Distributed Text Representations for Information Retrieval

Information Retrieval 1: Homework 2

Christoph Hönes
University of Amsterdam
Stud. nr.: 12861944

Tamara Czinczoll
University of Amsterdam
Stud. nr.: 12858609

Yke Rusticus
University of Amsterdam
Stud. nr.: 11306386

1 INTRODUCTION

With an ever growing expansion of the Web, the variety of information increases. Common information retrieval models such as TF-IDF or MB25 fail to catch semantics of documents, with the result that useful grammatical relations like synonymy and polysemy are ignored. In this work, we compare several models which represent text as dense vectors, and are therefore able to address these shortcomings. The first two models we take into account are Word2Vec and Doc2Vec [3–5]. These models rely on learned word embeddings that are aggregated to form the representation of a document. Then, we evaluate two variations on the LSI model [2], and finally the LDA model [1]. These models rely on the assumption that documents consist of a mixture of (latent) topics, which, when modeled, can be used for information retrieval. We trained and evaluated each model on the Associated Press dataset, which is a dataset in the news domain[1]. In the following section, we answer some relevant theoretical questions. In Section 3, we further describe the details of each model and their implementation for this work. We shortly touch on the experimental setup in Section 4, after which we present and discuss the results in the following sections.

2 THEORETICAL QUESTIONS

TQ1.1

We are using the Associated Press dataset, which is a dataset in the News domain. If we were using a dataset in the medical domain, what pre-processing steps would you include / exclude? Justify your choice.

For the news domain you want to retrieve highly relevant documents and do not care much about the others. For the medical domain, it might be an advantage to also retrieve only slightly relevant documents. This will make sure that no critical information is missed and the right diagnosis can be found, even for rare diseases. As pre-processing steps tokenization still needs to be applied and stopwords can still be removed, since they add noise. However, stemming should be excluded since many medical words have complex names and, e.g. for proteins stemming could lead to false matches between unrelated proteins. Furthermore, phrases should also be tokenized so that complex medical compound words can still be matched as a whole.

TQ2.1

Word2Vec, and word embedding models in general, might might not be suitable for retrieval. Why do you think this is?

To represent a document as vector, we aggregate the vectors (word embeddings) of all words that are present in the document. However, a document can contain many words that generally appear in different contexts. The direction of the final vector representation of the document could therefore only represent the average (or weighted average) of all the contexts in the document, with the result that it might not lie close to the context of a query, to which the document is actually relevant. That means that the cosine similarity between that document and the query could be low, even though the document is relevant. Furthermore, the aggregation of the word vectors to form a document vector are usually simple averaging operations. They do not take word order into account and thus possibly miss meaning.

TQ2.2

Comment on the key differences between methods you encountered in the previous assignment (TF-IDF, BM25) and Doc2Vec / Word2Vec.

In TF-IDF and BM25 only word counts are taken into account (in more or less complex ways), no semantics. The systems are not able to realize that *Netherlands* and *Holland* are both used to refer to the same entity and would thus not be able to match documents about the Netherlands to a query asking for Holland. In Doc2Vec and Word2Vec, context is learned and words are represented as dense vectors in a semantic space, with synonyms having very similar vector representations. Thus, a word embedding model would also correctly retrieve documents about the Netherlands. This example about synonyms holds for all grammatical relations for which the words occur in similar context.

TQ3.1

Explain the differences and similarities of LDA and LSA. Which one is more suitable for IR in your opinion?

LDA and LSA are both topic models that derive or learn a given number of topics from a set of documents. They both rely on the assumption that each document consists of a mixture of topics and that each topic consists of a set of words. The topics are so called latent, unobserved variables. With LSA, topics are derived by essentially compressing the combined document-term matrix (in which each term is given a score corresponding to the document in which it occurs) by using techniques from linear algebra (SVD) and approximations using matrix eigenvalues. With LDA, topics are learned from the data by repeatedly sampling from and updating Dirichlet distributions that describe the mixture of topics within documents and the mixture of words within topics. The difference in methods has the result that LDA generally scales well to new/unseen documents, whereas for LSA the whole topic representation needs to be re-calculated if too many unseen documents are taken into account. Therefore, we expect LDA to be the better choice for information retrieval.

[1] https://trec.nist.gov/data/docs_eng.html

3 MODELS AND IMPLEMENTATION

3.1 Word2Vec

The training objective of Word2Vec with negative sampling is to predict whether a word is a valid context word of the given center word. To generate the training data, we sample documents from the dataset and generate the word pairs. We generate three times as many negative word pairs by randomly picking a word from the entire vocabulary that is not an actual context word. The probability to pick a word is uniformly distributed. During training, we feed the data in batches using the nn.Embedding pytorch module. We feed the center words into one Embedding matrix and the context words into another. Then, we take the dot product of the two and apply the sigmoid activation function. The result is fed into the binary cross-entropy loss. To calculate the most similar words for a given word, its word vector is compared to all other vectors in the vocabulary using the cosine similarity. The vectors are then sorted according to highest similarity.

3.2 Doc2Vec

Doc2Vec represents entire documents as a single vector. It uses a more sophisticated technique than simple averaging over all word vectors (as done in the Word2Vec section). Instead, it trains an additional vector alongside word vectors that encodes the information about the document.

3.3 Latent Semantic Indexing/Analysis (LSI/LSA)

Latent Semantic Indexing (LSI) or sometimes also called *Latent Semantic Analysis (LSA)* is a kind of topic model that creates a low-rank approximation of the term-document matrix by using *Singular Value Decomposition (SVD)*. The k principal singular values of the $N \times M$ term-document matrix then represent the k topics (where N is the number of words in the vocabulary and M is the number of documents). We choose the initial number of topics for the AP corpus to be 500 as a default value but we will study the effect of different k-values on the model's performance in our experiments. From now on when we refer to default LSI models we mean that the model is trained with 500 topics. There are different ways to represent the term-document matrix. In our first LSI model we use *Bag of Words (BoW)* vectors that contain the raw term frequencies (counts) of the words within a document. These vectors are of size N and represent the M columns of our term-document matrix. Another approach uses the *term frequency inverse document frequency* (TF-IDF) scores of the words as term-document matrix. To implement these models we make use of the gensim python library which provides an API to train a LSI model. We remove all the rows from the term-document matrix where the word occurs in less than 50 documents as there are likely to be spelling mistakes or very unusual expressions. Additionally, we consider words that occur in more than 50% of the documents as uninformative and remove them as well. This pre-processing is used for our other models as well. The documents in the corpus as well as the queries can be projected in the LSI space and then compared via the cosine similarity of the document and the query vector. The indexing is then done by sorting all the cosine similarities between documents and the query in descending order.

3.4 Latent Dirichlet Allocation (LDA)

LDA is a generative topic model that makes use of variational inference to model a multinomial Dirichlet distribution over topics per document. For the purpose of this work, we have used the gensim implementation of LDA. The α and η values are determined automatically by the gensim model.The corpus and queries are then projected in LDA space and ranking is done via KL-Divergence.

4 EXPERIMENTAL SETUP

4.1 Word2Vec

Word2Vec was trained with negative sampling and using the nn.Embedding module with sparse Adam. We trained for 200 000 epochs with batches of size 1024. We used three times as many negative samples as positive ones.

4.2 Doc2Vec

To fine-tune the right vocabulary size we take into account that the min_count values (250, 50, 15, 5, 2) roughly result in vocabulary sizes of (10, 25, 50, 100, 200). We train the model for four epochs.

4.3 LSI/LSA and LDA

For the topic models we examined the change in performance when trying different numbers of topics. For the BoW-LSI and the TF-IDF-LSI models we tried 10, 50, 100, 500, 1000 and 2000 topics respectively. We left out 5000 and 10000 topics as the computation cost was too large for our resources and we do not expect it to improve performance a lot. For LDA we trained models with 10, 50, 100, 500 and 1000 topics for 6, 3, 3, 6, and 2 epochs respectively (unequal also due to time constraints).

4.4 Data

We use the queries provided and define the validation set as queries 76-100 and the test set as all remaining queries that are not in the validation set. The queries are accompanied by an annotated set of relevant and irrelevant documents, which we used to measure a model's performance.

5 RESULTS

AQ2.1

Find the top ten similar words to five words of your choice, based on their word2vec representations. Analyze the results. Are the words similar in your opinion? Word2Vec is trained with the best parameters on the entire dataset from Doc2Vec (see AQ4.3). The ten most similar words for *dog* are:

Words for *congress*: ['congress', 'excerpt', 'vital', 'plan', 'award', 'wallach', '8-year-old', 'rang', 'anticip', 'appoint']
Words for *dollar*: ['dollar', 'cautiou', 'economist', 'judiciari', 'push', 'substanti', 'needl', 'meaning', '46', 'price']
Words for *dream*: ['dream', 'u.s.', 'mississippi', 'year', 'nearli', 'crawl', 'matthew', 'corn', 'game', 'humor']
Words for *japan*: ['japan', 'wari', 'econom', 'navi', 'regular', 'exploit', 'miss', 'toss', 'expans', 'extradit']

Words for *war*: ['war', 'govern', 'power', 'talk', 'barricad', 'u.s.', "'s", 'put', 'airport', '“']
Words for *weather*: ['weather', '200,000', 'degre', 'prove', 'notifi', 'marri', 'defend', 'su', '1938', 'bushel']
Words for *dog*: ['dog', 'knife', 'felt', 'dem', 'learn', 'combin', 'texa', 'spectat', 'satan', 'conflict']
Words for *king*: ['king', 'communist', 'wednesday', 'estat', 'thursday', 'one-tim', 'commiss', 'depart', 'would', 'either']
Words for *presid*(president): ['presid', "'s", 'nation', 'state', 'govern', 'first', 'say', 'tuesday', '"', 'thursday']
Words for *green*: ['green', 'jordan', 'hous', 'roe', 'mario', 'takeov', 'dalla', 'volum', '47', 'gore']
Compared to pre-trained word2vec embeddings this one falls flat. A lot of nearest words do not have a direct connection to the original word. However, for each word one can see a connection with at least one of its ten most similar words, e.g. *weather* and *degre(e)*, or *dollar* and *price*. Most likely, the model has to be trained much longer to achieve good performance.

AQ2.2

Find the top ten similar documents to an arbitrary piece of text, using the doc2vec representations. Analyze the results. Are the documents similar in your opinion? If not, what is your analysis for the error?

The given document was randomly sampled from the dataset. It details the court case of a Japanese textbook writer that sued the Japanese Ministry of Education which ordered him to re-write the parts of the textbook detailing Japanese aggressions during Word War II. The most similar document found was the original document itself, which makes sense, as it should be the most similar. The second most similar document is an article about the same court case, only from a Chinese perspective, who are criticizing that war crimes against China are being covered up. Since this article is about the same event as the original one, the high similarity score is legitimate. The other retrieved documents are all about Japan. One is about its school system which can be counted as an error and is probably due to the word "textbook" in the original document. Most are about past Japanese war crimes, with several referencing the original textbook court case. Thus, Doc2Vec seems to work very well to identify similar documents.

AQ3.1

For each of the LSI models you built over AP, and for LDA, select the five top significant topics from your model. Check the top terms in each topic. Which topics actually represent a particular subject? Analyse the results. Do you observe a difference?

BoW-LSI with 500 topics, shows for the top 5 topics the words:
Topic 1: state, percent, new, one, peopl, ...
Topic 2: percent, 0, price, million, rate, ...
Topic 3: percent, cent, bush, million, market ...
Topic 4: bush, dukaki, cent, presid, democrat, ...
Topic 5: soviet, cent, nt́, govern, parti, ...
The BoW-LSI clearly captures some common news topics like demographics (Topic 1), economy (Topic 2/3), politics (Topic 4). But there is also an overlap between topics as the word percent is in the top 5 for the first three topics. We think this is due to BoW only taking into account raw frequency counts which leads to an overestimation of very frequent words.

TF-IDF-LSI with 500 topics, shows for the top 5 topics the words:
Topic 1: ticketron, telecharg, teletron, 239-6200, 221-1211, ...
Topic 2: y., stangeland, schuett, manton, volkmer, ...
Topic 3: borski, dewin, kolter, gaydo, yatron, ...
Topic 4: erdreich, viscloski, gallegli, jontz, hefley, ...
Topic 5: emi-manhattan, dárbi, wilburi, geffen, arista, ...
The TF-IDF-LSI has topics that do not intuitively make sense to a human. The top words are very specific like names rather than common words and somtimes contain wierd very rare words. We think that this is due to the nature of TF-IDF scoring high if the words are only present in very few documents. This underlines the importance of pre-processing and removing stopwords. This findings show that it is hard to catch all unimportant rare words without loosing rare words that are important for information retrieval. Due to the lack of semantic meaning of the top topics of the TF-IDF-LSI we expect it to perform worse than the BoW-LSI.

LDA with 500 topics, shows for the top 5 topics the words:
Topic 1: palestinian, isra, arab, israel, upris, ...
Topic 2: plo, palestinian, arafat, palestin, jordan, ...
Topic 3: like, get, work, thing, never, ...
Topic 4: ton, wheat, grain, corn, bushel, ...
Topic 5: rebel, guerrilla, govern, attack, kill, ...
For LDA, it shows that 4 of 5 topics represent more or less actual topics. Topics 1 and 2 for example, cover topics about the middle-east and topic 4 about (prices of) agricultural products.

AQ4.1

Report the retrieval performance in terms of MAP and nDCG for all of the methods, on a) all queries, and b) queries 76-100. To be precise, you need to report 24 numbers in a table.

Doc2Vec results are for the un-tuned default values (see Q4.3).

Model	MAP	nDCG
TF-IDF	21.97	58.19
Word2Vec	9.45	46.33
Doc2Vec	0.60	27.07
BoW-LSI	8.34	45.60
TF-IDF-LSI	7.73	44.00
BoW-LDA	5.81	40.39

Table 1: Metrics results of the default model on all queries. Mean average precision and Normalized Discounted Cumulative Gain in percent.

AQ4.2

Significance testing is a way of showing that if the difference observed between the performance of two models is due to chance or not. Use pytrec eval to perform a t-test between every pair of models, on the complete query set. You need to report the results of six tests. (see example)

Model	MAP	nDCG
TF-IDF	18.15	54.13
Word2Vec	5.82	41.23
Doc2Vec	0.09	26.0
BoW-LSI	5.67	42.20
TF-IDF-LSI	6.99	42.18
BoW-LDA	3.74	37.01

Table 2: Metrics results of the default model on validation queries. Mean average precision and Normalized Discounted Cumulative Gain in percent.

The t-test for the nDCG results:
The difference between the results of all models is significant (p-value<0.05), except for the difference between BoW-LSI and TF-IDF-LSI, where the p-value was 0.262. So the difference between the nDCG results of these models is insignificant. The p-value for the difference of the results of BoW-LDA and TF-IDF-LSI was 0.01. All other p-values were smaller than 10^{-5}.

The t-test for the MAP results:
Now both the differences between BoW-LSI and TF-IDF-LSI, and between BoW-LDA and TF-IDF-LSI are insignificant with p-values of respectively 0.0812 and 0.628. The other p-values are still small and the differences between results of the other models are therefore significant.

AQ4.3

Report the retrieval performance in terms of MAP and nDCG for all of the methods, on a) all queries, and b) test queries. Use the parameters leading to the best performance on validation set. Attach the result files corresponding to each run to your report.

The best model for BoW-LSI as well TF-IDF-LSI was the model with the highest number of topics (2000).

Model	MAP	nDCG
TF-IDF	21.97	58.19
Word2Vec	9.45	46.33
Doc2Vec	0.65	27.95
BoW-LSI	11.28	50.05
TF-IDF-LSI	10.37	47.83
BoW-LDA	5.81	40.39

Table 3: Metrics results of the fine-tuned model on all queries. Mean average precision and Normalized Discounted Cumulative Gain in percent.

Link to the google drive containing the TREC files of the ranking systems, see footnote[2].

5.0.1 Fine-tuning Doc2Vec. Fine-tuning the Doc2Vec model for window size, embedding dimension and vocabulary size was done per parameter while keeping the other two fixed to default values. The default values are (window=15, dimension=300, vocabulary=25k). The resulting best parameters on the validation set are (window=5, dimension=400, vocabulary=200k). These parameters fare worse when applied to all queries, then the best are (window=5, dimension=300, vocabulary=25k)

[2]https://drive.google.com/drive/folders/1bAxlKkO8uLrYGN8zNMv9CUKkcMIz28d4?usp=sharing

Model	MAP	nDCG
TF-IDF	22.74	59.01
Word2Vec	10.18	47.36
Doc2Vec	0.72	26.76
BoW-LSI	12.12	51.01
TF-IDF-LSI	10.43	48.22
BoW-LDA	6.22	41.08

Table 4: Metrics results of the fine-tuned model on test queries. Mean average precision and Normalized Discounted Cumulative Gain in percent.

AQ4.4

Perform a t-test between the result of each method using the best parameters and the default ones used in AQ4.1. Describe your observations.

Since we use only one setting for word2vec (the best one from doc2vec) we cannot compare it with 'default values'. With a p-value of 0.188 for nDCG and 0.92 for MAP, doc2vec shows no significant improvement after tuning. All other models show with low p-values $< 10^{-5}$ significant improvement after tuning. LDA is taken out of consideration, since we found the default model to be our best model.

AQ4.5

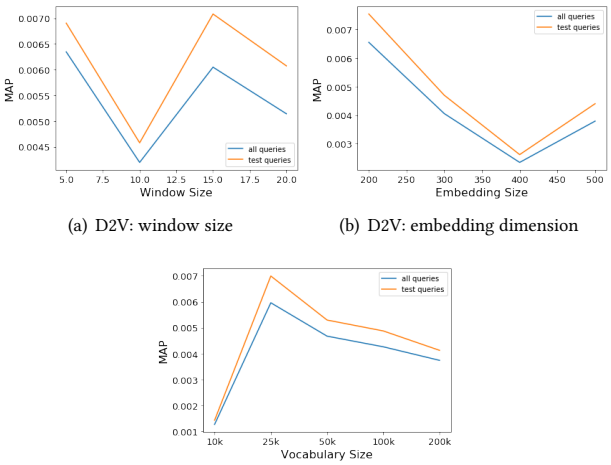
For each parameter, plot the retrieval performance in terms of MAP with respect to the value of the parameter, for a) all queries, and b) test queries. Describe your findings.

Figure 1 shows the behaviour of the Doc2Vec model when tuning either the context window size, the embedding dimension or the vocabulary size. For all, there is no clear trend, the curves rather describing a zig-zag line. The best values seem to be 15, 200 and 25k for window size, embedding dimension and vocabulary size respectively. It should also be noted that on the validation set, other parameters showed the best performance. This indicates that the validation set was not characteristic of the dataset.

AQ4.6

Doing a query-level analysis provides insights on the weaknesses and strengths of the retrieval models. For each of the four retrieval methods implemented above, find success and failure cases: queries for which the MAP was highest or lowest. Analyse the results, possibly with checking the qrels file. Discuss why you think each case happened.

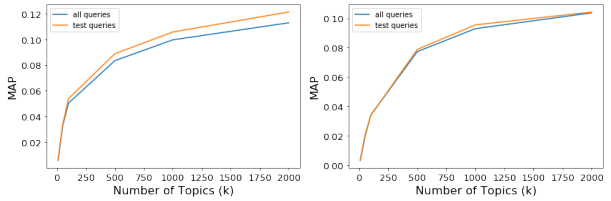
The tables on the next page show the top 5 success and failure cases. In general, there are some queries which are easy and hard for all models. For example, *Oil Spills* is easy for every model, probably because the two words appear together often and there is plenty of training data for them. In contrast to that, hard queries are generally longer and often contain hyphenated words, which probably makes these tokens rare, same as for abbreviations like *MCI*. Doc2vec seems to handle long queries better than the others



(a) D2V: window size (b) D2V: embedding dimension

(c) D2V: vocabulary size

Figure 1: Performance of the Doc2Vec model for different parameter settings on all queries as well as test queries.



(a) BoW-LSI (b) TF-IDF-LSI

Figure 2: Change in Mean Average Precision performance for increasing number of topics for LSI models.

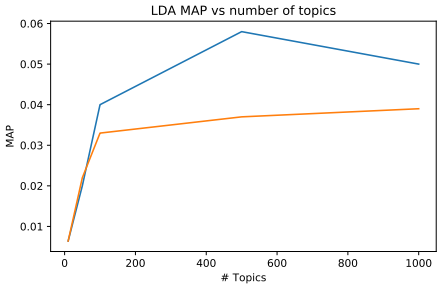


Figure 3: Change in Mean Average Precision performance for increasing number of topics for LDA.

and also performs well on queries that are more complex and need to capture some context. Word2Vec likely has trouble with queries that contain a lot of unknown tokens.

AQ4.7

Find the top-5 queries that have the highest variance in terms of MAP between different retrieval models. Provide an analysis why the performance of the models differs a lot on these queries compared to the rest.

Top 5 queries that have the highest variance in model performance between all the retrieval models from AQ4.3:

Surrogate Motherhood (MAP variance: 0.0789)

The Consequences of Implantation of Silicone Gel Breast Devices (MAP variance: 0.0768)

Vietnam Veterans and Agent Orange (MAP variance: 0.07532)

Oil Spills (MAP variance: 0.0543)

Smoking Bans (MAP variance: 0.0522)

It is hard to make any statement about the reasons why exactly this queries yield different performance results over the retrieval systems. We have 3 very short queries and 2 longer ones. Possibly, doc2vec is better for longer queries with context as it takes word order into account while the others work better for short queries. This might influence the variance of performance. The topic models might especially work well for queries that belong to a certain topic like Vietnam Veterans and Agent Orange belong to history and war while word2vec would probably perform poorly for this query because it mixes between the single words instead of capturing the overall context.

6 CONCLUSIONS

In summary, several models were evaluated on their performance in information retrieval. We found, surprisingly, that TF-IDF stands out on performance, compared to the dense representation models Word2Vec, Doc2Vec, LSI and LDA. This comes as a surprise, because tf-idf does not take grammatical relations into account such as synonymy. We performed t-tests and plotted results to validate our findings. However, we believe that due to time constraints, our results have been negatively influenced. Since training models takes a relatively large amount of time with limited resources, we were not able to fully explore the capabilities of each of the models in question. Nevertheless, these results could be taken as a starting point for further research.

REFERENCES

[1] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[2] Thomas K Landauer, Peter W Foltz, and Darrell Laham. 1998. An introduction to latent semantic analysis. *Discourse processes* 25, 2-3 (1998), 259–284.

[3] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*. 1188–1196.

[4] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

Best 5 queries word2vec	Worst 5 queries word2vec
Vietnam Veterans and Agent Orange	Funding Biotechnology
Oil Spills	MCI
Smoking Bans	"Black Monday"
Greenpeace	Find Innovative Companies
South African Sanctions	Natural Language Processing

Table 5: Success and failure queries in terms of MAP for the fine-tuned word2vec model.

Best 5 queries doc2vec	Worst 5 queries doc2vec
Electric Car Development	The Consequences of Implantation of Silicone Gel Breast Devices
Possible Contributions of Gene Mapping to Medicine	Machine Translation
Commercial Overfishing Creates Food Fish Deficit	Use of Mutual Funds in an Individual's Retirement Strategy
Impact of Government Regulated Grain Farming on International Relations	Term limitations for members of the U.S. Congress
"Downstream" Investments by OPEC Member States	Abuse of the Elderly by Family Members, and Medical and Nonmedical Personnel, and Initiatives Being Taken to Minimize This Mistreatment

Table 6: Success and failure queries in terms of MAP for the fine-tuned doc2vec model.

Best 5 queries BoW-LSI	Worst 5 queries BoW-LSI
Surrogate Motherhood	"Black Monday"
Vietnam Veterans and Agent Orange	Funding Biotechnology
Smoking Bans	Demographic Shifts across National Boundaries
Oil Spills	Industrial Espionage
Leveraged Buyouts	Fiber Optics Equipment Manufacturers

Table 7: Success and failure queries in terms of MAP for the fine-tuned BoW-LSI model.

Best 5 queries TF-IDF-LSI	Worst 5 queries TF-IDF-LSI
The Consequences of Implantation of Silicone Gel Breast Devices	Merit-Pay vs. Seniority
"Stealth" Aircraft	U.S. Army Acquisition of Advanced Weapons Systems
Oil Spills	Computer-aided Crime Detection
Nuclear Proliferation	Electric Car Development
Greenpeace	The Human Genome Project

Table 8: Success and failure queries in terms of MAP for the fine-tuned TF-IDF-LSI model.

Best 5 queries LDA	Worst 5 queries LDA
Oil Spills	MCI
Oil Spill Cleanup	"Black Monday"
Acid Rain	The Human Genome Project
Iran-Contra Affair	The Consequences of Implantation of Silicone Gel Breast Devices
Automobile Recalls	Merit-Pay vs. Seniority

Table 9: Success and failure queries in terms of MAP for the fine-tuned LDA model.