



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

Query-expansion Approaches to Microblog Retrieval

Sandeep Avula

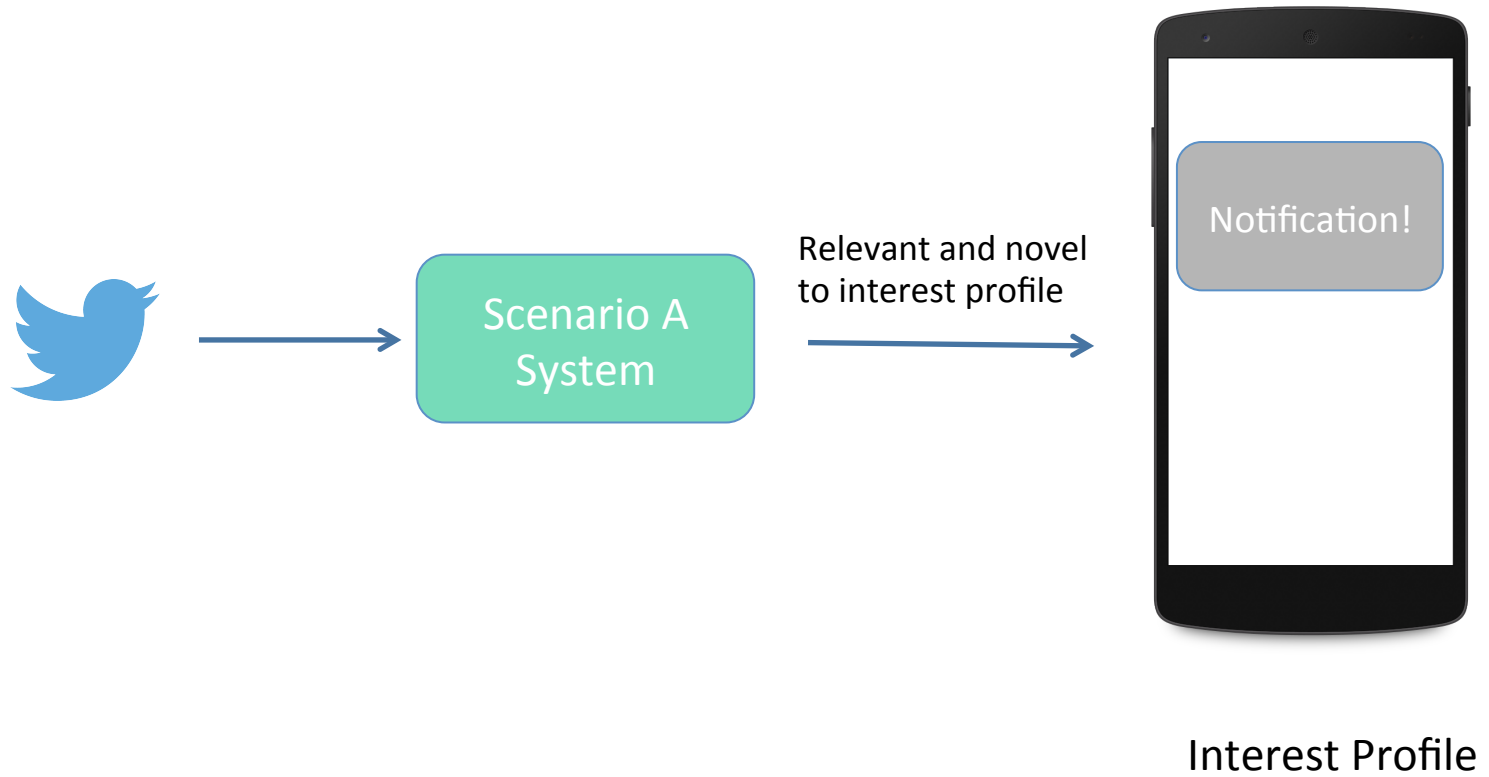
Jaime Arguello

School of Information and Library Science

University of North Carolina at Chapel Hill

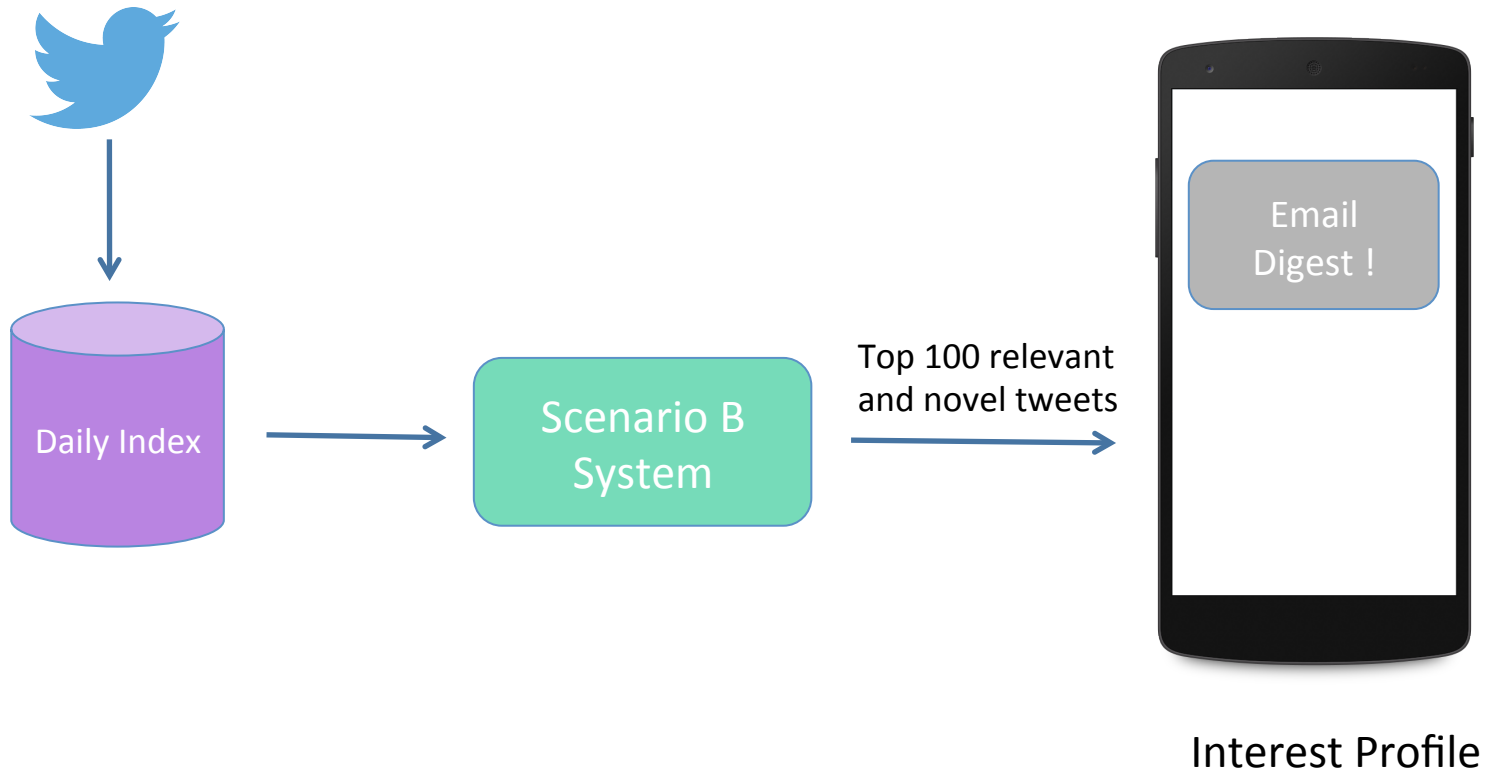
Tasks

- Scenario A



Tasks

- Scenario B



Interest Profile

225 topics were provided.

51 of were used for evaluation.

Text within the title tag was used.

Example:

<num> Number: MB339

<title>

Chincoteague Pony Swim

<desc> Description:

Find tweets about the Annual Pony Swim in Chincoteague, MD.

<narr> Narrative:

The user is attending the 2015 Pony Swim and auction in Chincoteague, MD.

Any information regarding attendance, logistics, entertainment, accommodations, and food is welcome.

Scenario B

Objective: Identify at most 100 ranked relevant and novel tweets per interest profile per day.

Run types:

- Automatic Run.
- Manual Preparation.
- Manual Intervention.

Scenario B

Objective: Identify at most 100 ranked relevant and novel tweets per interest profile per day.

Run types:

- Automatic Run.
- ~~Manual Preparation.~~
- ~~Manual Intervention.~~

Challenges with Tweets

- Terse.
- Lot of metadata surrounding the text.

Possible approaches:

- Enrich the tweet.
- Enrich the query.

Challenges with Tweets

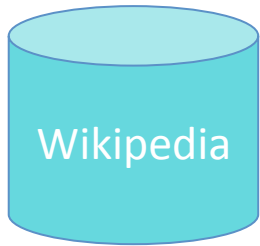
- Terse
- Lot of metadata surrounding the text.

Possible approaches:

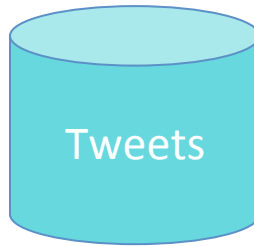
- ~~Enrich the tweet.~~
- Enrich the query.

Corpus

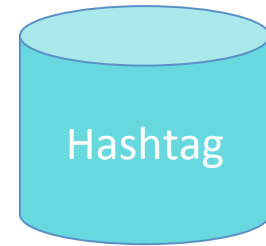
We expanded the query using the following corpora:



We indexed* each Wikipedia document to create the corpus



Tweets harvested for **20 days** before the evaluation period. Each tweet that passed our filtering phase was indexed.



We created **hashtag pseudo documents** from the tweets we collected for **20 days** prior to the evaluation period.

* Indexed using Lucene and also removed the stop words and stemmed the text.

Example of the Hashtag psuedo-documents

Suppose that there were only 3 tweets.

Tweet1: Today will be an awesome day **#yolo #bright #happy**

Tweet2: I love this song <song url>! Can't stop listening! **#yolo #happy**

Tweet3: Did well in the exam! **#happy #bright**

yolo:

Today will be an awesome day

I love this song <song url>! Can't stop listening!

bright:

Today will be an awesome day

Did well in the exam!

happy:

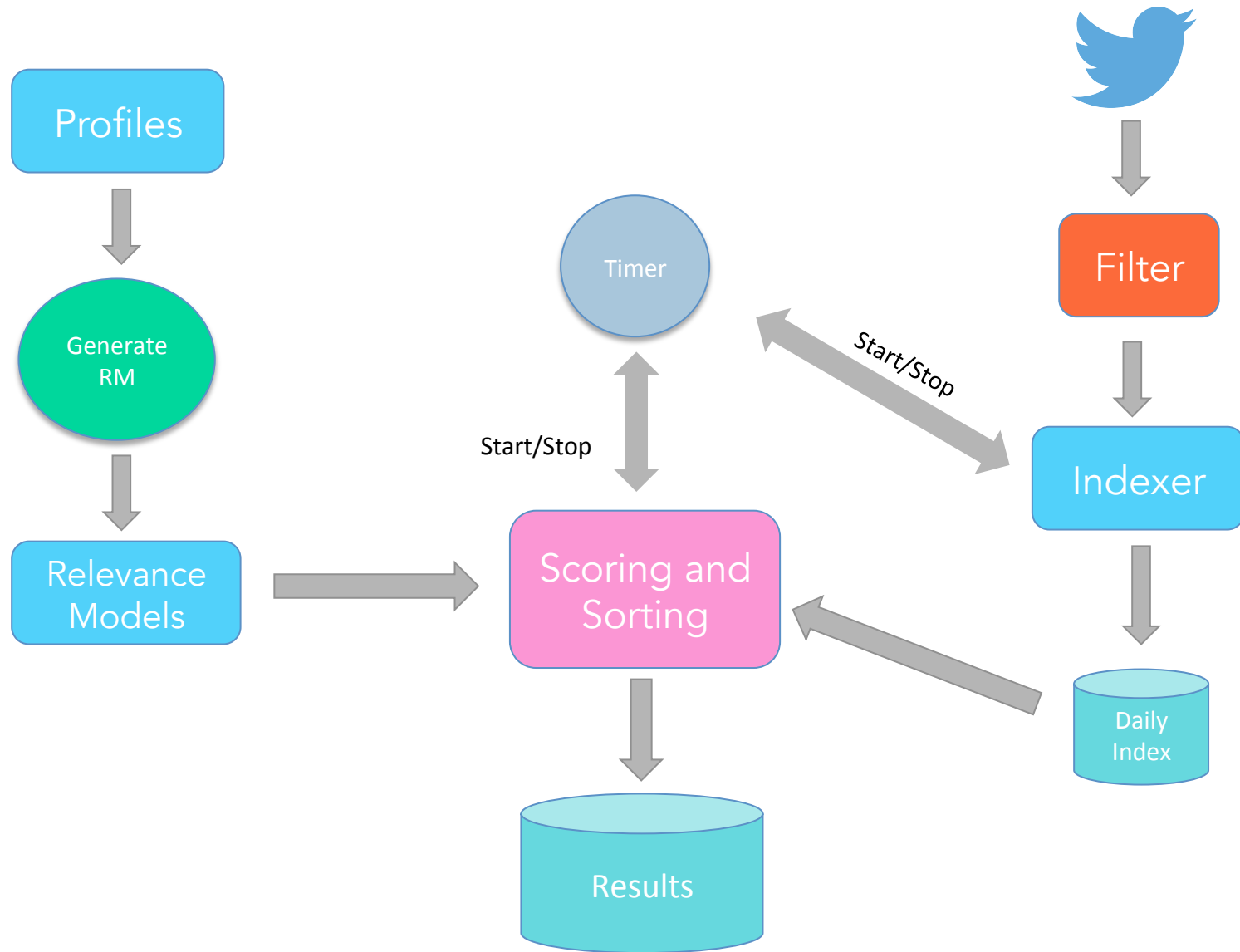
Today will be an awesome day

I love this song <song url>! Can't stop listening!

Did well in the exam!

Hashtag
pseudo-documents

System Architecture



Standard Query Likelihood Model

(Smoothed with a Dirichlet prior)

For a query Q and a Document D , the query likelihood model is as follows:

$$\text{score}(Q, D) = \prod_{w \in \mathcal{V}} P(w|\hat{\theta}_D)^{P(w|\theta_Q)},$$

$$P(w|\hat{\theta}_D) = \frac{\#(w, D) + \mu P(w|\theta_C)}{|D| + \mu}$$

$$P(w|\theta_Q) = \frac{\#(w, Q)}{|Q|}$$

$$P(w|\theta_C) = \frac{\#(w, D)}{|C|}$$

$$\text{score}(Q, D) = \sum_{w \in \mathcal{V}} P(w|\theta_Q) \log(P(w|\theta_D)).$$

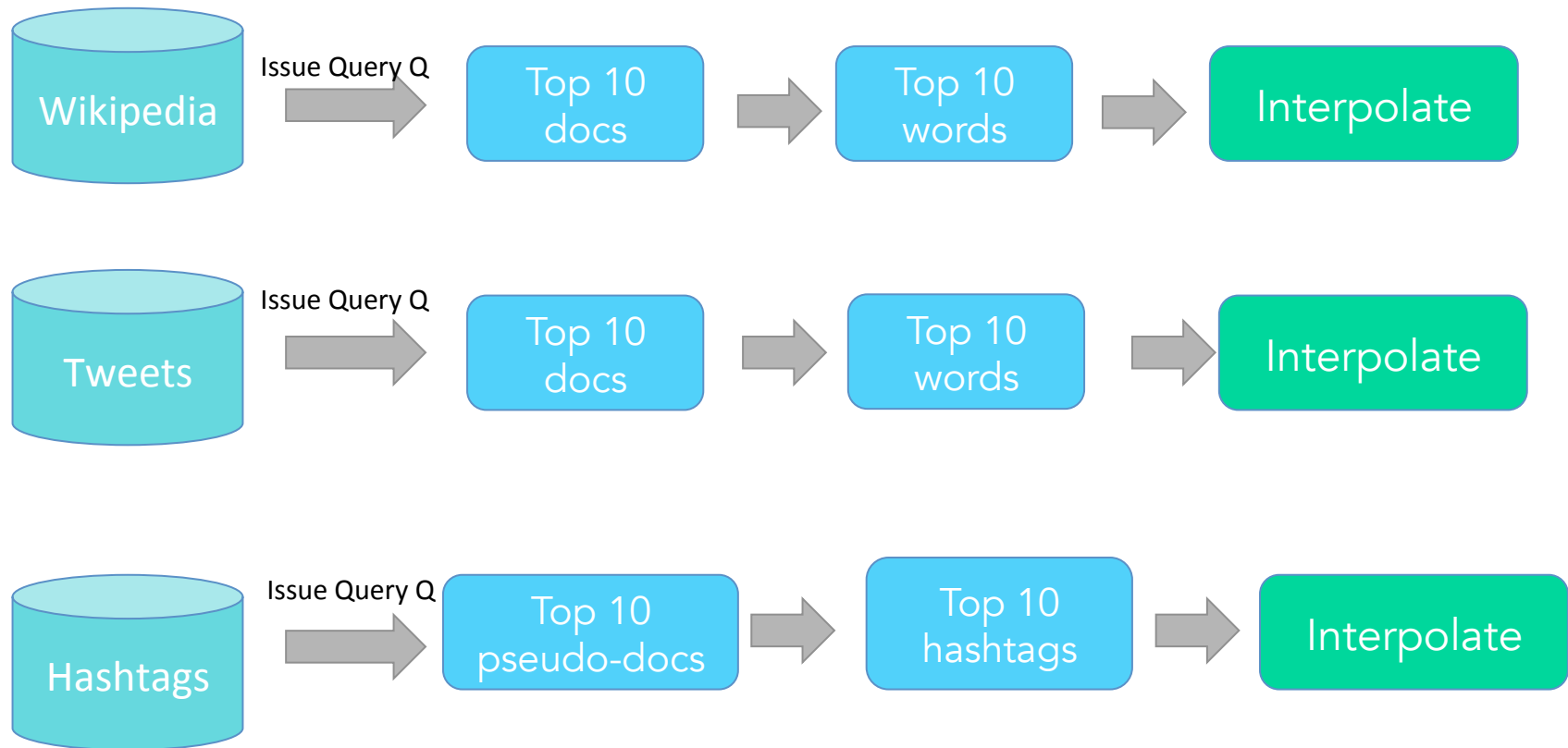
$\#(w, X)$: Denotes the number of times w occurs in X

$|X|$ denotes the number of term occurrences in X .

\mathcal{V} denotes the entire vocabulary of the collection

μ is the dirichlet prior set to 1000.

Continuation



Relevance Models for Wikipedia and Tweets

Similar technique was applied on both the Wikipedia and the tweets corpus.

$$Score(Q, D) = \sum_{w \in \mathcal{V}} P(w|\hat{\theta}_R) \log P(w|\hat{\theta}_D)$$

Relevance Models for Wikipedia and Tweets

Similar technique was applied on both the Wikipedia and the tweets corpus.

$$Score(Q, D) = \sum_{w \in \mathcal{V}} \boxed{P(w|\hat{\theta}_R)} \log P(w|\hat{\theta}_D)$$

Relevance Models for Wikipedia and Tweets

Similar technique was applied on both the Wikipedia and the tweets corpus.

$$Score(Q, D) = \sum_{w \in \mathcal{V}} \boxed{P(w|\hat{\theta}_R)} \log P(w|\hat{\theta}_D)$$

$$P(w|\hat{\theta}_R) = \lambda P(w|\theta_Q) + (1 - \lambda)P(w|\theta_R)$$

Relevance Models for Wikipedia and Tweets

Similar technique was applied on both the Wikipedia and the tweets corpus.

$$Score(Q, D) = \sum_{w \in \mathcal{V}} \boxed{P(w|\hat{\theta}_R)} \log P(w|\hat{\theta}_D)$$

$$P(w|\hat{\theta}_R) = \lambda P(w|\theta_Q) + (1 - \lambda) \boxed{P(w|\theta_R)}$$

Relevance Models for Wikipedia and Tweets

Similar technique was applied on both the Wikipedia and the tweets corpus.

$$\text{Score}(Q, D) = \sum_{w \in \mathcal{V}} \boxed{P(w|\hat{\theta}_R)} \log P(w|\hat{\theta}_D)$$

$$P(w|\hat{\theta}_R) = \lambda P(w|\theta_Q) + (1 - \lambda) \boxed{P(w|\theta_R)}$$

$$P(w|\theta_R) = \frac{1}{\mathcal{Z}} \sum_{D \in \mathcal{R}_t} P(w|\theta_D) \times \text{score}(Q, D)$$

$$\mathcal{Z} = \sum_{D \in \mathcal{R}_t} \text{score}(Q, D)$$

Relevance Models for Hashtags

For hashtags, we approached this slightly differently.

$$Score(Q, D) = \sum_{w \in \mathcal{V}} P(w | \hat{\theta}_Q^{hash}) \log P(w | \hat{\theta}_D)$$

Relevance Models for Hashtags

For hashtags, we approached this slightly differently.

$$Score(Q, D) = \sum_{w \in \mathcal{V}} \boxed{P(w | \hat{\theta}_Q^{hash})} \log P(w | \hat{\theta}_D)$$

Relevance Models for Hashtags

For hashtags, we approached this slightly differently.

$$Score(Q, D) = \sum_{w \in \mathcal{V}} \boxed{P(w|\hat{\theta}_Q^{hash})} \log P(w|\hat{\theta}_D)$$

$$P(w|\hat{\theta}_Q^{hash}) = \lambda P(w|\theta_Q) + (1 - \lambda)P(w|\theta_Q^{hash})$$

Relevance Models for Hashtags

For hashtags, we approached this slightly differently.

$$Score(Q, D) = \sum_{w \in \mathcal{V}} P(w | \hat{\theta}_Q^{hash}) \log P(w | \hat{\theta}_D)$$

$$P(w | \hat{\theta}_Q^{hash}) = \lambda P(w | \theta_Q) + (1 - \lambda) P(w | \theta_Q^{hash})$$

Relevance Models for Hashtags

For hashtags, we approached this slightly differently.

$$Score(Q, D) = \sum_{w \in \mathcal{V}} P(w | \hat{\theta}_Q^{hash}) \log P(w | \hat{\theta}_D)$$

$$P(w | \hat{\theta}_Q^{hash}) = \lambda P(w | \theta_Q) + (1 - \lambda) P(w | \theta_Q^{hash})$$

$$P(h | \theta_Q^{hash}) = \frac{1}{Z} \prod_{q \in Q} \frac{\#(q, H) + \mu P(q | \theta_{C_H})}{|H| + \mu}.$$

Relevance Models for Hashtags

$$Score(Q, D) = \sum_{w \in \mathcal{V}} P(w|\hat{\theta}_Q^{hash}) \log P(w|\hat{\theta}_D)$$

$$P(w|\hat{\theta}_Q^{hash}) = \lambda P(w|\theta_Q) + (1 - \lambda) P(w|\theta_Q^{hash})$$

$$P(h|\theta_Q^{hash}) = \frac{1}{\mathcal{Z}} \prod_{q \in Q} \frac{\#(q, H) + \mu P(q|\theta_{C_H})}{|H| + \mu}.$$

H is the pseudo document associated with the hashtag h .

C_H denotes the collection of pseudo documents.

\mathcal{Z} is the normalizing coefficient.

\mathcal{V} denotes the entire vocabulary of the collection

The query enrichment in this procedure was through the hashtags and not the words in the documents as was in the other approaches.

Inclusion Criteria

- English tweets.
- Tweets without swear words.
- Tweets with at least one URL or hashtag.
- Tweets with no more than three hashtags, one URL or one user mention.
- Tweets with at least 30% non-stopwords.

Duplication

- Novelty an important criteria for evaluation.
- Removed duplicate tweets.
- Duplicate defined as having a similarity ≥ 0.7 when using the Jaccard Coefficient to measure the similarity.
- Between duplicate tweets, the tweet with a higher score was put in the list.

Results

Evaluation Metric: nDCG

Run tags:

Hashtag relevance model.

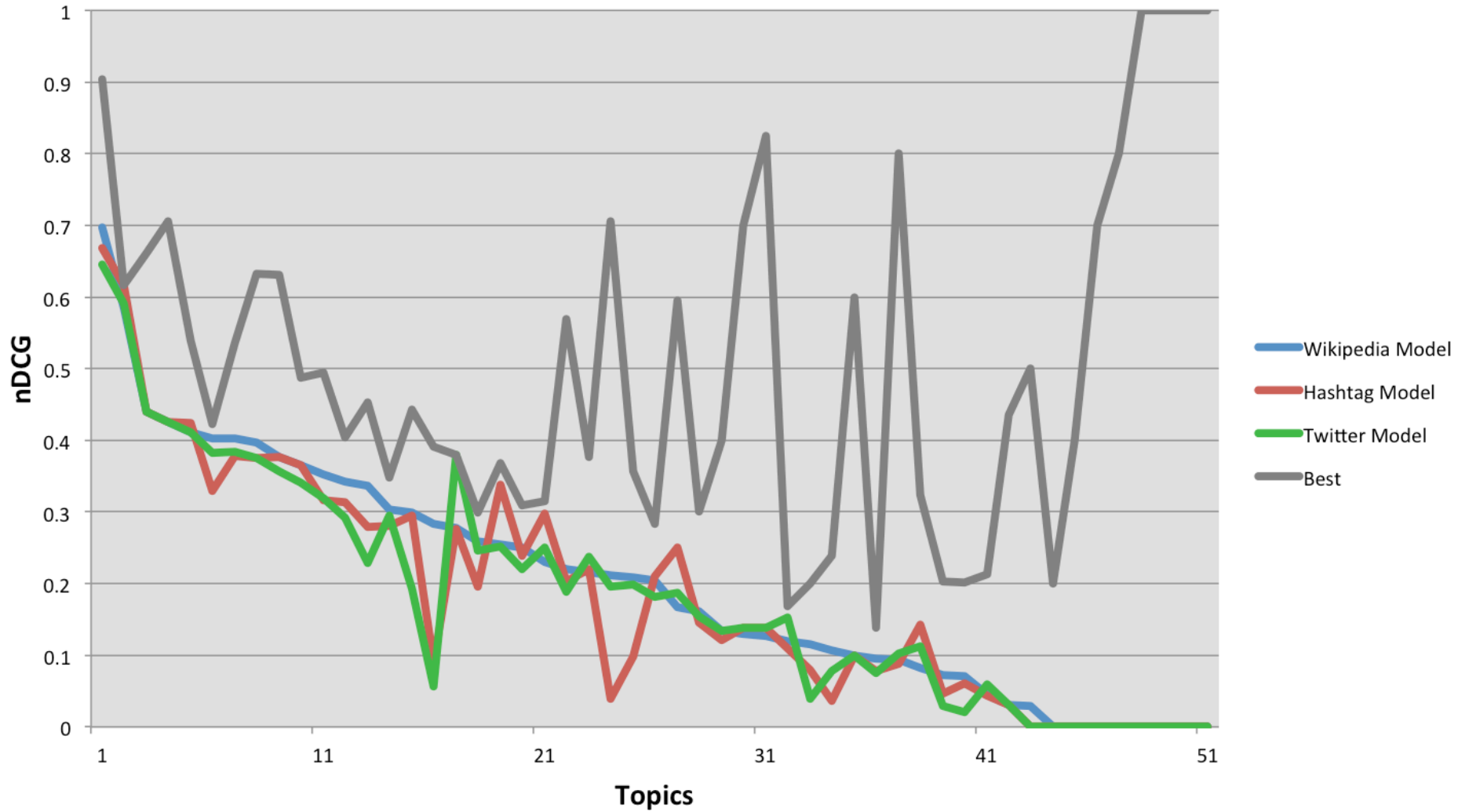
Tweet relevance model.

Wikipedia relevance model.

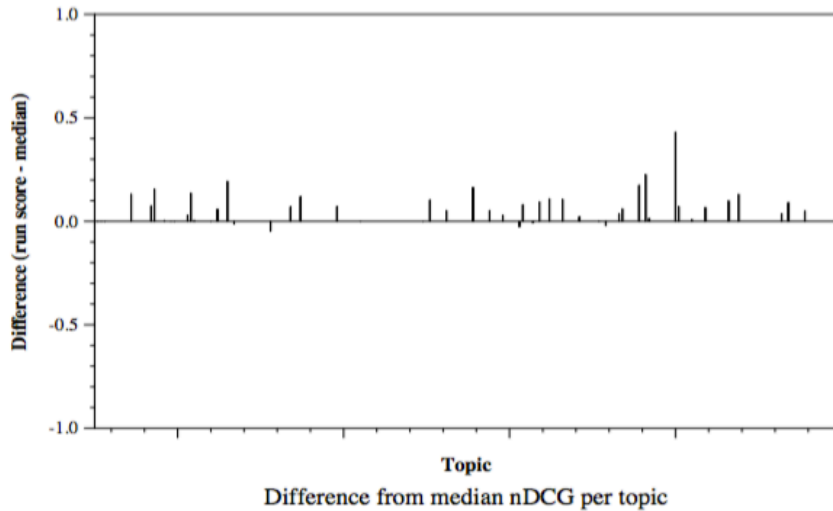
Hashtag Model	Tweet Model	Wiki Model	Best
0.1902	0.1890	0.2045 *	0.50142

* $p < 0.05$

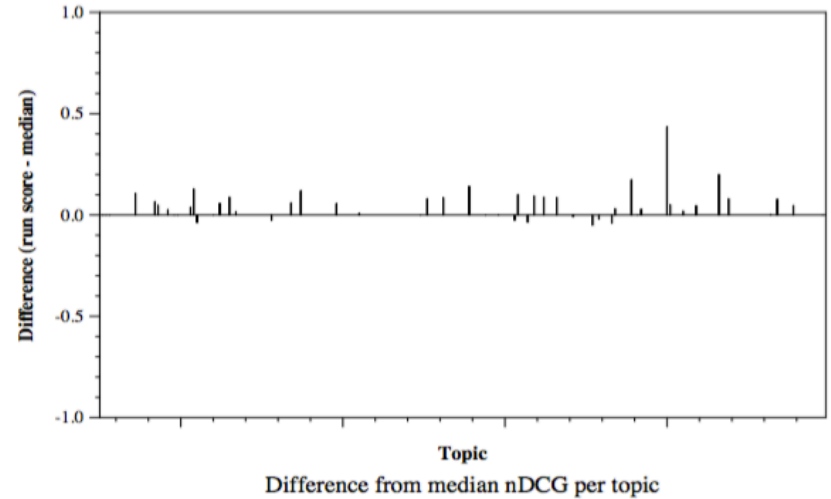
Results



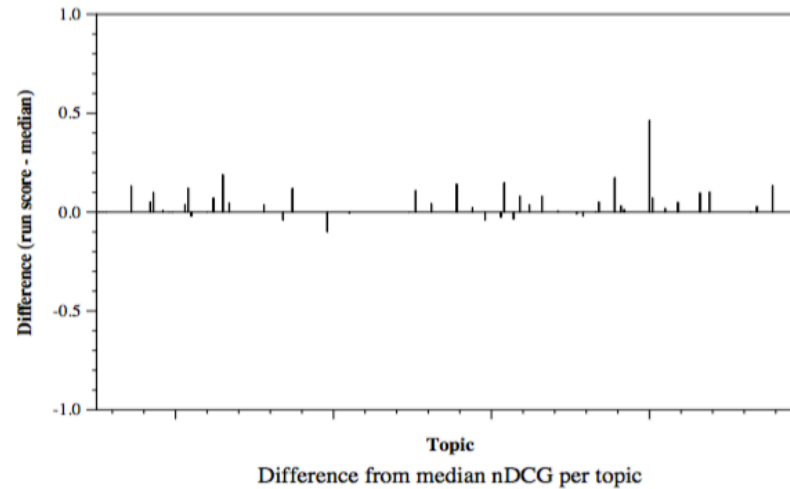
Results



Wikipedia Model



Twitter Model



Error Analysis

- Did we filter aggressively?
- Did we manage novelty efficiently?
- Likelihood model the right approach?

Error Analysis

- Did we filter aggressively? – Yes.
- Did we manage novelty efficiently? – No.
- Likelihood model the right approach? – Not yet sure.

Error Analysis

- Percentage of tweets we missed in the cluster: 40% (approx.)
- We filtered duplicate tweets within the same day, but not across days
- Correlation between the Wikipedia model scores per topic to the size of clusters for the topics: 0.593

Error Analysis

nDCG was 0, 8 times for each of the three runs.

Average MaxNDCG when our nDCG was 0: 0.7625

Queries that worked and those that didn't

Worked	Didn't Work
Iran nuclear agreement	summer Seasonal Affective Disorder (SAD)
"The Vatican Tapes" movie	Hershey, PA quilt show
Stephen Colbert Late Show	Indian-Pacific train

Conclusion

- Tried three approaches to enrich the query.
- Each approach corresponded to using three different corpora to enrich the query.
- The Relevance model that used pseudo-relevance feedback from the Wikipedia worked best.
- Filtered aggressively.
- Did not manage duplicates effectively.

Thank you !

Questions?