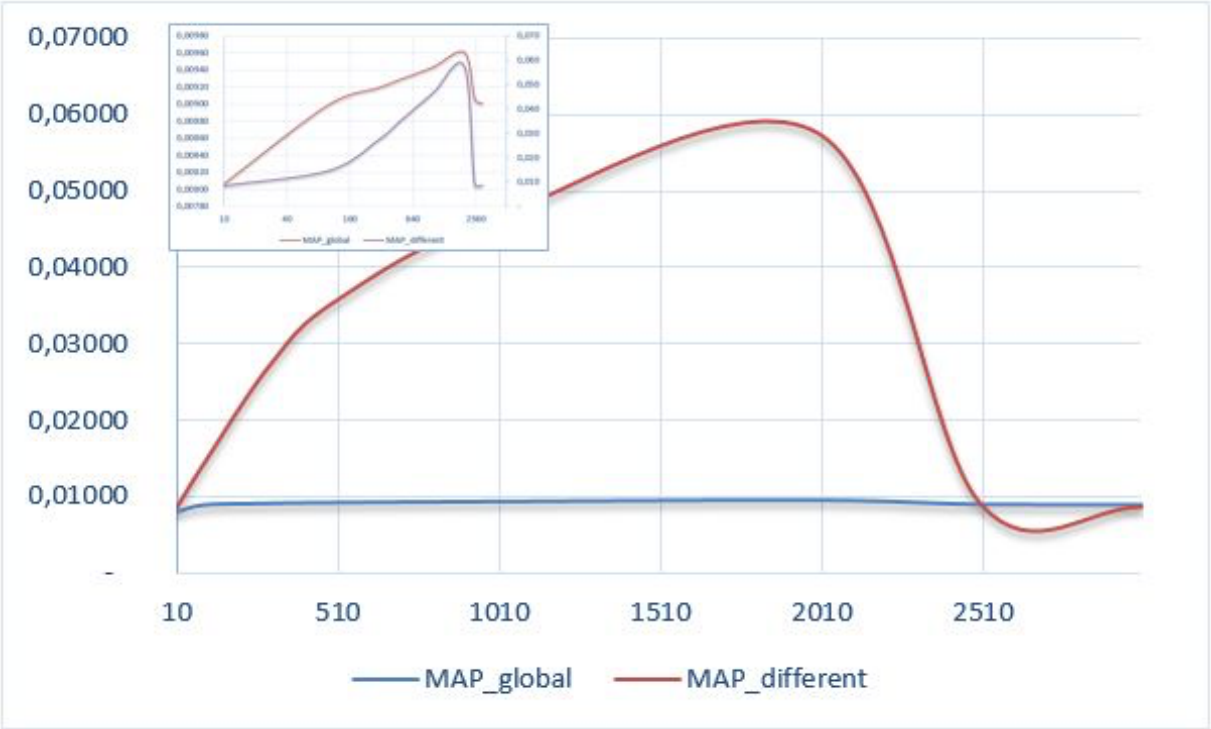
- Time comparison of cosine similarity and hamming similarity



RANDOM PROJECTION

One global threshold and Different thresholds

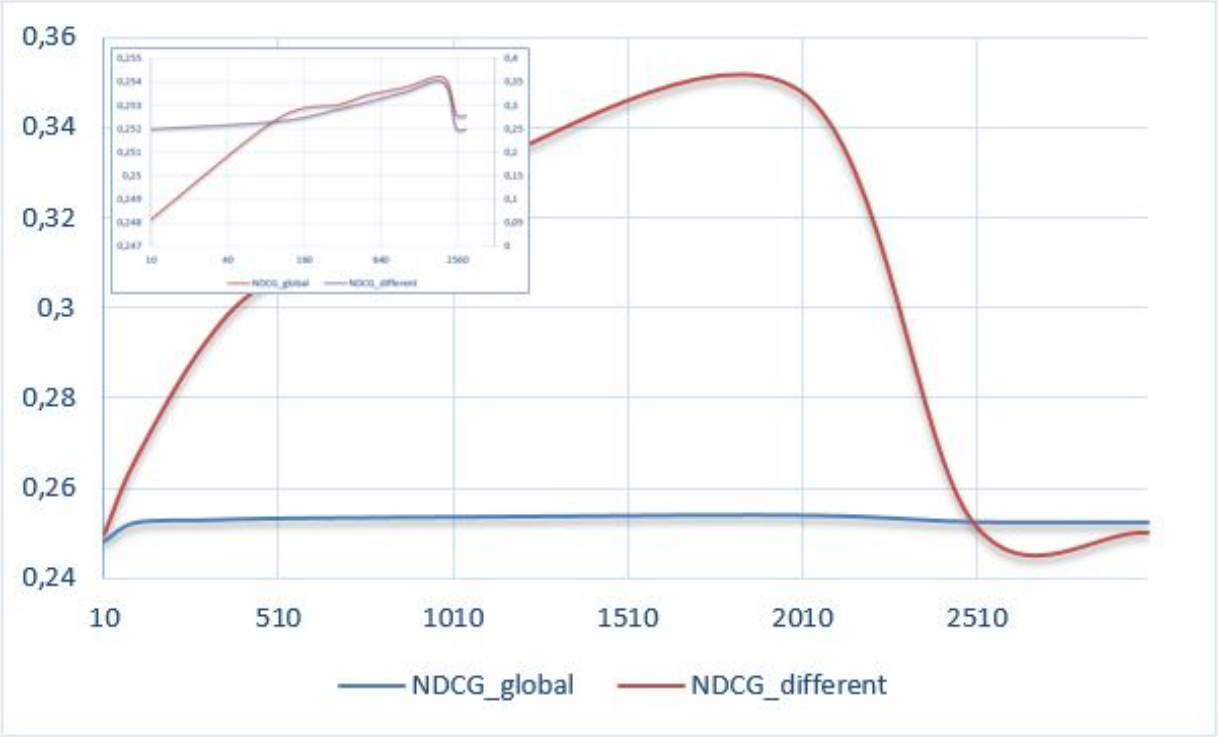
- MAP comparison of one global threshold and different thresholds



RANDOM PROJECTION

One global threshold and Different thresholds

- nDCG comparison of one global threshold and different thresholds



TIERED INDEX

1. Tiered index is the an approach which splits posting list into several tiers according to term frequency value.

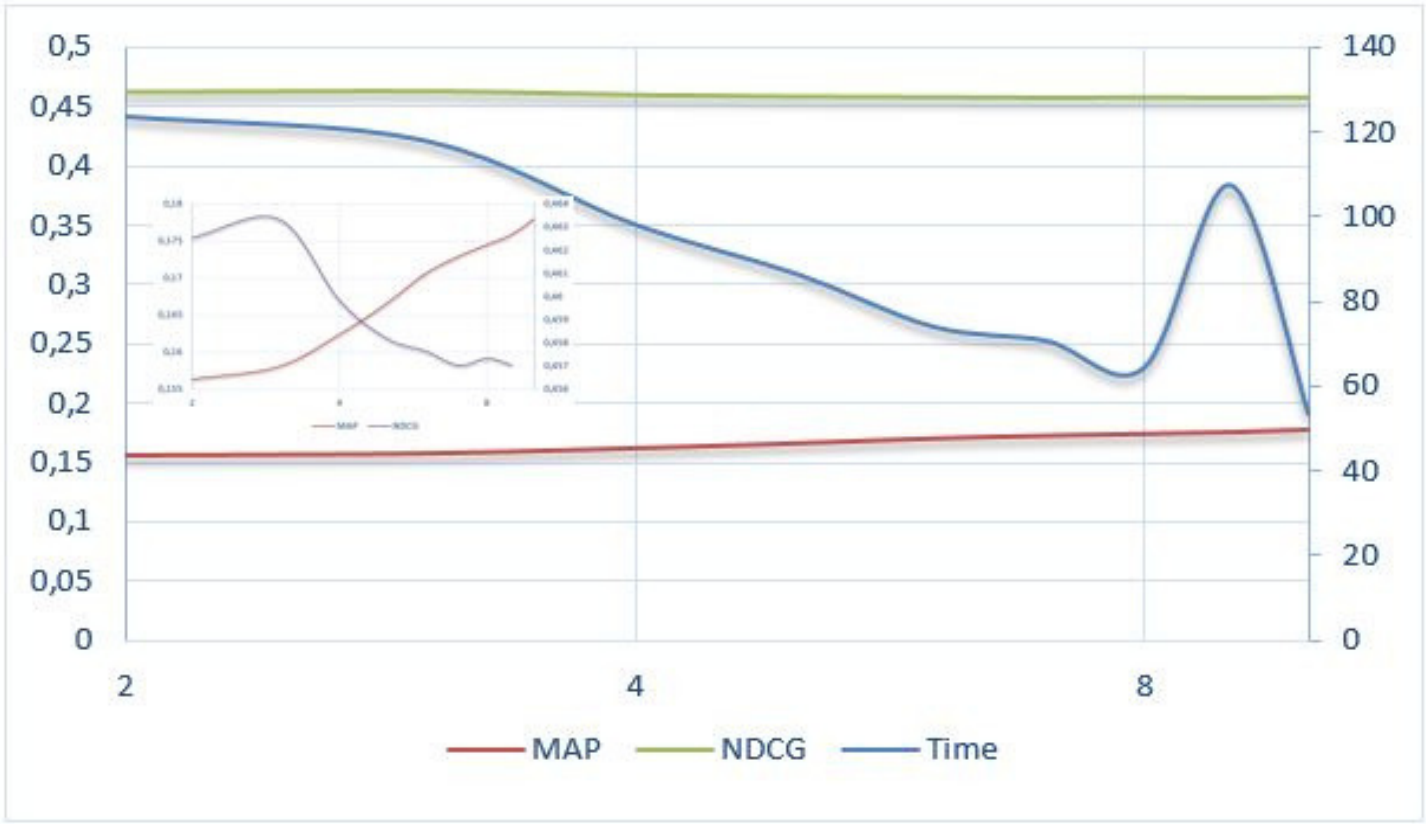
2. Posting list was split into the same length tiers.

3. Three variables:

   - Tiers number
   - Required percentage of query terms a document contains (p).
   - Top K value

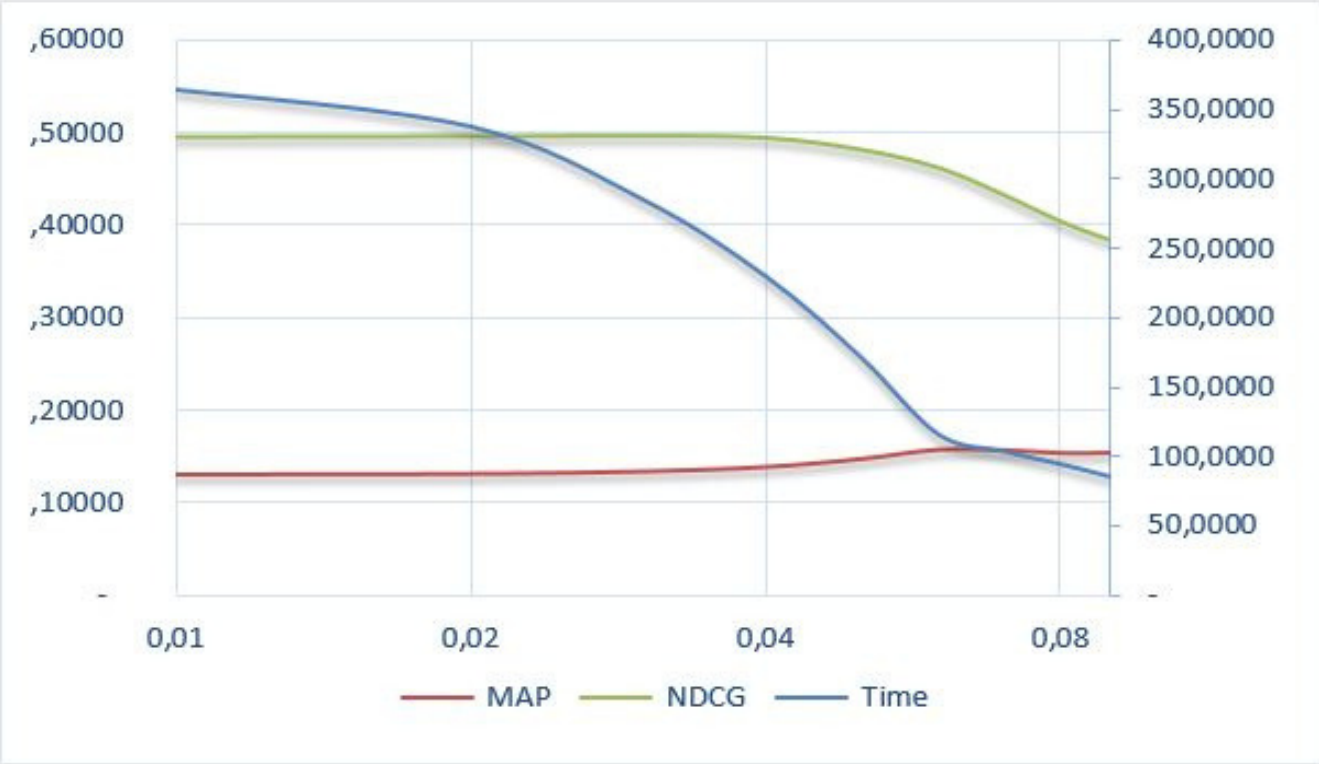
TIERED INDEX

- Influence of tiers number, K=50, p=0.06



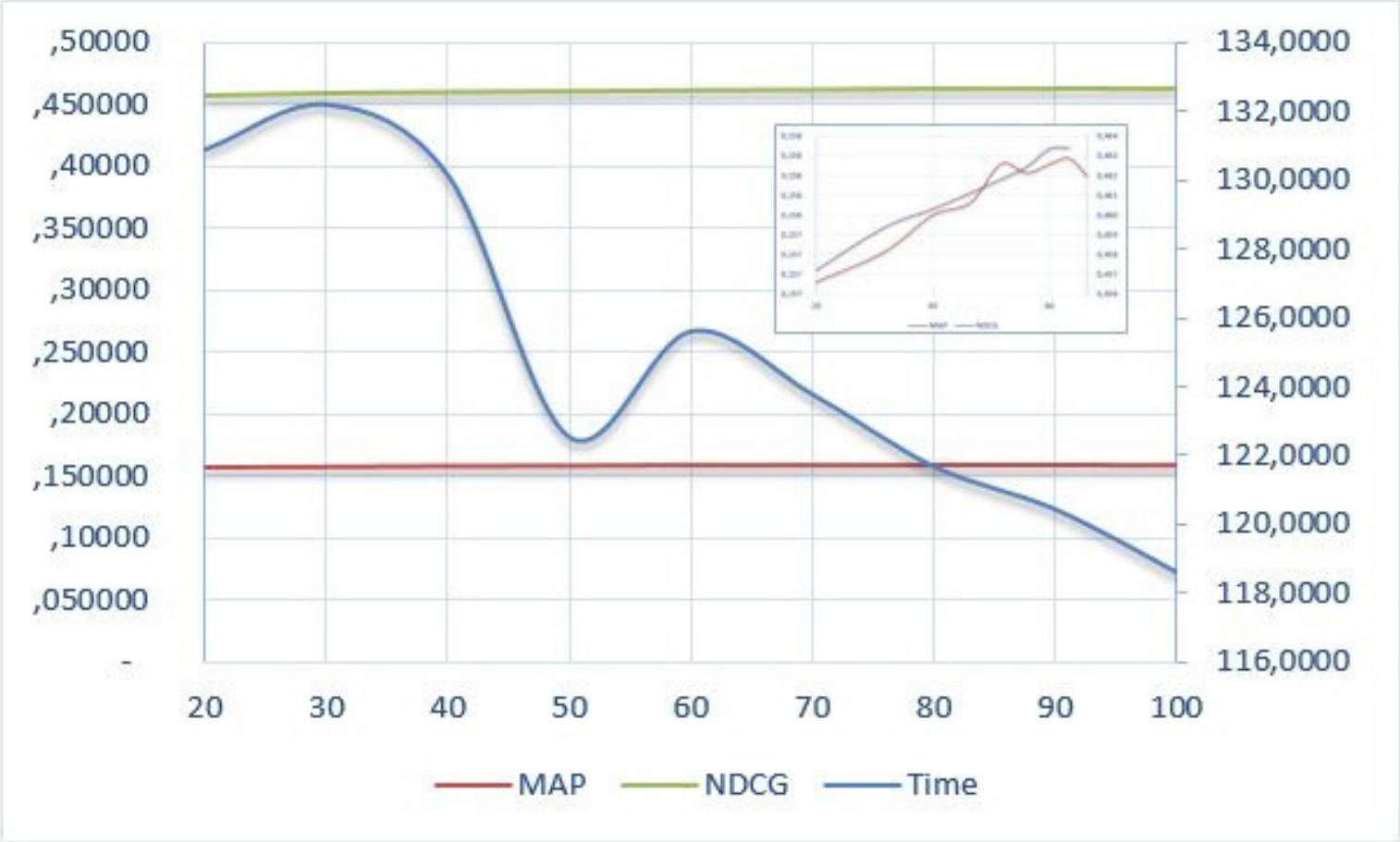
TIERED INDEX

- Influence of value p, K=50, tiers=3



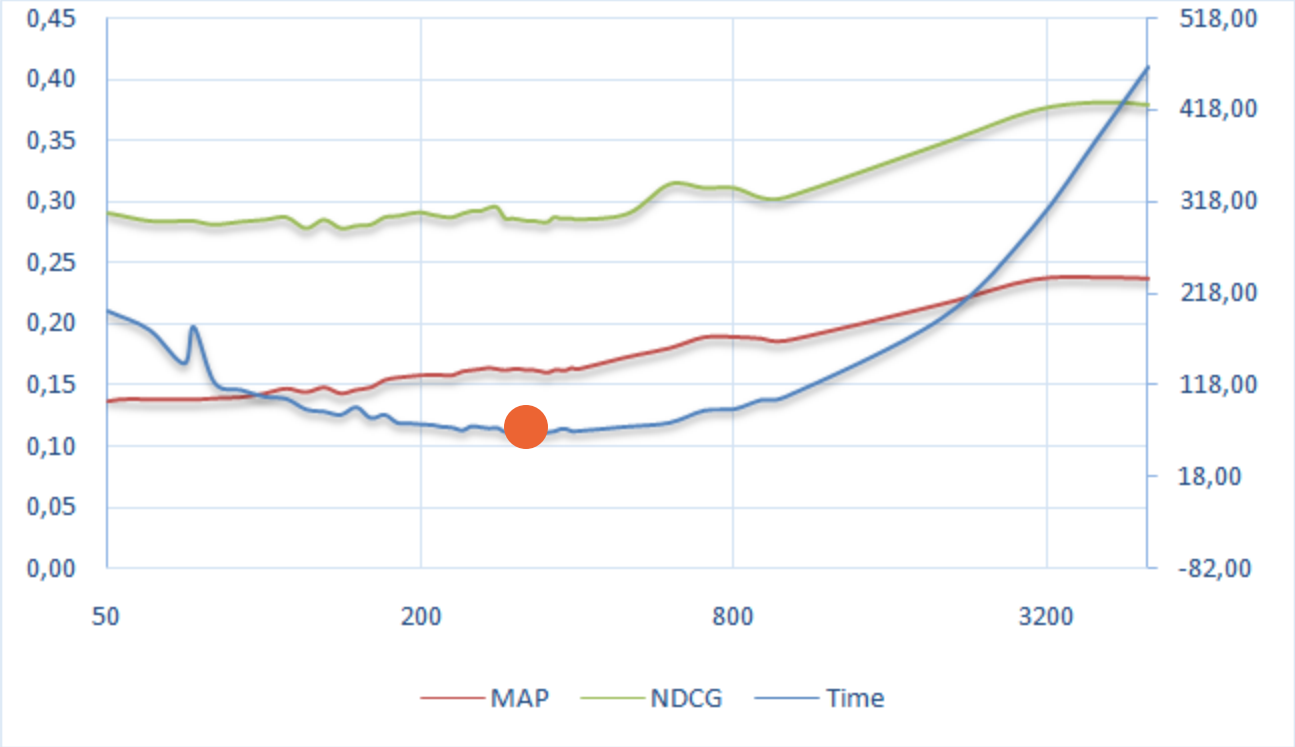
TIERED INDEX

- Influence of Top K value, tiers=3, p=0.06



PRE-CLUSTERING

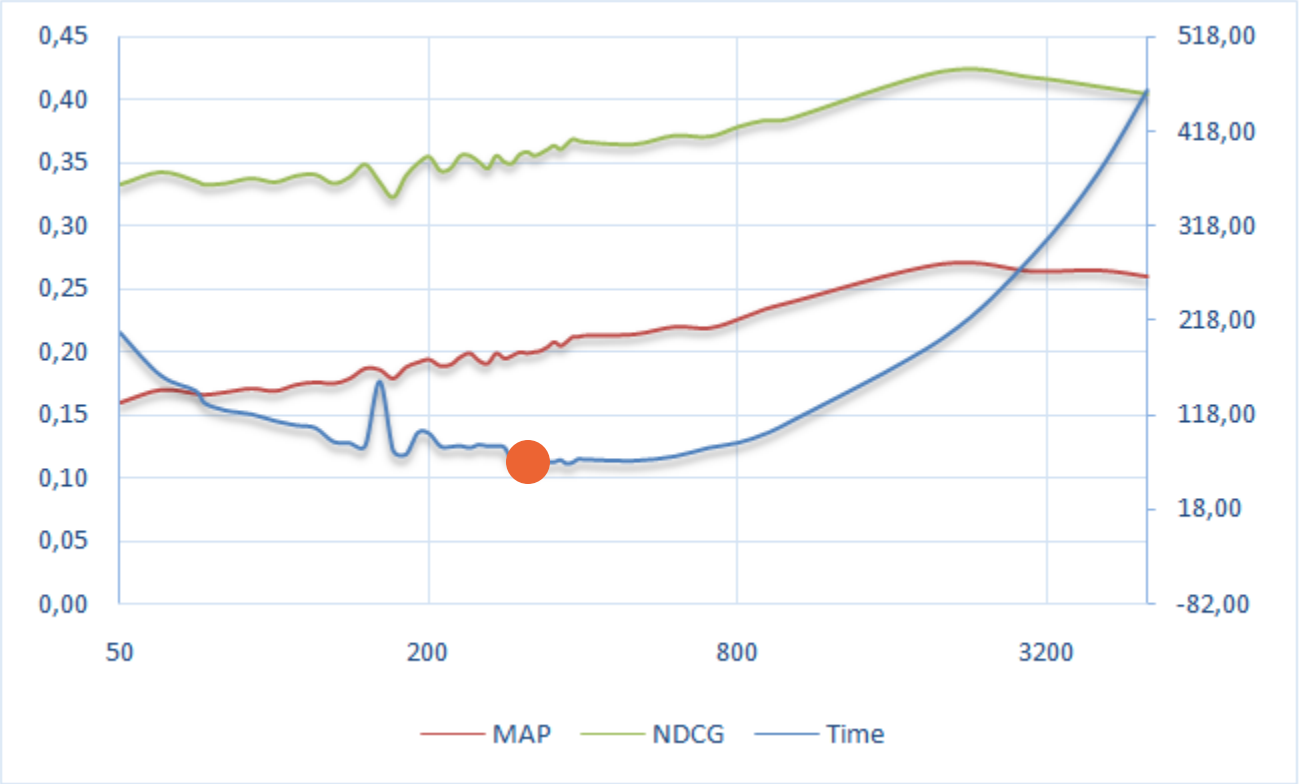
- Single-path clustering w/ 47 different leaders size ∈ √N & [50,5000]



- Trade-off: 300 leaders, 67s Time , 0.16 MAP, 0.29 nDCG

PRE-CLUSTERING

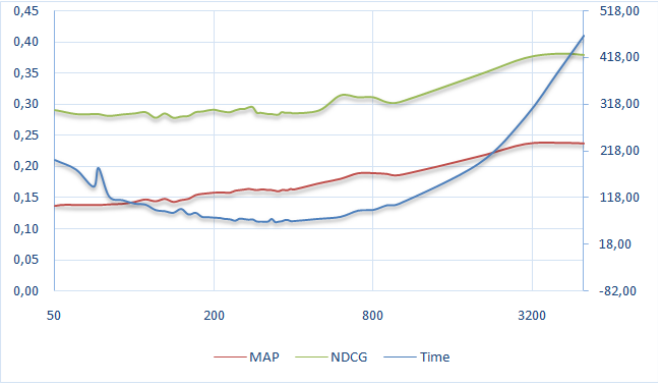
- K-means clustering w/ 47 different leaders size ∈ √N & [50,5000]

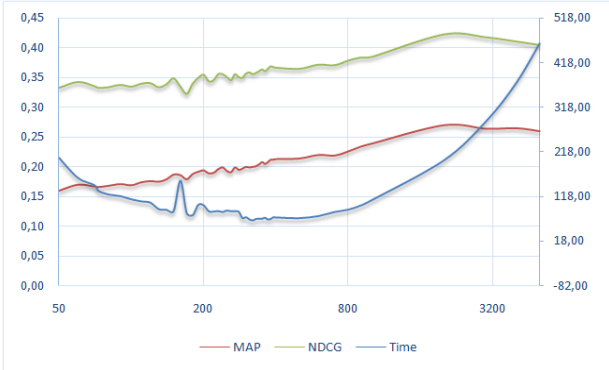


- Trade-off: 300 leaders, 71s Time , 0.20 MAP, 0.36 nDCG

PRE-CLUSTERING

- Single-path vs. K-means





avg.	Time	MAP	nDCG
Single-path	110,03	0,165	0,296
K-means	113,004	0,200	0,358
%	3%	22%	21%

AGENDA

LEARN UNDER FIRE

1. Data Structure
2. Reprocessing
3. Evaluation Metrics
4. Modeling & Results
5. Comparison Overview

COMPARISON OVERVIEW

- MAP & nDCG: 3237 queries + 5371 documents
- Time: 1 query + 5371 documents (avg. of random 200 queries)

- Basic model: TFIDF schema 0, cosines similarity
- Tiered Index:
- Pre-Clustering: K-means, 300 leaders
- Random Projection: 2000 length, different threshold, hamming

	Basic Model	Tiered	Clustering	Random Projection
MAP	0,13	0,16	0,20	0,06
MAP %	-	**26%**	**60%**	**-54%**
nDCG	0,49	0,46	0,36	0,35
nDCG %	-	**-6%**	**-27%**	**-29%**
Time	0,26	0,03	0,03	0,09
Time %	-	**-88%**	**-88%**	**-63%**

THANK YOU!

We appreciate the coaching supports you provided

GROUP 9