Laval University at TREC Dynamic Domain 2016: Subtopic extraction focused on Named Entities

Robin Joganah, Richard Khoury and Luc Lamontagne
Department of Computer Science and Software Engineering, Université Laval, Québec, Canada
robin.joganah.1@ulaval.ca, richard.khoury@ift.ulaval.ca, luc.lamontagne@ift.ulaval.ca

***Abstract*— This paper describes the results submitted by Laval University to the TREC 2016 Dynamic Domain track. We submitted five runs. For this year we decided to focus around Named Entities to interpret the subtopics. Named Entities are one of the two types of queries we identified in this challenge, the other being queries about concepts. We describe in this paper our experiments to determine if targeting this type of query with a specific pipeline can lead to a global improvement of the system by taking into account the specificity of the queries.**

***Index Terms*—Dynamic Domain, Information Retrieval, Topic Modeling**

I. Introduction

The Dynamic Domain track challenge [12] consists of retrieving information (IR) for a specific domain (such as Ebola or Polar scientific documents) through a dynamic process that takes into account feedback from a simulated user. The aim of this challenge is to return a diversified set of documents that both explores the full range of topics within the domain and researches in greater depth topics the user shows interest in.

For this year's competition, we submitted 5 runs that use different parameters and different pipelines. The main pipeline we investigated focuses on named entities. We extended the dynamic search pipeline we described in our previous work [6] by taking advantage of Named Entities at multiple stages of the dynamic search. Our idea is that an improvement on a specific type of query can have an impact on the whole system. Indeed, queries that contain one or more named entities are frequent in the TREC 15 and TREC 16 dataset. We could divide the dynamic search problem in queries that are either concept-oriented or entity-oriented.

We compared this pipeline with our system from TREC 15 [6]. We also compared different similarity configurations, like BM25 and TF-IDF, and we will show that these parameters have a huge impact on the performance of the system.

II. Related work

Dynamic Domain is a new domain and different approaches have been tested during the first TREC Dynamic Domain track. The spectrum of approaches is broad, ranging from Partially Observable Markov Decision Process (POMDP) [11] to re-ranking of documents by similarity with user feedback [11, 9]. As part of our previous participation, our team [6] experimented with techniques like clustering [3] and topic modeling [1] to inject diversification in the information retrieval process. These techniques had some success in the Judged-only task that contained documents assigned to at least one topic. However, for the main task where the collection contains documents that are not associated to any topics, the results we obtained were lesser due to the noise of un-judged documents. In these conditions, it becomes more difficult to effectively model the topics that are relevant to the interests of the user. That is why we decided this year to focus on information extraction and noise reduction.

Information extraction has an important role in natural language processing, most of the techniques [5] trying to take advantage of Named Entity Recognition (NER) and Relation Extraction (RE). This information helps to find keywords and keyphrases that can help to find relevant words. The extraction of keyphrases has also been studied [4] and usually takes advantage of information extraction and topic modeling to rank the most diversified and relevant phrases to cover the document entirely. These techniques remind us that one important aspect of the Dynamic Domain task is to interpret information from the user feedback, which is a textual passage taken from the document and returned by the system. During our TREC experiments of last year [6] we showed that our system pipeline was able to take advantage of named entity recognition on the text passages to expand our query and find new relevant documents.

In order to fully take advantage of information extraction algorithms, there is a need to clean our dataset to remove noise that can lead to poor data mining performances [13]. We pre-process the dataset to extract the relevant article text from webpages and remove the noise around the article introduced by page footers, menus and news sections. This pre-processing step was performed using the BoilerPipe library [7].

III. Global view of the system

The system is composed of two phases: the information retrieval process and the feedback processing. The first one occurs during the initial phase and for each turn until the systems decide to stop. The feedback processing occurs after the initial search phase has occurred and when the user provides his feedback about the first results returned by the system. These phases are illustrated in Figure 1.

A. Initial Search Phase & Information Retrieval Process

The first phase consists of a retrieval process to obtain an initial set of documents based on the user's original query. In our experiments, the retrieval process makes use of the popular Solr search engine to retrieve a set of *n* documents from the dataset. We retain five documents from the original query. A classic IR system like Solr can retrieve the top documents by keywords, but it does not provide the mechanisms to diversify the results as needed by the Dynamic Domain challenge. For this, it is necessary to discover topics present within the result set and to return one representative document per topic. We experimented with two algorithms for this purpose: we used K-means clustering to discover clusters of documents within the search results [10], and an implementation of Latent Dirichlet Allocation (LDA) algorithm [1] for topic modelling. The models provided by these additional algorithms allow us to re-rank the documents and select the five best results to return to the user.

B. Simulated User's feedback

The second phase of the dynamic retrieval process takes into account the user's feedback about the five documents submitted to him/her. This feedback consists of a highlighted passage from each relevant document returned by the system. If the document is relevant to multiple subtopics, then we have a passage for each subtopic with a rating between 1 and 4.

We use a NER algorithm on the highlighted passage to extract information to be added to the initial query. With this new query, we proceed with the same information retrieval process as we did in the initial search phase.

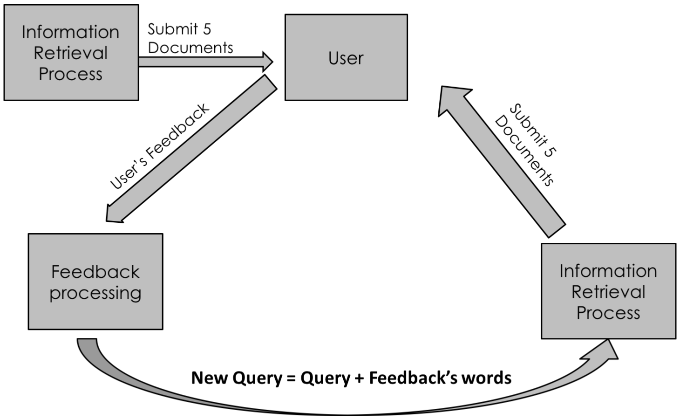
Information Retrieval Process
Submit 5 Documents
User
User's Feedback
Submit 5 Documents
Feedback processing
Information Retrieval Process
New Query = Query + Feedback's words

Fig. 1. Global System Overview

C. Stopping Criteria

As illustrated in Figure 1, phase 2 of our system is a loop and the system has to determine when to halt the search and break out of this loop. This should happen at the point when an additional iteration will no longer yield useful refinements of the search results. We used a hard-coded limit of two iterations, which means that the system will only consider once the user's feedback once. This decision is motivated by the empirical realization that we were not able, during our experiments, to acquire additional information to stabilize or improve the value of the CubeTest after two iterations.

IV. Information Retrieval Process

Hence retrieval occurs during two steps of our dynamic search process. The first time is during the initial search phase, which we discussed previously, where it must generate an initial set of five documents based on the original query. It occurs again at every iteration of the dynamic refinement phase of the system to obtain a new set of documents using the expanded query. The choice of IR process has an impact on the performance of our entire system, since the Cube Test (CT) and μ-ERR metrics used to evaluate the task penalize sessions with irrelevant documents as a waste of user's time.

A. Solr and Similarity Measures

We used the Solr search engine as a baseline to evaluate our IR configurations. We tested two different similarity measures to find documents, default similarity (TFIDF) and BM25, and compared the results we obtained with those. We used them to retrieve documents given a query, either the user's original query or the expanded queries reformulated by our dynamic refinement stage. The top five documents returned are kept as Solr recommendations, the five most relevant documents given the query. Meanwhile, the top *n* documents returned by Solr are used as a corpus to build models using LDA or K-Means algorithms, where *n* is determined at the beginning of the run.

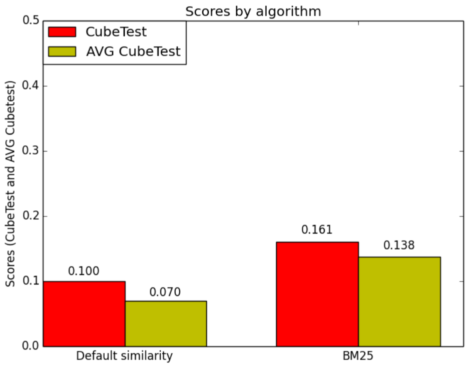

Fig. 2. TF-IDF and BM25 comparison on Ebola dataset

We present in Fig. 2 our comparison of the results obtained with BM25 and TF-IDF (which is the default similarity in Solr). We can see that the choice of similarity measure can have a huge impact on the results. In future works, we plan on comparing these measures with the language model which has been used by other participants during the last year's competition [9]. A language model is also the default similarity in other search engines like Indri, so this is a promising alternative to explore.

B. Latent Dirichlet Allocation (LDA)

LDA is an algorithm that takes as input a collection of documents and discovers the groupings of topics it implies. Topics are represented as probability distributions over words contained in the documents. In our system, LDA is responsible of discovering five different topic groups from the top *n* documents (with *n* between 20 and 200) returned by Solr for a specific query. Next, the system builds five expanded queries, one for each of the five topic groups, by adding the five most probable words of each topic distribution to the current query. It then runs a new Solr search with each of the five new queries, and keeps the top document of each search to form the set of five LDA recommendations.

C. K-Means

K-Means is a clustering algorithm that builds clusters around centroids. In our system, we use this algorithm to create five clusters of documents from a list of 200 documents returned by Solr. To build the clusters, we make TFIDF vectors to represent the documents and of a cosine distance to estimate their similarity. The system keeps the document closest to each cluster centroid to create the set of five K-means recommendations.

V. Named Entity Focused Pipeline

The main difficulty with topic modeling and clustering is the topics of interest are sometimes buried in noise. For example, a query about a person can return a document containing a few sentences related to the person and a lot of irrelevant content. If we consider the whole document to be relevant during our model construction, the extra text not related to the user's interests will lead to a noisy or bad model.

To solve this issue, we apply a sentence segmentation algorithm to each document and keep only the sentences where at least some of the topic words appear. In the future, we plan on keeping neighboring sentences as well, to add a notion of context to our model, and see how it impacts the system.

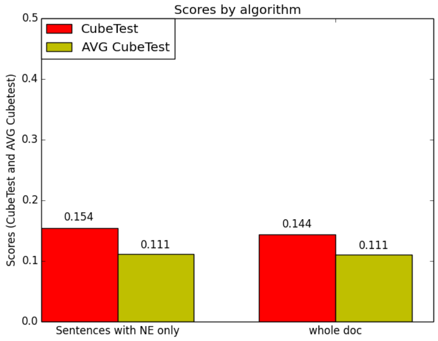

Fig. 3. Impact of selection of sentences with part of the query

We tested this pipeline on the Ebola dataset and our results are presented in Figure 3. We did not make any distinction between concept-oriented or entity-oriented queries during our experiments, but our results indicate that this pipeline has a positive impact for the CubeTest metric. However, we can see that the average CubeTest remains the same. We intend to test this specific pipeline solely on entity-oriented queries in future work and process concept-oriented queries with a different pipeline.

VI. Descriptions of the systems submitted

A. Data

The dataset proposed for this competition is separated in two domains.

1) Ebola

The Ebola dataset is related to the Ebola outbreak in 2014-2015, it contains 194,481 web pages.

2) Polar

The polar dataset contains 244,536 files.

B. Data Pre-Processing

We used BoilerPipe to extract articles from webpages in order to focus on the real content and to remove the noise found on web pages like frames, page footers, menu, sign-in forms, etc.

We also used Solr built-in filters with Porter stemming.

C. Systems

Our five systems are based on a similar pipeline which uses NLTK Named Entity Recognition (NER) algorithms to extract named entities from the text passage highlighted by the simulated user. These words are then added to the query to retrieve documents and run our algorithms over these documents.

We can separate the systems into two different categories:

- A baseline system that only uses the best results from the search engine
- Systems using clustering or topic modeling to expand the queries.

In this second category, we have three systems that use the whole document to perform the analysis and one system (UL_LDA_NE) that uses only sentences containing some part of the query.

1) UL_BM25

This is the baseline system that takes the 5 best results retrieved by Solr using the BM25 measure.

2) UL_LDA_200

This system take the first 200 documents retrieved by Solr with the initial query and the BM25 measure of similarity. It performs topic modeling with LDA to find 5 different topics. These topics are composed of words that are more likely to be part of each topic. We select the top *n* words to reformulate a query and search for documents with this query. The best document is added to our list. If one document appears in two lists, we take the second result from the first topic. At the end of the process, we obtain 5 documents, one from each topic query.

3) UL_LDA_NE

This system also uses LDA and BM25 measures of similarity but only uses the top 20 documents to expand the initial query. We added sentence segmentation to have sentences intersecting with the initial query in order to focus topic modeling around content more likely to be relevant.

4) UL_Kmeans

This system is similar to UL_LDA_200, the only difference being the use of K-means instead of LDA to search for different documents. In this case, we perform clustering with $K = 5$ and we take the documents that are the most similar to each centroid.

5) UL_LDA_Psum

This system uses topic modeling in a different way. We normalize the probabilities for each document to be a part of each topic by the probability of each topic. Then we choose documents that cover most of the popular topics.

TABLE I
SYSTEMS COMPARISON

ID Submission	**Similarity measure**	**Clustering**	**Topic modeling**	**Clustering / Topic modeling applied on n documents**	**Feedback processing**	**Sentences segmentation and filtering**
UL_LDA_200	BM25	No	Yes	200	NER on user's feedback	No
UL_LDA_NE	BM25	No	Yes	20	NER on user's feedback	Yes
UL_BM25	BM25	No	No	0	NER on user's feedback	No
UL_Kmeans	BM25	Yes	No	200	NER on user's feedback	No
UL_LDA_Psum	BM25	No	Yes	200	NER on user's feedback	No

TABLE II
SUBMISSIONS RESULTS

ID Submission	**ACT**	**CT**	**nDCG**	**nERRIA**	**AVG-NDCG**	**AVG-nERRIA**	**nSDCG**
UL_LDA_200 @ 2 iterations	0.0815	0.0995	0.2110	0.1772	0.0121	0.0096	0.0431
UL_LDA_NE @ 2 iterations	**0.1092**	**0.1309**	**0.2779**	**0.2319**	**0.0192**	**0.0177**	0.0703
UL_BM25 @ 2	0.1031	0.1097	0.2520	0.2131	0.0098	0.0083	**0.0759**
UL_Kmeans @ 2 iterations	0.0815	0.0803	0.1922	0.1685	0.0072	0.0064	0.0386
UL_LDA_Psum @ 2 iterations	0.0274	0.0438	0.1039	0.0750	0.0052	0.0034	0.0165
Median @ 10 iterations	0.0985	0.0801	0.3142	0.2778	0.0064	0.0054	0.0543
Median @ 2 iterations	0.1352	0.1281	0.3142	0.2777	0.0162	0.0139	0.0940

VII. TREC Submissions Results Analysis

The results reported in Table II are the scores from metrics used in TREC 16 such as CubeTest [8], ERR [2] and NDCG (Normalized Discounted Cumulative Gain).

The results indicate that we are below the TREC median at iteration 2 when we stop the dynamic search process, except for the CubeTest of our UL_LDA_NE run. These results we present are limited by our baseline system using a BM25 measure that might be less accurate than those based on language models. The second limitation is that our technique (UL_LDA_NE) which targets mainly named entity topics is not intended to work well on queries pertaining to concepts. For instance, most queries for the Polar dataset are concept-oriented queries, which our system cannot handle as well.

However, with five different systems presented for the competition, we can compare the results to see how each technique influences the system's behaviour. We can see that LDA (UL_LDA_200) and K-means (UL_Kmeans) have the same impact and that they have a negative effect if we compare their results with the baseline. This can be explained by the fact that we used n=200, which means that we are working with a large number of document to update the initial query. We can see that our pipeline using NE recognition and only 20 documents lead to better clustering or topic modeling.

The Psum (UL_LDA_Psum) system uses LDA to find documents that might covers multiple topics. The poor performance of the system suggests that a document covering multiple well-represented topics does not necessarily have relevant information about new topics. However, this hypothesis would need to be tested in further work.

VIII. Conclusion and Future Work

In this paper, we presented a pipeline focused around named entity topics. Experimental results indicate that it works better than our baseline system. We can observe some improvements on the entire IR process when using this pipeline, but its impact on the polar domain specifically seems to be limited due to the concept-oriented nature of queries in this domain. These results motivate us to devise a specific pipeline for concept-oriented queries and a process to classify queries by types, with the aim of providing specialized processing for different types of queries. We also plan to work on these concepts by putting more emphasis on keyphrase extraction that could help the unsupervised system to discover the user's interests. We also intend to devote more efforts to developing strategies to determine when the system should stop, since that is an important step of the automated process.

IX. References

[1] D. ANDRZEJEWSKI and D. BUTTLER, *Latent topic feedback for information retrieval*, *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2011, pp. 600-608.

[2] O. CHAPELLE, D. METLZER, Y. ZHANG and P. GRINSPAN, *Expected reciprocal rank for graded relevance*, *Proceedings of the 18th ACM conference on Information and knowledge management*, ACM, 2009, pp. 621-630.

[3] D. COHN, R. CARUANA and A. MCCALLUM, *Semi-supervised clustering with user feedback*, Constrained Clustering: Advances in Algorithms, Theory, and Applications, 4 (2003), pp. 17-32.

[4] K. S. HASAN and V. NG, *Automatic Keyphrase Extraction: A Survey of the State of the Art*, *ACL (1)*, 2014, pp. 1262-1273.

[5] J. JIANG, *Information Extraction from Text*, in C. C. Aggarwal and C. Zhai, eds., *Mining Text Data*, Springer US, Boston, MA, 2012, pp. 11-41.

[6] R. JOGANAH, R. KHOURY and L. LAMONTAGNE, *Laval University and Lakehead University at TREC Dynamic Domain 2015: Combination of Techniques for Subtopics Coverage*, *Proceedings TREC 2015*, Gaithersburg, 2016.

[7] C. KOHLSCHÜTTER, P. FANKHAUSER and W. NEJDL, *Boilerplate detection using shallow text features*, *Proceedings of the third ACM international conference on Web search and data mining*, ACM, 2010, pp. 441-450.

[8] J. LUO, C. WING, H. YANG and M. HEARST, *The water filling model and the cube test: multi-dimensional evaluation for professional search*, *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, ACM, 2013, pp. 709-714.

[9] J. LUO and H. YANG, *Re-ranking via User Feedback: Georgetown University at TREC 2015 DD Track*, *Proceedings of TREC 2015*, Gaithersburg.

[10] M. STEINBACH, G. KARYPIS and V. KUMAR, *A comparison of document clustering techniques*, *KDD workshop on text mining*, Boston, 2000, pp. 525-526.

[11] H. WU, *Modeling Search Engine's Explorations in Dynamic Search: An Ontological Perspective*, (2016).

[12] H. YANG, J. FRANK and I. SOBOROFF, *TREC 2015 Dynamic Domain Track Overview*, (2016).

[13] L. YI, B. LIU and X. LI, *Eliminating noisy information in web pages for data mining*, *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2003, pp. 296-305.