Université de Montréal at the NTCIR-11 IMine Task

Arbi Bouchoucha, Jian-Yun Nie and Xiaohua Liu
Dept. of Computer Science and Operations Research
University of Montreal
Montreal (Quebec), Canada
{bouchoar, nie, liuxiao}@iro.umontreal.ca

ABSTRACT

In this paper, we describe our participation to the NTCIR-11 IMine task, for both subtopic mining and document ranking sub-tasks. We experimented a new approach for aspect embedding which learns query aspects by selecting (good) expansion terms from a set of resources. In our participation, we used five representative resources: ConceptNet, Wikipedia, query logs, feedback documents and query suggestions from Bing, Google and Yahoo!. Our method is trained in a supervised manner according to the principle that related terms should correspond to the same aspects. We tested our approach when using a single resource, and when using different resources. Experimental results show that our best document ranking run is ranked No. 2 of all 15 runs in terms of coarse-grain and fine-grain results.

Team Name

UDEM

Sub-tasks

Subtopic Mining (English), Document Ranking (English)

Keywords

Diversified Query Expansion, Query Classification, Aspect Embedding, Resource Integration

1. INTRODUCTION

In NTCIR-11, the UDEM group participated in IMine task, which includes two sub-tasks: subtopic mining and documents ranking, both for English topics.

We used a new method to automatically learn latent aspects [1] for diversifying search results, by using multiple resources. Our approach is based on embedding to select a set of 'good' expansion terms for an original query, such as each expansion term can be mapped into one or several possible aspect(s) of the query. Candidate expansion terms can be selected from external resources (*e.g.*, Wikipedia, query logs, WordNet [16]), or from query suggestions (*e.g.*, those of Bing or Yahoo!). Our approach belongs to a family of search result diversification (SRD) approaches, recently proposed by Bouchoucha et al. [4, 5] and Vargas et al. [21] which is called diversified query expansion (DQE). DQE explicitly models the query aspects and attempts to select documents that cover as much as possible these aspects. In comparison with the usual SRD approaches, DQE tries to first generate a set of diversified (*i.e.*, non-redundant) expansion terms to help retrieve more diversified documents, then select diversified results from them.

The approach that we describe in this paper is related to the works that use several resources for query expansion (or query reformulation). In instance, Bendersky et al. [1] collect expansion terms (concepts) from news-wire and Web corpora. These resources are then used to compute the importance (weight) of each concept, and to perform pseudo-relevance feedback. They show that combining multiple resources is usually more effective than considering any resource in isolation, and that such combination yields to improve the diversity of search results. Recently, Deveaud et al. [10] conclude that the more we use several resources, the more likely we can improve the topical representation of the user information need. He et al. [13] propose the combination of click-logs, anchor text and web n-grams to generated related terms for query expansion, and experimentally show that combining several resources leads to select expansion terms from a good quality. Vargas et al. [21] adapt an existing diversification approach (namely $xQuAD$ [18]) to select diverse expansion terms from feedback documents. Their method selects expansion terms from groups of documents that cover the same query subtopic. More recently, Bouchoucha et al. [4] leveraged one resource, namely ConceptNet, or several resources [5] to select diverse expansion terms for the purpose of query subtopics coverage. In particular, the authors show that multiple resource tend to complete each other, and that using different resources usually yields to select expansion terms from a good quality which contributes to a better coverage of the query aspects. Liu et al. [14] proposed a new DQE approach called compact aspect embedding, which exploits trace norm regularization to learn a low rank vector space for the query. In their work, each expansion term is mapped into an aspect vector space and similar expansion terms (corresponding to the same query subtopic) are pushed together in the vector space. Despite that the authors in [14] use one single resource (namely query logs) for selecting candidate expansion terms, they experimentally show that learning latent aspect embedding helps to improve the state-of-the-art SRD approaches and leads to better relevance and diversity results.

In our participation in NTCIR-11 IMine task, we use the set of expansion terms automatically generated by our approach to design the different query subtopics, which is the

[1] In this paper, we interchangeably use the terms 'aspect' and 'subtopic' to refer to the same concept which is the user intent.

purpose of the subtopic mining sub-task. Afterwards, we expand each query by the set of expansion terms (aspects) obtained by our approach, and submit the expanded query to a retrieval system (we used Indri/Lemur [2] in our experiments). This leads to a set of diversified search results for the original query, which is the purpose of the second sub-task to which we participated (that is, document raking sub-task). Note that the set of returned documents regarding to each expanded query are not processed by any further selection, such as MMR [7] or xQuAD [18], although this is possible.

In addition, this year, participants to NTCIR-11 IMine task should also judge the 'class' of the query (*i.e.*, whether it is *ambiguous*, *broad* or *clear*). For that, we also learn a classifier for a query by collecting a set of features, and according to the class of the query, we propose different diversification methods.

It is worth noting that we applied our embedding framework only in the subtopic mining sub-task. The two remaining sub-tasks (*i.e.*, document ranking and query classification) do not benefit from the use of our approach. First, we develop a set of features for query classification. This classification is done offline, independently from the subtopic mining and document ranking sub-tasks. Then, we apply our embedding framework to collect expansion terms from a good quality for the purpose of better diversifying search results by covering as much as possible the query subtopics. This process is applied in different ways, according to the query type (*ambiguous*, *broad* and *clear*), since we observe that ambiguous queries need to be diversified more than broad queries. These latter need a further diversification than clear queries [3]. Finally, the outputted documents returned for the expanded queries are directly used for the document ranking sub-task. We provide the details of our participation in the reminder of this paper.

2. EMBEDDING FRAMEWORK

In this section, we first describe our embedding framework leading to learn aspects for a query. Then, we provide details about the similarity functions that we used for each resource, in order to compute the similarity between expansion terms.

2.1 Proposed Approach

We learn an embedding function to find the set of aspects underlying a query. An embedding vector spans over a set semantic dimensions, similar to topics in LDA [2], which could be used to describe different query aspects. An embedding vector is learned for each candidate expansion term. Each dimension in the embedding vector is intended to correspond to an aspect of the query. The embeddings are learned so that similar terms (measured using all resources and query suggestions) are also mapped into similar aspect vectors as follows:

$$\min \sum_{r \in R} \sum_{e_i, e_j \in E, i \neq j} \frac{1}{2} \cdot \omega_r \cdot (sim(\vec{e}_i, \vec{e}_j) - sim_r(e_i, e_j))^2$$

$$\text{subject to: } ||\vec{e}||_2^2 = 1, e^k \geq 0, k = 1, 2, \cdots, N, \forall e \in E. \tag{1}$$

Here, R denotes the set of resources we use to suggest similar terms, E means all expansion terms collected from all resources; $\omega_r \in [0,1]$ is the weight of resource r. All the weights are normalized based on $\sum_r \omega_r = 1$. For simplicity, we adopted a uniform distribution of ω_r in our participation, that is $\omega_r = \frac{1}{|R|}$ for all resources. N is the dimensions of aspect space; $\vec{e}_k$ is the vector corresponding to the k^{th} aspect, and e_k is the expansion term representing that aspect. All aspect vectors are normalized to 1 using ℓ_2-norm.

The basic idea is that a good aspect representation should make two known similar terms similar, whatever the resource used to recognize the similarity between them. To illustrate how our approach works in practice, let us consider the example query "apple" ((query #51)). If expansion terms like 'company' and 'store' are semantically similar according to some resource, then we want that the corresponding two vectors will be mapped into similar vectors in the embedding space, since they correspond to the same aspect of the query "apple" (which is *the apple company*). However, the embedding vector corresponding to a term like 'fruit' should appear far from of the vector corresponding to terms 'company' and 'store', since it corresponds to another aspect (or interpretation) of "apple" (which is *apple fruit*).

To solve the optimization problem described above, we use gradient descent [3], and we iteratively update the aspect vectors using the gradient descent rule until we observe no significant updates of the gradients with respect to all the aspect vectors [4]. Note that, for the aspect vectors initialization, we adopted a uniform distribution by setting each dimension to $\frac{1}{\sqrt{N}}$, without promoting any aspect to the other, because we find that this setting experimentally works good. Also, notice that the our objective function described in Formula 1 is guaranteed to converge towards a local minimum since the space's solution (E_r) is finite (there is usually a limited number of candidate expansion terms for any query).

In Formula 1, $sim(\vec{e}_i, \vec{e}_j)$ denotes the global similarity between $\vec{e}_i$ and $\vec{e}_j$ which is computed by using Formula 2:

$$sim(\vec{e}_i, \vec{e}_j) = \sum_{k=1,\cdots,N} e_i^k \cdot e_j^k \tag{2}$$

where e_i^k (resp. e_j^k) represents the value of the k^{th} dimension of aspect vector $\vec{e}_i$ (resp. $\vec{e}_j$). $sim_r(e_i, e_j)$ in Formula 1 is used to compute the local similarity between two expansion terms e_i and e_j, according to resource r. In the Section 2.2, we describe each similarity function that we used, regarding to each resource.

Finally, we observe that most of the candidate expansion terms suggested by the different resources for the same query tend to be redundant, since different resources can suggest similar expansion terms. To remove the redundancy of these terms, and to ensure a better coverage of the query subtopics, we further apply the MMRE (Maximal Marginal Relevance based Expansion) procedure, that was proposed by Bouchoucha et al. [4, 5], as follows:

$$e^* = \arg max_{e \in E}\{\beta \cdot sim_r(e, q) - (1-\beta) \cdot \max_{e' \in ES} sim_r(e, e')\} \tag{3}$$

[2] http://www.lemurproject.org/indri.php

[3] More precisely, we observed that it is better to not diversify clear queries since diversification of this type of queries risks to hurt the performance compared to a standard baseline.

[4] $\frac{||\nabla_i^{(t+1)} - \nabla_i^{(t)}||_2^2}{||\nabla_i^{(t)}||_2^2} < 0.0001, \forall e_i \in E$, where $\nabla_i^{(t)}$ means the gradient with respect to $\vec{e}_i$ after the t^{th} iteration.

where q denotes an original query; ES represents the set of expansion terms already selected; $E = \bigcup_r E_r$ means all expansion term collected from the different resources that we consider in this work; $\beta \in [0, 1]$ trades relevance to redundancy; $sim_r(e, q)$ denotes the resource specific similarity between the expansion term e and the original query q. Note that, if an expansion terms e is selected from a resource r, then we apply the similarity functions defined for that resource r (that is, $sim_r(e, q)$ and $sim_r(e, e^{'})$). Also, all similarities based resources are normalized to [0,1] to make them comparable. Following [5], we estimate $sim_r(e, q)$ by using the following simpler similarity between the expansion term and any sub-string of the query:

$$sim_r(e, q) = \max_{s \in q} sim_r(e, s) \cdot \frac{|s|}{|q|} \tag{4}$$

Here, s is a sub-string of q, $|s|$ denotes the number of words in s, and $sim_r(e, s)$ is the local similarity between e and s, based on resource r, as will be defined in the section 2.2.

2.2 Resource-based Similarity Functions

In our participation, we have investigated five typical resources: ConceptNet, Wikipedia, query logs, feedback documents and query suggestions from Bing, Google and Yahoo! which are provided by NTCIR organizers for INTENT2. We used the topics of the latter for training purposes. The reason of using several resources is twofold: (1) Several existing studies (*e.g.* [1, 5, 10, 13] to name just a few) showed that using multiple resources can yield better results for search result diversification, and (2) One single resource is usually not enough to ensure a good coverage of query subtopics and hence, combining several resources can help cover more subtopics of the query. In this work, we use the same definitions of the similarity functions proposed by Bouchoucha et al. [4, 5], computed on ConceptNet, Wikipedia, query logs and feedback documents. Now, we provide an overview of these resource-based similarity functions.

First, the similarity measured according to ConceptNet (which is a network of terms) considers the number of common nodes (terms) connected between two terms: The more there are common nodes, the more the two terms are considered similar. The similarity function $sim_C(e_i, e_j)$ between two expansion terms e_i and e_j based on ConceptNet is similar to the well-known *Jaccard* coefficient and is computed as follows:

$$sim_C(e_i, e_j) = \frac{|N_{e_i} \cap N_{e_j}|}{|N_{e_i} \cup N_{e_j}|} \tag{5}$$

where N_{e_i} (resp. N_{e_j}) is the set of nodes from the graph of ConceptNet that are related to the node of the concept e_i (resp. e_j). The more common node e_i and node e_j share, the more they are considered to be (semantically) similar.

Second, for Wikipedia, we use the outlinks, categories, and the set of terms that co-occur with the original query or a part of the query. In cases where no Wikipedia pages match the query or a part of the query, we use Explicit Semantic Analysis (ESA)[12] to get semantically related Wikipedia pages, from which to extract the outlinks, categories and representative terms to obtain a set of candidate expansion terms. The similarity function $sim_W(e_i, e_j)$ between two expansion terms e_i and e_j based on Wikipedia is defined by Formula 6, where W_i (W_j) is the set of vectors containing term e_i (e_j) obtained by ESA, and $sim(w_i, w_j)$ is simply the cosine similarity of vector w_i and w_j.

$$sim_W(w_i, e_j) = \frac{1}{|W_i||W_j|} \sum_{w_i \in W_i, w_j \in W_j} sim(w_i, w_j) \tag{6}$$

Third, for query logs, the set of candidate expansion terms includes the queries that share the same click-through data with the original query, as well as the reformulated queries that appear in a user session within a 30 minutes-time window. The similarity function $sim_{QL}(e_i, e_j)$ between two expansion terms e_i and e_j based on query logs is defined by Formula 7 :

$$sim_{QL}(e_i, e_j) = \frac{|Q_{e_i} \cap Q_{e_j}|}{|Q_{e_i} \cup Q_{e_j}|} \tag{7}$$

where Q_{e_i} (resp. Q_{e_j}) is the set of reformulated queries of the original query which contain term e_i (resp. term e_j).

Fourth, for initial search results, we consider top K returned results as relevant documents (K is experimentally set to 50 in our experiments), and use Pseudo-Relevance Feedback (PRF) to generate the set of candidate expansion terms from feedback documents. The similarity function $sim_D(e_i, e_j)$ between two expansion terms e_i and e_j based on the feedback documents is defined by Formula 8 :

$$sim_D(e_i, e_j) = \frac{2 \cdot freq(e_i, e_j)}{\sum_{e'} freq(e_i, e') + \sum_{e'} freq(e_j, e')} \tag{8}$$

where $freq(e, e^{'})$ refers to the co-occurrence of term e and $e^{'}$ within a fixed window of size 15.

Finally, for the fifth resource (query suggestions), we use an additional similarity function that we defined as follows: The similarity between two candidate expansion terms e_i and e_j using the query suggestions (denoted hereafter by $sim_{QS}(.,.)$), is computed as follows:

$$sim_{QS}(e_i, e_j) = \frac{2 \cdot n(e_i, e_j)}{n(e_i) + n(e_j)} \tag{9}$$

where $n(e_i)$ (resp. $n(e_j)$) is the number of times term e_i (resp. e_j) appears in the query suggestions, and $n(e_i, e_j)$ is the number of times when both terms e_i and e_j appear in the set of suggestions for the same query.

3. SUBTOPIC MINING SUB-TASK

3.1 Query Classification

The first step in this sub-task is to classify a query into one of the three target classes: *ambiguous*, *broad* or *clear*. For this purpose, we learned a SVM classifier by collecting a set of features. Table 1 describes the set of features that we consider in this paper.

We organize our features into *query-dependent* features and *query-independent* features. While the former are computed on-the-fly at querying time, the latter are computed on offline (*i.e.* at indexing time).

For the query-dependent features, we simply consider two features: the number of query terms (`NumTerms`) and a boolean value reflecting whether the query is a question or not (`Quest`). The idea behind using the first feature is that ambiguous queries tend to be short, following previous works such as [17] which shows that the average length of ambiguous queries is about one word. Broad and clear queries, however, are well formatted queries that contain several words (which

Table 1: All features computed in this work for automatically classifying queries. (Here, q denotes a query that we want to classify into *clear*, *broad* or *ambiguous* and D denotes the set of top 50 retrieval results of q).

Feature	Description	Total
***** Query-dependent:***		
`NumTerms`	Number of terms of q after removing stopwords	1
`Quest`	Whether q is a question?	1
***** Query-independent:***		
** Feedback documents-based:*		
`SongEtAl`	The set of 11 features described in [19] (excepting the feature about the number of terms in q)	11
`AvgPMI`	Average mutual information score between the terms of q and the top 10 terms that co-occur a lot with the terms of q in D	1
`ClarityScore`	Clarity score of q computed on D and the whole collection [9]	1
** Wikipedia-based:*		
`NumInterp`	Number of (possible) interpretations of q in the Wikipedia disambiguation page of q	1
`WikiLength`	Wikipedia page length (number of *different* words) that matches with q	1
** Query logs-based:*		
`NumClicks`	Max, Min and average number of clicked URLs for q in all the sessions	3
`PercentageClicks`	Percentage of shared clicked URLs between different users who issued q	1
`ClickEntropy`	Click entropy of the query q [11]	1
`NumSessions`	Total number of sessions with q	1
`SessionLength`	Max, Min and average session duration (in seconds) with q	3
`NumTermsReform`	Total number of different terms added by users to reformulate q in all the sessions	1
`ReformLength`	Max, Min and average number of terms added by users to reformulate q in all the sessions	3
** ConceptNet-based:*		
`NumDiffNodes`	Number of different adjacent nodes that are related to the nodes of the graph of q	1
`AvgCommonNodes`	Average number of common nodes shared between the nodes of the graph of q (*i.e.* nodes that are connected to at least two edges)	1
`NumDiffRelations`	Number of different relation types defined between the adjacent nodes in the graph of q	1
Grand Total		**33**

helps to make a clearer description of the user). Concerning the second feature, we hypothesize that if a query is a question, the user is more likely looking for a narrow and specific answer. For instance, by issuing a query like "what is a natural number" (query #95), the user is looking for a precise definition of a natural number, and the user intent behind such kind of queries is clear. However, this hypothesis requires a further investigation in the future to be confirmed.

For the query-independent features, we take advantage of several heterogeneous resources to derive different features, and we define four resource-based features accordingly. It is worth noting that, for some queries that do not appear in some resource, we cannot extract any feature for that query from the resource. In that case, the corresponding features' values are set to 0. Some of the features from feedback document are from the previous work of Song et al. [19]. We used the same experimental setting described in [19] in order to categorize the top 50 returned documents returned for a given query. The idea is that, classifying a query consists of categorizing the feedback documents of that query into at least one of the predefined categories [5]. Basically, it is expected that feedback documents of ambiguous queries correspond to more than one category and those of clear queries correspond to solely one category. Also, we argue that the `ClarityScore` feature introduced in [9] is an important feature in our query classifier, since this is an indicator of the ambiguity level of a query.

Concerning Wikipedia, we derive solely two features: `NumInterp` and `WikiLength` which compute the number of possible interpretations of q in the Wikipedia disambiguation page, and the number of different words that appear in the Wikipedia page of q, respectively. The first feature (`NumInterp`) is mainly used to estimate whether a query is ambiguous or not. Indeed, an ambiguous query usually has different interpretations, and in most of the cases, it has different interpretations in the corresponding Wikipedia disambiguation page. The idea behind the use of the second feature (`WikiLength`) is that, the more a Wikipedia page suggests *different* words (which are candidate expansion terms for q), the less clear the query is. This is because, when the query is clear, there is a limited number of possible aspects to which the user is looking for. Consequently, the number of expansion terms regarding to that query should also be limited.

For the features based on query logs, we consider the query reformulations, the click-through data and the query sessions to derive classification features. Indeed, when the query is ambiguous, different users have different information needs, and thus, their behaviors are diverse, *e.g.*, the clicked URLs from one user may be very different from those clicked by another user, because these users give different in-

[5] We use the 29 predefined categories described in `http://www.ccc.ipt.pt/~ricardo/datasets/GISQC_DS.html`

terpretations to the same (ambiguous) query. In contrast, when the query is clear, it is generally expected that most of the users are seeking for the same information need, yielding to almost the same clicked URLs. For instance, for the feature `PercentageClicks`, the more there are different users sharing the same clicked URLs for the query, the more the considered query is clear, and vice versa. Also, the less users added terms to reformulate their query, the more this query is clear.

Finally, the three considered features based on ConceptNet are calculated on the graph of the query (if the query appears in ConceptNet). The idea of deriving such features is from our observations when comparing the graphs of ambiguous, broad and clear queries in ConceptNet. In particular, we found that the nodes in the graph of ambiguous queries are generally unconnected. This is intuitive because they correspond to different pieces of information that are not related (apart from the common ambiguous word). For example, the two interpretations 'programming language' and 'island' of the same ambiguous query "java" do not share any common node in the graph of ConceptNet. On the other hand, we also observed that the nodes in the graph of broad and clear queries are often connected because they share some common aspects.

3.2 Query Disambiguation and Predicting Subtopic Importance

Once we classified each query into one of the three target classes (ambiguous, broad, clear), we now disambiguate the ambiguous queries to generate the first level of subtopics, which corresponds to the query interpretations. Following existing works in literature which show the usefulness of Wikipedia for disambiguation task (such as [6, 17, 20]), we also use Wikipedia disambiguation pages, together with query logs, to find the query interpretations. First, we match each ambiguous query with its Wikipedia disambiguation page. This leads to a first set of query interpretations. However, only a few of these interpretations correspond to the user interest. Therefore, if the query appears in the log (which is the case of the ambiguous queries that we consider in this work), then we collect all the expansion terms added by users in their sessions to reformulate the query and match these terms with the Wikipedia disambiguation pages.

We now compute a score for each Wikipedia disambiguation page regarding to the original query, as follows:

$$score_q(i) = \frac{n_q(i)}{|E_q|} \tag{10}$$

where q is the original (ambiguous) query, E_q is the set of different expansion terms added by users in the logs to reformulate q, and $n_q(i)$ is the number of terms from E_q that appear in the i^{th} Wikipedia page of q. Of course, we normalize these scores for all disambiguation pages corresponding to query q to make them comparable.

At the end, we keep an interpretation of q if its score is higher than a threshold value (in our experiments, we set this threshold value to 0.1). If more than five interpretations have scores higher than our threshold, we keep only the first five strongest interpretations since at most five interpretations should be returned. We call each possible interpretations of q a 'sub-query'.

If q is broad, we directly apply our embedding framework described in Section 2 to estimate the aspect relative importance. If q is ambiguous, we systematically apply our embedding framework for each sub-query of q. Finally, the interpretation relative importance of each ambiguous query is estimated using Equation 10. Since there is no need to return aspects for clear queries (as suggested by the NTCIR organizers), we decide not to apply our embedding framework for that kind of queries.

4. DOCUMENT RANKING SUB-TASK

The second sub-task to which we participated this year is document ranking, which consists of returning a diversified ranked list of documents, for each query. We adopt a selective diversification strategy, depending on the query class (ambiguous, broad, clear). First, if the query is clear, we do not diversify results and simply rely on the documents returned by a standard retrieval model (such as KL language model). This strategy is used because we observed that diversifying search results for clear queries does not improve search results, and instead can decrease the performance of search results in terms of relevance and diversity [6]. Second, we expand each broad query by using the set of expansion terms (with their weights) that we obtained by our embedding framework, and report the search results of these expanded queries using Indri. Finally, for ambiguous queries, we report the search results of each sub-query after being expanded using our embedding framework, and then combine the different sets of documents into a single one. At each iteration, we greedily select the document d^* according to this formula:

$$d^* = argmax_{d \in \cup D_i - S}\left(\frac{rel(d) \ \cdot \ score_q(i)}{rank(d)}\right) \tag{11}$$

where D_i is the set of documents corresponding to the i^{th} sub-query; S is the set of documents already selected; $score_q(i)$ is the score computed for the i^{th} sub-query according to Equation 10; $rank(d)$ is the rank of document d in its set; and $rel(d)$ denotes the normalized relevance score of document d [7].

5. EXPERIMENTAL SETTING

Document collection and query set. We indexed the collection of documents ClueWeb12-B13 [8] by using the Indri/Lemur toolkit. Both the index and the queries were stopped using the INQUERY stoplist, without stemming. Table 2 reports some statistics of our index. Our baseline is KL language model with Dirichlet smoothing ($\mu = 2000$). The test topics consists of a set of 50 English queries collected from both Sogou and Bing search logs. These queries are from different classes (ambiguous, broad, and clear).

Used resources. In our participation of this year, we used five different resources that were available for us to collect expansion terms for subtopic mining sub-task:

[6] To make such observation, we considered a subset of queries from TREC 2009, 2010 and 2011 Web tracks, that we judged as clear queries by using our classifier, and compared the performance of search results for these queries, before and after diversification. Overall, we observed that diversifying search results for clear queries does not really improve the overall performance.

[7] We use the language-model normalization formula, that is $x = \frac{exp(x)}{\sum_{x'} exp(x')}$.

[8] `http://lemurproject.org/clueweb12`

Table 2: Statistics of our index.

Documents	52,343,021
Unique Terms	163,310,576
Total Terms	39,795,552,546
Size (compressed)	389 GB
Size (uncompressed)	1.95 TB

(1) The last version of ConceptNet [9] which encompasses 8.7M assertions shared between 3.9M nodes;
(2) The English Wikipedia dumps of July 8th, 2013 which contains 4.3M English articles;
(3) The log data of Microsoft Live Search 2006, which spans over one month (starting from May 1^{st}) consisting of almost 14.9M queries shared between around 5.4M user sessions;
(4) The top 50 results returned for the original query (feedback documents);
(5) The query suggestions from Bing, Google and Yahoo! provided by the NTCIR-10 organizers for INTENT2 [10], as an additional resource. Note that the training query topic set of INTENT2 is completely different from the test queries for NTCIR-11 IMine task of this year, which makes difficult to learn much from the suggestions collected last year for a different set of queries. Despite this difference between training and test topics, we used this data yet to assess whether query suggestions can help to improve our results.

Query classification. For the query classification task, we used SVM-Light [11] which provides an implementation of a non linear SVM (with RBF kernel). During these experiments, the parameters for SVM are set to their default values. When learning our query classifier, we used the set of features that we already defined (see Table 1), and use the publicly available set of 450 training queries [12].

Parameter Setting. In our model, we have a unique parameter than should be set, that is β (in Formula 3) which controls the trade-off between relevance and non-redundancy when selecting expansion terms from different resources. This parameter is determined by using the query sets from TREC 2009 and TREC 2010 Web tracks for training while the query set from TREC 2011 Web track is used for test. During this procedure, we optimize for α-nDCG [8] at cutt-off 10. Note that the relevance judgments for these three query sets are available by TREC assessors, and that these topics use the ClueWeb09 (category B) as a document collection.

6. RESULTS

In this section, we describe the three runs that we submitted for each sub-task, then we make a comparison between these runs and discuss our results.

6.1 Description of the Runs

During our participation, we adopted three methodologies, and submit one run for each methodology for subtopic mining. Each of the three methodologies leads to a different set of expansion terms for each test query. Each submitted run for document ranking sub-task corresponds to the retrieval results of the expanded query resulted in the subtopic mining sub-task. So there is a strict relation between the submitted runs in the two sub-tasks (*E.g.*, ”UM13-D-E-1A” run for document ranking sub-task matches with ”UM13-S-E-1A” run for subtopic mining sub-task).

The three runs are as follows:
(1) 1^{st} *run* :: **UM13-S-E-1A** (**UM13-D-E-1A**): We use an explicit modeling of query aspects based on embedding, by incorporating five resources: query logs, Wikipedia, ConceptNet, documents feedback, and the query suggestions from Bing, Google, and Yahoo!.
(2) 2^{nd} *run* :: **UM13-S-E-2A** (**UM13-D-E-2A**): We use an explicit modeling of query aspects based on embedding, by incorporating four resources (query logs, Wikipedia, ConceptNet, documents feedback).
(3) 3^{rd} *run* :: **UM13-S-E-3A** (**UM13-D-E-3A**): We use an explicit modeling of query aspects based on embedding, by incorporating one single external resource (query logs).

6.2 Performance of our Query Classifier

In this section, we evaluate the performance of our classifier. To do that, we use the standard metrics of *precision*, *recall* and *F1-measure.* In the query classification context, these metrics are defined as follows:

$$Precision_x = \frac{Number\ of\ queries\ correctly\ tagged\ as\ x}{Number\ of\ queries\ tagged\ as\ x} \tag{12}$$

$$Recall_x = \frac{Number\ of\ queries\ correctly\ tagged\ as\ x}{Number\ of\ queries\ whose\ class\ is\ x} \tag{13}$$

$$F1_x = \frac{2 \cdot Precision_x \cdot Recall_x}{Precion_x + Recall_x} \tag{14}$$

Here, x could be one of the three target classes, *i.e.*, *ambiguous* or *broad* or *clear.* In Table 3, we report the performance of our classifier in terms of precision, recall and F1-measure for each of the three target classes.

Table 3: Performance of our query classifier.

Query Class (x)	$Precision_x$	$Recall_x$	$F1_x$
ambiguous (a)	75.00%	56.30%	64.30%
broad (b)	44.83%	76.47%	56.52%
clear (c)	88.89%	47.06%	61.54%

From these statistics, we observe that our classifier can correctly classify about 60% of the queries. In particular, we found out that our classifier *fails* to distinguish between broad and clear queries, and it can better distinguish ambiguous queries from non-ambiguous ones. Our observation is in line with the existing work in query classification, such as [19]. Indeed, a clear query (*e.g.*, query #95: “what is a natural number”) has a specific meaning and overs a narrow topic (*e.g.*, the precise and specific definition of a natural number for the query #95). A broad query (*e.g.*, query #70: “lost season 5”) covers a variety of subtopics, but these subtopics tend to overlap in general since they share

[9] `http://conceptnet5.media.mit.edu`
[10] `http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings10/NTCIR/Evaluations/INTENT/ntc10-INTENT2-eval.htm`
[11] `http://svmlight.joachims.org`
[12] `http://www.ccc.ipt.pt/~ricardo/datasets/GISQC_DS.html`

an amount of information (*e.g.*, aspects 'series', 'download' and 'information' for the query #70 are almost about the same user intent, which is *downloading information about the series of lost season 5*). This overlap between the aspects of a broad query make the behavior of a broad similar to that of a clear query, which explains why our classifier fails to distinguish between broad and clear queries. These observations, however, need a further and deeper investigation in the future to be confirmed. Also, we intend to better improve the performance of our classifier in the future by collecting more important features.

6.3 Results and Discussion

The results of our three runs for subtopic mining sub-task and document ranking sub-task are reported in Table 4 and Table 5, respectively. The results are computed on a set of evaluation metrics proposed for subtopic mining [15]. These metrics include *Hscore* which measures the quality of the hierarchical structure, *i.e.*, whether the second-level subtopic is correctly assigned to the first-level one; *Fscore*, which measures the quality of the first-level subtopic, *i.e.*, whether the submitted first-level subtopics are correctly ranked and whether all important first-level subtopics are found; *Sscore*, which measures the quality of the second level subtopics; and finally *H-measures*, which is a combination of the three previous metrics (*Hscore*, *Fscore* and *Sscore*) [15].

Table 4: Results of our three runs for subtopic mining sub-task (over 33 unclear topics).

	Hscore	*Fscore*	*Sscore*	*H-measure*
UM13-S-E-1A	0.2056	**0.1624**	**0.0059**	0.0047
UM13-S-E-2A	**0.2064**	**0.1624**	**0.0059**	**0.0049**
UM13-S-E-3A	0.1766	**0.1624**	0.0049	0.0037

Table 5: Results of our three runs for document ranking sub-task (over all 50 topics).

	Coarse-grain results	*Fine-grain results*
UM13-D-E-1A	**0.6254**	**0.5566**
UM13-D-E-2A	0.6001	0.5309
UM13-D-E-3A	0.4474	0.3770

From Table 4, we observe that the best performance for subtopic mining is obtained by the second run. However, the difference compared to the first run is really small. For example, in both first and second run, we obtained the same *Fscore* and *Sscore*, and the difference in term of *Hscore* and *H-measure* is 0.0008 and 0.0002, respectively. Recall that in both runs, we used the four typical resources (ConceptNet, Wikipedia, query logs and feedback documents), but in the first run, query suggestions are used as well. From these results, it seems that more resources do not necessarily guarantee better results. A possible reason is that when more resources are used, the risk of selecting noise expansion terms (*i.e.*, aspects) becomes higher. Maybe, our observation that query suggestions do not help is due to the fact that the training and test topic sets are different, which makes difficult to learn much from such queries. However, compared with the third run, which only uses one resource, we see that more resources generally lead to better results. Indeed, when a single resource is used, it becomes difficult to ensure a good coverage of the different aspects of the query. Finally, it is worth noting that the results that we obtained for subtopic mining are very sensitive to the quality of our query classifier. For example, if an ambiguous query is labeled as non-ambiguous (*i.e.*, broad or clear) by our classifier, then *H-score* will be low for such query due to the low quality of the hierarchical structure, and consequently, a low value of *S-score*.

Figure 1: Example of a topic ("windows") along with its first and second level subtopics.

```
<topic number="61" type="ambiguous">
  windows software;1;0.64;0;update;1;0.96;
  windows software;1;0.64;0;installer;2;0.91;
  windows software;1;0.64;0;8;3;0.89;
  windows software;1;0.64;0;versions;4;0.82;
  windows software;1;0.64;0;license;5;0.75;
  windows software;1;0.64;0;defender;6;0.63;
  windows software;1;0.64;0;replacement;7;0.61;
  windows software;1;0.64;0;recovery;8;0.54;
  windows software;1;0.64;0;vista;9;0.41;
  windows software;1;0.64;0;live;10;0.29;
  windows house;2;0.35;0;catalog;1;0.90;
  windows house;2;0.35;0;treatment;2;0.82;
  windows house;2;0.35;0;glass;3;0.81;
  windows house;2;0.35;0;paint;4;0.78;
  windows house;2;0.35;0;pictures;5;0.77;
  windows house;2;0.35;0;construction;6;0.63;
  windows house;2;0.35;0;sizes;7;0.55;
  windows house;2;0.35;0;tinting;8;0.49;
  windows house;2;0.35;0;manufacturer;9;0.31;
  windows house;2;0.35;0;pulls;10;0.27;
</topic>
```

Figure 1 shows an example of a topic ("windows") along with its identified aspects. The aspects were obtained using our first run. From Figure 1, we observe that this topic has two level subtopics. The first level corresponds to the different interpretations of the (ambiguous) query, that is *software* and *house*, with their corresponding weights 0.64 and 0.35, respectively. For each of the first level subtopics, we identified 10 aspects, accordingly with their weights. *E.g.*, *vista* is the 9^{th} aspect of query "windows" which is related to the first level subtopic *software*, and its weight is 0.41.

For the document ranking sub-task to which we participated, the best performance were obtained using the first method (see Table 5), and again, we observed that using several resources leads to a better results than using a single one. For a further understanding of the performance of each method, we also report in Table 6 and Table 7 more detailed statistics of the three runs for document ranking, over 33 unclear queries.

For both coarse-grain and fine-grain results, our first run outperforms the two other ones in all adhoc and diversity measures (excepting in *ERR* measure in which the second run provides a better score than the first one, but the difference is relatively small). In particular, *I-rec@10* measures the subtopic recall which evaluates the proportion of subtopics covered by documents. For this measure, we obtained a good coverage of aspects by using our first and second method (about 80% and 75% in coarse-grain results, and 63% and 59% in fine-grain results).

Table 6: Detailed coarse-grain results of our three runs for document ranking sub-task (over 33 unclear topics).

	AP@10	*RBP*	*nDCG@10*	*ERR@10*	*I-rec@10*	*D#-nDCG@10*
UM13-D-E-1A	**0.5479**	**0.1655**	**0.5108**	0.4236	**0.7899**	**0.6511**
UM13-D-E-2A	0.4782	0.1489	0.4750	**0.4251**	0.7520	0.6137
UM13-D-E-3A	0.2520	0.1025	0.3162	0.2880	0.5692	0.4397

Table 7: Detailed fine-grain results of our three runs for document ranking sub-task (over 33 unclear topics).

	AP@10	*RBP*	*nDCG@10*	*ERR@10*	*I-rec@10*	*D#-nDCG@10*
UM13-D-E-1A	**0.5479**	**0.1480**	**0.4629**	**0.2628**	**0.6310**	**0.5469**
UM13-D-E-2A	0.4782	0.1340	0.4301	0.2602	0.5874	0.5089
UM13-D-E-3A	0.2520	0.0915	0.2901	0.1807	0.3798	0.3331

7. CONCLUSION AND FUTURE WORK

In this paper, we described our participation of this year to the NTCIR-11 IMine task, for both subtopic mining and document ranking sub-tasks. We experimented with a new approach for diversified query expansion based on embedding which selects (good) expansion terms from a set of resources. Related expansion terms are grouped together since they correspond to the same semantic aspect. Our results suggest that using more resources can generally yield better results, especially when ranking documents. In particular, in our best run for document ranking, we obtained good results in both adhoc and diversity results, and we were ranked No. 2 among all 15 runs.

Several issues could be investigated in our future work. For example, the resources with which we experimented were equally weighted. However, in practice different resources should have different weights depending on the query. Also, in our experiments on subtopic mining, we found that query suggestions is not so helpful to improve the performance of our model in terms of selecting good expansion terms for the query. However, further investigation is required to understand the reasons.

8. REFERENCES

[1] M. Bendersky, D. Metzler, and W. B. Croft. Effective query formulation with multiple information sources. In *Proc. of WSDM*, pages 443–452, Washington, USA, 2012.

[2] D. M. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[3] L. Bottou. Stochastic learning. In *Advanced Lectures on Machine Learning*, Lecture Notes in Artificial Intelligence, pages 146–168. Springer Verlag, Berlin, Germany, 2004.

[4] A. Bouchoucha, J. He, and J.-Y. Nie. Diversified query expansion using conceptnet. In *Proc. of CIKM*, pages 1861–1864, Burlingame, USA, 2013.

[5] A. Bouchoucha, X. Liu, and J.-Y. Nie. Integrating multiple resources for diversified query expansion. In *Proc. of ECIR*, pages 98–103, Amsterdam, Netherlands, 2014.

[6] R. Bunescu and M. Pasca. Using encyclopedic knowledge for named entity disambiguation. In *Proc. of EACL*, pages 9–16, Trento, Italy, 2006.

[7] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking documents and producing summaries. In *Proc. of SIGIR*, pages 335–336, 1998.

[8] C. L. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proc. of SIGIR*, pages 659–666, Singapore, Singapore, 2008.

[9] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *Proc. of SIGIR*, pages 299–306, New York, NY, USA, 2002. ACM.

[10] R. Deveaud, E. SanJuan, and P. Bellot. Estimating topical context by diverging from external resources. In *Proc. of SIGIR*, pages 1001–1004, New York, USA, 2013.

[11] Z. Dou, R. Song, and J.-R. Wen. A large-scale evaluation and analysis of personalized search strategies. In *Proc. of WWW*, pages 581–590, New York, NY, USA, 2007.

[12] E. Gabrilovich and S. Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proc. of IJCAI*, pages 1606–1611, San Francisco, USA, 2007.

[13] J. He, V. Hollink, and A. de Vries. Combining implicit and explicit topic representations for result diversification. In *Proc. of SIGIR*, pages 851–860, New York, USA, 2012.

[14] X. Liu, A. Bouchoucha, A. Sordoni, and J.-Y. Nie. Compact aspect embedding for diversified query expansions. In *Proc. of AAAI'14*, pages 115–121, 2014.

[15] Y. Liu, R. Song, M. Zhang, Z. Dou, T. Yamamoto, M. Kato, H. Ohshima, and K. Zhou. Overview of the ntcir-11 imine task. In *Proc. of NTCIR'14*, page (pages to appear), Tokyo, Japan, 2014.

[16] G. A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, Nov. 1995.

[17] M. Sanderson. Ambiguous queries: Test collections need more sense. In *Proc. of SIGIR*, pages 499–506, New York, NY, USA, 2008. ACM.

[18] R. L. Santos, C. Macdonald, and I. Ounis. Exploiting query reformulations for web search result diversification. In *Proc. of WWW*, pages 881–890, Raleigh, USA, 2010.

[19] R. Song, Z. Luo, J.-Y. Nie, and H.-W. Hon. Identification of ambiguous queries in web search. *Information Processing and Management*, 45(2):216 – 229, 2009.

[20] M. Strube and S. P. Ponzetto. Wikirelate! computing semantic relatedness using wikipedia. In *Proc. of AAAI*, pages 1419–1424. AAAI Press, 2006.

[21] S. Vargas, R. L. T. Santos, C. Macdonald, and I. Ounis. Selecting effective expansion terms for diversity. In *Proc. of OAIR*, pages 69–76, Paris, France, 2013.