Learning to Generate Diversified Query Interpretations using Biconvex Optimization

Ramakrishna B Bairi
IITB-Monash Research Academy
IIT Bombay
Mumbai, India, 400076
bairi@cse.iitb.ac.in

Ambha A
IIT Bombay
Mumbai, India, 400076
ambha.career@gmail.com

Ganesh Ramakrishnan
IIT Bombay
Mumbai, India, 400076
ganesh@cse.iitb.ac.in

Abstract

The wealth of information present in the World Wide Web has made internet search a de-facto medium for obtaining any required information. Users typically specify short and/or ambiguous queries and expect the answer to appear at the top. Hence, it can be extremely important to produce a diverse but relevant set of results in the precious top k positions. This calls for addressing two types of needs: (i) producing relevant results for queries that are often short and ambiguous and (ii) selecting a set of k diverse results to satisfy different classes of information needs. In this paper, we present a novel technique using a Biconvex optimization formulation as well as adaptations of existing techniques from other areas, for addressing these two problems simultaneously. We propose a graph based iterative method to choose diversified results. We evaluate these approaches on the QRU (Query Representation and Understanding) dataset used in SIGIR 2011 workshop as well as on the AMBIENT (Ambiguous Entities) dataset and present results on generating diversified query interpretations. We also compare these approaches against other online systems such as *Surf Canyon, Carrot2, Exalead and DBpedia* and empirically demonstrate that our system produces competitive results.

1 Introduction

The growth of internet has resulted in the proliferation of electronic documents on the World Wide Web. Every search engine, be it generic or application and domain specific, serves as a portal to access these documents. User queries, in general, are short and often tend to be ambiguous and/or underspecified. In addition, a query can have multiple *concealed interpretations*. For example, *Sun* could be interpreted as "The sun as a star", "Composition of Sun", "Sun Micro systems company", "Sun news paper", "Sun Record music company", and so on. We believe that, in addition to these concealed interpretations, *related interpretations* are also equally important. As examples, "Solar Cells" and "Photosynthesis", could be interpretations related to this query. To improve user interaction and to guide him/her in further refining the query, it could help if the search engine generated these relevant interpretations as well. Due to the sheer size of online information and its diversity, the possible interpretations to a short query are enormous. In addition, users expect their intended answer to be present in the top few search results. This calls for presenting a diversified but relevant set of results in the top k positions. Note that, in this paper we consider the diverse search results produced by the search system as *interpretations* of the query in some sense. In addition, we consider each search result is a document describing some aspect related to the query. Hence we restrict our notion of interpretation to each such *document* in the search result.

We present an original method as well as adaptations of some existing methods to solve this problem. As for our proposed method, we construct an interpretation graph with potential interpretations as its nodes and edges indicating their similarity. Inspired by the works on GCD (Dubey et al., 2011) and MMR (Carbonell and Goldstein, 1998), we develop a new technique for diversity ranking of interpretations. As part of this technique, we propose an algorithm (Rel-Div) to learn the node and edge weights of the interpretation graph iteratively by solving a biconvex optimization (Gorski et al., 2007) problem. At query time, we solve a convex optimisation problem to choose

International Joint Conference on Natural Language Processing, pages 733–739,
Nagoya, Japan, 14-18 October 2013.

k diverse nodes and present them as interpretations to the user query. We identify interpretations relevant to the query using a publicly available internet encyclopedia. Though we used Wikipedia as the source, we believe that the repository can be easily extended to accommodate other catalogs like YAGO and Freebase.

We compare our diversification approach with other diversification approaches (which were applied not necessarily to solve the same problem as ours) such as variants of GCD (Dubey et al., 2011), Affinity Propagation (Frey and Dueck, 2006),(Frey and Dueck, 2007). We evaluated results on benchmark queries from the SIGIR 2011 workshop's QRU (Query Representation and Understanding) dataset and the AMBIENT data sets. In addition, we compare the diversity of interpretations generated by these approaches against those of other online systems such as Surf Canyon, Carrot2, Exalead and DBpedia (URLs of all these systems listed under References)

We summarize our contributions as: 1) Top-K diversity ranking using a graph based approach. 2) Iterative Graph weight learning technique - A new iterative technique for learning the node and edge weights for an interpretation graph by solving a biconvex optimisation problem.

The rest of the paper is organized as follows: In Section 2 we present related work. In Section 3 we describe our technique of iterative graph weight learning and diversity ranking. In Section 4 we demonstrate the utility of our technique by applying it to the interpretation generation task from Wikipedia. In Section 5, we present experimental evaluations. We conclude our work in the subsequent section.

2 Prior work

Most of the prior research has focused on generating diversified result urls. The approach presented by (Swaminathan et al., 2009) filters initial search results and covers diversified topics based on bag of words measures. Yisong and Joachims (Yue and Joachims, 2008) train a model using Struct SVM and encode diversity as a penalty function (this is penalty for not covering certain topics). Most recently, Brandt et al.(Brandt et al., 2011) and Raman *et. al.* (Raman et al., 2011) proposed an approach for *dynamic ranking* and then group URLs with similar intentions.(Dubey et al., 2011) formulate the problem of ensuring diversity as that of identifying relevant urls which are most likely to be visited by the random surfer. We propose a new approach for interpretation generation. The report (Hearst, 2006) by M.A Hearst claims that clustering based on similarity measure may not always result in meaningful interpretations or labels. So, instead of dynamically generating labels, we pick labels or relevant interpretations for a query from the pool of labels. We use Wikipedia as a primary source to capture these interactions along with their semantic relations. (Hahn et al., 2010),(Ben-Yitzhak et al., 2008) produce Wikipedia pages as search results and align the search results along a set of fine grained attributes/facets. In our work, facets (which we refer to as interpretations) are neither predefined nor necessarily fine grained. Moreover, as we will see, our interpretations need not be restricted to Wikipedia entities. Closest to our approach is the approach of (Ma et al., 2010). They apply page ranking technique on the graph constructed using query log statistics to obtain diversified interactions.

3 Diversified Interpretation Generation

3.1 Our Problem

Given a large corpus U of documents and a short user query q, we define a function $H(q, U)$ that returns a subset of documents $S = \{e_1...e_n\} \subseteq U$, satisfying the query q. The function $H(q, U)$ acts as a filtering function to retrieve the documents S that are syntactically and/or semantically related to the query q. In its simplest form, $H(q, U)$ can just return U without performing any filtering, which is not generally useful. It is important to design an $H(q, U)$ (*e.g.*, keyword based lookup, semantics matching, *etc.*) that can help reduce the search space in a meaningful manner. Our goal is to choose a set of k documents from S and we assume that to best satisfy the user intention, these k documents presented to the user should be diverse yet highly relevant to the query q.

3.2 The Training Algorithm

We expect groups of documents in S to be related to each other via some semantic relations. We initially construct a document-relation graph using $e_1...e_n$. We refer to this graph as an *Interpretation Graph*, since the documents in this graph are obtained as various interpretations of the query. While the nodes are documents from S, each edge is a relation between the documents. A relation

could be one of synonymy, hyponymy, meronymy, homonymy, *etc.*. These relations could be obtained from external catalogs such as Wikipedia, Wordnet, *etc.*

Each node in the graph is assigned a score which represents the relevance of the node to the query. We use the notation b_q to represent the column vector (of size $n \times 1$) containing all the node relevance scores. The weight on an edge represents the degree of similarity between the two nodes connected by that edge. We use the notation C_q (of size $n \times n$) to represent the matrix of edge scores reflecting similarity between pairs of nodes. Note that, each column C_q^i of the matrix C_q represents an document e_i and the cell values in that column indicate the similarity of document e_i with other documents. The scores in b_q are used to ensure that the subset of k interpretations are relevant to q, whereas the similarity scores in C_q are used to ensure diversity in the subset of k interpretations.

We assume that we are provided training data, consisting of queries and their correct interpretations. Our goals in training are to 1) develop a model for the node score b_q, 2) develop a model for the edge potentials C_q and 3) learn parameters of these models such that the set of k relevant yet diverse nodes obtained from the graph using b_q and C_q are consistent with the training data. Thus, implicit in our third goal is the following subproblem, which is also our query time inference problem: 4) compute a subset of k best interpretations using b_q and C_q, that represent k diverse, but relevant interpretations. A part of the graph for the query "sun" is depicted in Figure 1

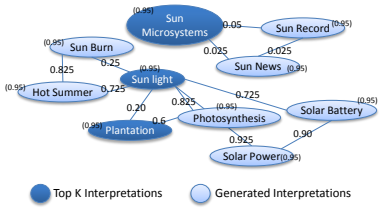


Figure 1: Interpretation Graph for the query *Sun*

3.2.1 Modeling node potentials (b_q)

In order to build a learning model for b_q, it is important to define a good set of features that characterize the node's relevance to the query. Let $N_{1..|N|}(q, S)$ be a set of $|N|$ query independent node features. Each feature $N_f(q, S)$ evaluates the relevance of documents in S to the query q and returns a vector of scores. These feature functions are problem specific and crafted carefully to bring out the relevance between query and documents (such as term overlaps, n-gram matches, etc). In Section 4 we provide some practical examples of node features.

The node potential vector b_q is obtained by combining the scores returned by individual feature functions $N_f(q, S)$. One of the obvious choices is to use Logistic Regression (Yan et al., 2003).*i.e.* $b_q[i] = \frac{1}{1+e^{-\sum_{f=1}^{|N|} w_f N_f(q,S)[i]}}$. The weight vector $W^T = [w_1 ... w_{|N|}]$ is learnt through supervised training explained in Section 3.2.3.

3.2.2 Modeling edge potentials (C_q)

To learn the edge potentials, it is important to define a good set of features that measure the similarities between every pair of nodes and return similarity scores. Higher the score, more similar are the nodes. Let $C_{1..|C|}(S)$ be the set of $|C|$ edge features that evaluate similarities between documents in S and each returns a $n \times n$ matrix of scores. These feature functions are problem specific and crafted carefully to bring out the similarities between the documents. In Section 4 we provide some practical examples of edge feature construction using Wikipedia.

The edge potential matrix C_q is obtained as $C_q = \sum_{f=1}^{|C|} \lambda_f C_f(S)$ where $0 \leq \lambda_f \leq 1$ and $\sum \lambda_f \geq 1 \ \forall f$. The weight vector $\lambda^T = [\lambda_1 ... \lambda_{|C|}]$ is learnt through supervised training explained in Section 3.2.3.

3.2.3 Learning feature weights W^T, λ^T

Proposition 1:

$$b_q \approx \sum_{j=1}^{k} \tilde{C}_q^{i_j} \quad (1)$$

for sufficiently large k diverse documents, where, $\tilde{C}_q$ is the matrix $C_q$with the columns scaled so that the diagonal cell values match the relevance value, *i.e.*, $\tilde{C}_q(i,i) = b_q(i)$. The values $i_1 ... i_k$ represent indices of k columns of matrix $\tilde{C}_q$. Hence, $\tilde{C}_q^{i_j}$ is the i_jth column of matrix $\tilde{C}_q$.

The intuition behind this approximated equality comes from the fact that, two similar documents should have similar relevance score with the query and we are interested in selecting k diverse documents. Let e_i be one of these k diverse documents. If the documents $e_{j_1} ... e_{j_p}$ are similar to e_i, then, $b_q[i] \approx b_q[j_1] \approx ... \approx b_q[j_p]$ and $C_q[i, i] \approx C_q[i, j_1] \approx ... \approx C_q[i, j_p] \approx 1$ and $C_q[t] \approx 0$, $t \notin j_1 ... j_p$. But, we already know that $\tilde{C}_q[i, i] = b_q[i]$. That implies, $b_q[j_1] \approx \tilde{C}_q[i, j_1]$,

$b_q[j_2] \approx \tilde{C}_q[i, j_2]$, ... $b_q[j_p] \approx \tilde{C}_q[i, j_p]$. When we take the summation on all diverse k documents, the Equation 1 holds.

Based on the above proposition, we present an algorithm to learn weights W^T and λ^T iteratively in a supervised learning setup. The training data is provided in a vector r_q (of size $n \times 1$) such that $r_q[i] = 1$ if the document e_i is relevant to the query (and one of diverse documents), otherwise, $r_q[i] = 0$. Note that, the quantity $\tilde{C}_q r_q$ represents the sum of k columns (assuming k number of 1s in r_q) and is the RHS of Equation 1.

Our training objective is to learn λ^T and W^T such that Equation 1 holds. Formally, the problem being solved is:

$$\underset{\lambda_1 \ldots \lambda_{|C|}, w_1 \ldots w_{|N|}}{argmin} D\left(\frac{1}{1+e^{-\sum_g w_g N_g}}, \sum_f \lambda_f \tilde{C}_f r_q\right) \tag{2}$$

where $D(x, y)$ is a distance measure between x and y. (for e.g., KL Divergence, Euclidean, etc.); $\tilde{C}_f$ is the normalized C_f as in Proposition 1.

Applying the coordinate descent technique, we learn the weights W^T and λ^T iteratively using two steps outlined in Equation 3 and Equation 4, each of them convex in the respective optimization variables, hence pur optimisation problem is biconvex.

div-step: Learn $\lambda_1^{(t)}, \lambda_2^{(t)}, \ldots$ holding $w_1^{(t-1)}, w_2^{(t-1)}, \ldots$ constant, by solving:

$$\underset{\lambda_1, \lambda_2, \ldots}{argmin} D\left(\frac{1}{1+e^{-\sum_g w_g^{(t-1)} N_g}}, \sum_f \lambda_f^{(t)} \tilde{C}_f r_q\right) \tag{3}$$

rel-step: Learn $w_1^{(t)}, w_2^{(t)}, \ldots$ holding $\lambda_1^{(t-1)}, \lambda_2^{(t-1)}, \ldots$ constant, by solving:

$$\underset{w_1, w_2, \ldots}{argmin} D\left(\frac{1}{1+e^{-\sum_g w_g^{(t)} N_g}}, \sum_f \lambda_f^{(t-1)} \tilde{C}_f r_q\right) \tag{4}$$

In *div-step*, we learn λ^T by holding W^T fixed and honoring Equation 1. In *rel-step*, we learn W^T by holding λ^T fixed. The relevance and divergence is enforced during training through the vector r_q.

We learn node and edge feature weights iteratively by recognizing and assigning weights to prominent node and edge features that satisfy queries of different types. Having all statistically driven computation of weights for edge features can minimize the side effect of poor node features and likewise computing weights for node features can decrease the consequences of poor edge features.

Algorithm 1 outlines the training procedure. I_q^+, I_q^- are the set of relevant and irrelevant documents for each query q in the ground truth that is used for training.

3.3 Query-time Inference

For a new user query q, inference problem is to choose k diversified results. Using $H(q, \mathcal{C})$ we reduce the search space drastically and get the set $\mathcal{S}$. Otherwise, we need to run our inference on entire set U, which is very expensive. We then compute the node and edge feature matrices for all defined node and edge features. These individual feature matrices are then combined (using λ^T and W^T) to obtain vector b_q and matrix C_q. Based on Proposition 1, our inference objective is to choose k columns from the matrix $\tilde{C}_q$ such that their sum is as close as possible to b_q. Formally, the problem being solved is:

$$\underset{i_1 \ldots i_k}{argmin}\ D\left(b_q, \sum_{j=1}^{k} \tilde{C}_q^{i_j}\right) \tag{5}$$

where $i_1 \ldots i_k$ are indices of k columns of $\tilde{C}_q$.

Determining the exact solution (i.e. $i_1 \ldots i_k$ columns) to the above optimization problem turns out to be computationally infeasible. Hence, we have to resort to an approximate solution. Algorithm 2 describes a greedy inference procedure. At each step we pick one column from $\tilde{C}_q$ that minimizes the distance in Equation 5 most. However, we also ensure that the picked column is most diverse from the already selected columns in the previous steps. At the end of k steps we will have k diverse, but relevant documents.

Algorithm 1 Training

Input: Set of training data instances $\{q, I_q^+, I_q^-, N_f, C_f, r_q\}$
Output: W^T and λ^T
initialize variables W^T and λ^T
learn initial W^T using Logistic Regression ▷ uses $\{q, I_q^+, I_q^- N_f\}$
▷ $\tilde{C}_q, \tilde{C}_f$ used below are normalized C_q, C_f as in Proposition 1
while not converged($| b_q - \tilde{C}_q r_q |$) **do**
  b_q= compute relevance matrix using W^T and I_q^+
  find λ^T so that $D\left(b_q, \sum_f \lambda_f \tilde{C}_f r_q\right)$ is minimized ▷ W^T is fixed
  $p_q = \sum_f \lambda_f \tilde{C}_f r_q$
  find W^T so that $D\left(\frac{1}{1+e^{-\sum_f w_f N_f}}, p_q\right)$ is minimized ▷ λ^T is fixed
end while
return (W^T, λ^T)

Semantic relations and values from Wikipedia page excerpts

1. Synonym: *All redirected names of the Wikipedia page.*

2. Association: *All valid hyperlinks of a Wikipedia page.*

3. Frequent: *All phrases occurring more than two times within a Wikipedia page section.*

4. Synopsis: *All nouns, verbs, adjectives from the abstract and titles of the sections in a Wikipedia page*

5. Hyponym: *All pages/sub categories of selected categories ending with Wikipedia page title. Ex: For Sony: robotics at Sony.*

6. Meronym: *All phrases which occur both in wordnet meronyms and with in Wikipedia pages.*

7. Hypernym: *All parent categories of selected categories.*

8. Homonym: *Pages referring to one or more disambiguation page.*

9. Sibling: *Siblings are the sub categories/pages which do not follow hyponym pattern. Ex: For Sony: list of sony trademarks*

Table 1: Semantic Relations

Algorithm 2 Inference

Input: User query q, Corpus U, λ^T, W^T, N_f,C_f
Output: k diverse interpretations
Generate $\mathcal{S} = H(q, C)$ and build a graph using documents in $S = \{e_1, .., e_n\}$
Compute b_q using W^T and node features $N_{1..|N|}(q, S)$
Compute $C_q = \sum_f \lambda_f C_f(\mathcal{S})$ and normalize as in Proposition 1
$R = \{$ï¿œ$\}$ ▷ set of selected indices
$Q = \{i_1, .., i_n\}$ ▷ indices to select
for $i = 1$ to k **do**

$$\underset{c_k \in Q/R}{argmin} \left\{ D\left(b_q, \underset{r \in R \cup \{c_k\}}{\sum} \left(C_q^r\right)\right) \times \left(1 - \frac{1}{Z} min\left(D\left(C_q^{R_1}, C_q^{c_k}\right), ..., D\left(C_q^{R_{|R|}}, C_q^{c_k}\right)\right)\right) \right\}$$

▷ (query match) × (dissimilar to selected), z is normalizer
  $R = R \cup \{c_k\}$
end for
return k interpretations representing k columns $R_1, ..., R_{|R|}$

4 An example using Wikipedia

In this section we apply our Rel-Div technique to generate diverse but relevant results to a short and/or ambiguous user query using Wikipedia. For *e.g. Beagle, Laptop Charger*, *Sony Camera*, etc. We do not support queries which are highly rich in semantics like *Who invented music, Earn money at home* or very specific in nature like *DB2 error code 1064*.

In this case, U is a set of all Wikipedia entities (a.k.a. pages/articles). Note, in the context of Wikipedia, every document is treated as an entity. We defined $H(q, U)$ as a set of filters which return Wikipedia entities S, called candidate interpretations, relevant to the user query. In order to build this filter function, we made use of prominent Wikipedia attributes (Title, Infobox entries, Frequently occurring words, etc) and different semantic relations between Wikipedia entities (Association via hyperlinks, Page Redirects, See Also links, etc). Table 1 summarizes these Wikipedia signals, which are captured for every entity.

4.1 Node Features

Query Match: Calculates the term overlap between query terms and the semantic relation terms of an interpretation. For e.g., for the query *Sony*, *PlayStation 2* is one of the interpretations, which has multiple occurrence of term Sony in one or more semantic relations.

No. of Semantic Relation match: Total number of semantic relations that contain the query terms. For e.g., for the query Sony, PlayStation 2 interpretation may have 3 semantic relations (Synonym, Association and Frequent) containing term Sony.

Title score: Captures the interpretation title match to the query terms.

4.2 Edge Features

Interpretation Content Overlap: This feature measures the similarity between two interpretations by considering the amount of overlap between the words in these interpretation title and content.

Decaying Recursive Similarity: We considered neighborhood of an interpretation (hyperlinked entities, parent categories, subcategories, and grand parent pages) in the similarity measurement. However, an appropriate weight which decays with distance is set to avoid influence of farther neighborhood nodes.

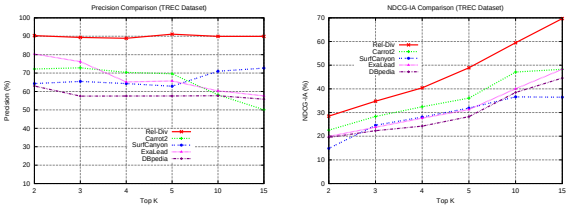
Link based proximity: Determined by the depth $D(lca)$ of the least common ancestor (LCA) of interpretations I_i and I_j from the root of Wikipedia category structure and the hop distance $len(I_i, I_j)$ from I_i to I_j through LCA. Link proximity is defined as $LP(I_i, I_j) \propto D(lca) * len(I_i, I_j)$. When multiple LCAs exist, we define the proximity as $\max(LP(I_i, I_j))$.

5 Experimental Evaluation

We used Wikipedia as our knowledge source. We captured different signals shown in Table 1 for every Wikipedia entity.

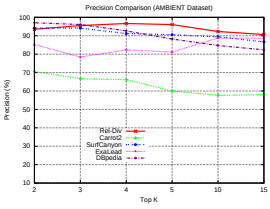
5.1 Dataset

The QRU dataset used in SIGIR 2011 contains 100 TREC queries with various interpretations. We restricted our space of interpretations to Wikipedia entities. We also experimented with ambiguous queries from the AMBIENT dataset which contains 40 one word queries.

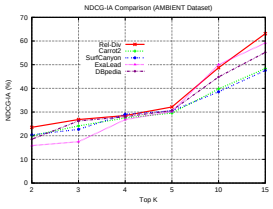


(a) Precision : TREC

(b) NDCG-IA : TREC



(c) precision : AMBIENT



(d) NDCG-IA : AMBIENT

Figure 2: Comparison with external systems

		Precision(%)			Recall(%)			NDCG-IA(%)		
		@5	@10	@10	@5	@10	@10	@5	@10	@10
TREC	Rel-Div	**91.13**	**89.93**	**89.83**	**7.02**	**13.85**	**20.4**	48.9	59.47	**69.7**
	M-Div	89.87	84.27	84.32	6.74	12.71	18.88	**49.71**	**62.68**	67.39
	M-Div-NI	83.75	80	80	6.83	12.81	19.35	42.48	60.88	66.52
	AFP	78.3	76.9	80.7	6.3	12.4	18.1	34.2	38.8	47.6
AMBIENT	Rel-Div	96.05	92.3	90.67	7.33	**14.57**	**21.61**	**32.12**	**48.72**	**63.1**
	M-Div	96.15	**94.15**	**93.56**	**7.43**	14.37	21.61	32.41	47.49	58.09
	M-Div-NI	**96.2**	93.58	93.19	7.33	13.87	21.11	22.93	43.59	55.84
	AFP	88.4	90.9	92.3	6.9	13.6	21.47	32.09	45.9	55.1

Table 2: Results of different approaches

5.2 Evaluation methodology

Manually, interpretations for each query are marked as relevant or irrelevant and each interpretation is assigned one or more topics. The system is trained on 30and tested on the rest. We evaluated results on queries of length one or two. The relevance of any interpretation to the query is measured using precision at different positions and the diversity is estimated using NDCG-IA (Agrawal et al., 2009). Recall measurement is tricky. It is practically not possible to manually inspect all Wikipedia entities and determine how many are actually relevant for a query. Hence we based our recall on the candidate interpretations generated. We manually counted number of relevant interpretations present in the candidate interpretations and measured how many of these relevant interpretations appeared in the top k interpretations.

In our experiments, we also consider a couple of other approaches to diversification, which have been reported in literature, though used in other problem settings. These include variants of GCD and affinity propagation (Frey and Dueck, 2006; Frey and Dueck, 2007).

M-Div *:* Uses page rank matrix M as in GCD instead of the C_q matrix.

M-Div-NI *:* Similar to M-Div, but node and edge weights are learnt independently, without any iterations. This acts as GCD implementation.

AFP*:Exemplar* nodes of Affinity propagation are taken as interpretations.

5.3 Comparison with other approaches

While experimenting with our proposed approach, we found best performance when D in div-step was chosen to be KL-divergence and D in rel-step was chosen as the Euclidean distance. In Table 2, we compare the proposed diversification algorithm against M-Div, M-Div-NI and AFP on precision, recall and NDCG-IA measures.

We observed that our Ranking algorithm Rel-Div performs at par with (and sometimes even better than) M-Div and M-Div-NI. However, one of the major advantage of our method compared to M-Div and M-Div-NI is that, we need not calculate the inverse of C_q matrix, which is a computationally intensive process for a large dimension matrices. We conclude from the results that the Rel-Div performs consistently better than other approaches when both relevance and diversification are considered across all types of queries.

5.4 Comparison against other systems

We compare the diversity in search result using our approach against those from four other systems, *viz.*, carrot2, SurfCanyon, Exalead and DBPedia to demonstrate that the Rel-Div approach produces high diversity in the search results, which is evident from the Figure 2.

6 Conclusion

We presented a body of techniques for generating top k interpretations to a user query using some internet encyclopedia, (in particular, Wikipedia was used in the experiments that were reported). Our approach is hinged on catering to two needs of the user, *viz.*, that all the interpretations are relevant and that they are as diverse as possible. We addressed this using a bunch of node features and edge features based on semantic relations and learn these feature weights together iteratively. We present experimental evaluations and find that our approach performs well on both the fronts (diversity and relevance) in comparison to existing techniques and publicly accessible systems. We believe technique can be improved for better handling of multiword queries by adopting deep NLP parsing techniques, which will form part of our future work.

References

Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. 2009. Diversifying search results. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, WSDM '09, pages 5–14, New York, NY, USA. ACM.

Ori Ben-Yitzhak, Nadav Golbandi, Nadav Har'El, Ronny Lempel, Andreas Neumann, Shila Ofek-Koifman, Dafna Sheinwald, Eugene Shekita, Benjamin Sznajder, and Sivan Yogev. 2008. Beyond basic faceted search. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, pages 33–44, New York, NY, USA. ACM.

Christina Brandt, Thorsten Joachims, Yisong Yue, and Jacob Bank. 2011. Dynamic ranked retrieval. In *Proceedings of the fourth ACM international conference on Web search and data mining*, WSDM '11, pages 247–256, New York, NY, USA. ACM.

Surf Canyon. http://www.surfcanyon.com/.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 335–336, New York, NY, USA. ACM.

Carrot2. http://search.carrot2.org/stable/search.

DBPedia. http://dbpedia.org/facetedsearch.

Avinava Dubey, Soumen Chakrabarti, and Chiranjib Bhattacharyya. 2011. Diversity in ranking via resistive graph centers. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 78–86, New York, NY, USA. ACM.

Exalead. http://www.exalead.com/search/.

Freebase. http://www.freebase.com/.

Brendan Frey and Delbert Dueck. 2006. Mixture modeling by affinity propagation. In *Advances in Neural Information Processing Systems 18*, pages 379–386. MIT Press, Cambridge, MA.

Brendan J. Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. *Science*, 315:2007.

Jochen Gorski, Frank Pfeuffer, and Kathrin Klamroth. 2007. Biconvex sets and optimization with biconvex functions: a survey and extensions. *Math. Meth. of OR*, 66(3):373–407.

Rasmus Hahn, Christian Bizer, Christopher Sahnwaldt, Christian Herta, Scott Robinson, Michaela BÃŒrgle, Holger DÃŒwiger, and Ulrich Scheel. 2010. Faceted wikipedia search. In Witold Abramowicz and Robert Tolksdorf, editors, *Business Information Systems*, volume 47 of *Lecture Notes in Business Information Processing*, pages 1–11. Springer Berlin Heidelberg.

Marti A. Hearst. 2006. Clustering versus faceted categories for information exploration. *Commun. ACM*, 49(4):59–61, April.

Hao Ma, Michael R. Lyu, and Irwin King. 2010. Diversifying query suggestion results. In *AAAI*.

Karthik Raman, Thorsten Joachims, and Pannaga Shivaswamy. 2011. Structured learning of two-level dynamic rankings. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 291–296, New York, NY, USA. ACM.

Ashwin Swaminathan, Cherian V. Mathew, and Darko Kirovski. 2009. Essential pages. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '09, pages 173–182, Washington, DC, USA. IEEE Computer Society.

YAGO. http://www.mpi-inf.mpg.de/yago-naga/.

Lian Yan, Robert H. Dodier, Michael Mozer, and Richard H. Wolniewicz. 2003. Optimizing classifier performance via an approximation to the wilcoxon-mann-whitney statistic. In *ICML*, pages 848–855.

Yisong Yue and Thorsten Joachims. 2008. Predicting diverse subsets using structural svms. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 1224–1231, New York, NY, USA. ACM.