Rel-Div: Generating Diversified Query Interpretations from Semantic Relations

Ramakrishna Bairi[1], A. Ambha[2], and Ganesh Ramakrishnan[2]

[1] IITB-Monash Research Academy, IIT Bombay
[2] IIT Bombay

Abstract. Accelerated growth of the World Wide Web has resulted in an increase in appetite for searching over Internet to fulfill the information needs. Understanding user intent plays a pivotal role in determining the quality of search results and improving user satisfaction. But short, ambiguous or underspecified queries make the process of determining the concealed user intention harder. Identifying diversified but relevant interpretations for a query with an impressive accuracy, is still an active area of research. These varied interpretations originate from entities associated with the user query and relations between the entities. We address the problem of generating diverse but relevant interpretations by utilizing an Internet encyclopedia (Wikipedia) as a primary source entities and their relations. By preprocessing the encyclopedia, we build a rich repository of *Semantic Relations*, which characterize these entities and their relationships. We present algorithms to enumerate pertinent interpretations for a query based on the repository. The proposed approach uses the repository to generate candidate interpretations over which we apply graph based iterative approaches to generate diversified and relevant results. We empirically evaluate the effectiveness of our approach, with the 'Query Relevance and Understanding' dataset of TREC 2011 workshop and AMBIENT (Ambiguous Entities) dataset.

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 purpose 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 under-specified. In addition, a query can have multiple *concealed interpretations*. For example, *Sun* could be interpreted as "The sun as a star", "Sun Micro systems company", "Sun news paper" 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. Out of many possible interpretations to a short query, users expect their intended answer to be present in the top few search results. This calls for the need of presenting a diversified but relevant set of results in the top K positions.

Most prior research has focused on generating diversified result [14, 16, 3, 12, 5, 17, 9, 8, 10, 2, 13, 11]. Inspired by the work GCD [5] and MMR [4], we develop a new technique for diversity ranking of interpretations based on an *interpretation graph*. As part of this technique, we propose an algorithm to learn the node and edge weights of the interpretation graph iteratively in a biconvex optimization setting.

2 Diversified Interpretation Generation

2.1 Our Problem

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

2.2 The Training Algorithm

We expect groups of entities in $\mathcal{S}$ to be related to each other via some semantic relations. We initially construct an entity-relation graph using $e_1...e_n$. We refer to this graph as an *Interpretation Graph*, since the entities in this graph are obtained as various interpretations of the query. While the nodes are entities from $\mathcal{S}$, each edge is a relation between the entities. 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 entity e_i and the cell values in that column indicate the similarity of entity e_i with other entities. 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.

Learning 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 $N_{1..F_N}(q,\mathcal{S})$ be a set of F_N node features. Each feature $N_f(q,\mathcal{S})$ evaluates the relevance of entities in $\mathcal{S}$ 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 entities. The node potential vector b_q is obtained by combining the scores returned by individual feature functions $N_f(q,\mathcal{S})$. One of the obvious choices is to use Logistic Regression [15].*i.e.* $b_q[i] = \frac{1}{1+e^{-\sum_{f=1}^{F_N} w_f N_f(q,\mathcal{S})[i]}}$. The weight vector $W^T = [w_1...w_{F_N}]$ is learnt through supervised training explained in Section 2.2.

Learning 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 $C_{1..F_E}(S)$ be the set of F_E edge features that evaluate similarities between entities in $\mathcal{S}$ 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 entities. The edge potential matrix C_q is obtained as $C_q = \sum_{f=1}^{F_E} \lambda_f C_f(\mathcal{S})$ where $0 \leq \lambda_f \leq 1$ and $\sum \lambda_f \geq 1\, \forall f$. The weight vector $\lambda^T = [\lambda_1...\lambda_{F_E}]$ is learnt through supervised training explained in Section 2.2.

Training Feature Weights W^T, λ^T. Our technique is based on following proposition Proposition 1:

$$b_q \approx \sum_{k=1}^{K} \tilde{C}_q^{i_k} \tag{1}$$

for sufficiently large K diverse entities, 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, $\tilde{C}_q^{i_k}$ is the i_kth column of matrix $\tilde{C}_q$.

The intuition behind this approximated equality comes from the fact that, two similar entities should have similar relevance score with the query and we are interested in selecting K diverse entities. Let e_i be one of these K diverse entities. If the entities $e_{j_1}...e_{j_p}$ 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 <u>all diverse</u> K entities, the Equation 1 holds.

Based on the above observation, 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 entity e_i is relevant to the query (and one of diverse entities), 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:

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

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.

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, resulting in binconvex optimization.

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

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

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

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

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. r_q is provided by the user as part of training data.

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 entities for each query q in the ground truth used for training.

Inference. For a new user query q, inference problem is to choose K diversified results. Using $\mathcal{H}(q, \mathcal{C})$ we reduce the search space drastically and get the set $\mathcal{S}$. Otherwise, we need to run our inference on entire set $\mathcal{C}$, 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:

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

where $i_1 ... i_K$ are indices of K columns of matrix $\tilde{C}_q$.

Finding out exact solution (i.e. $i_1 ... i_K$ columns) to the above optimization problem turns out to be computationally infeasible. 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.

Algorithm 1. Training

```
1: Input: Set of training data instances $\{q, I_q^+, I_q^-, N_f, C_f, r_q\}$
2: Output: $W^T$ and $\lambda^T$
3: initialize variables $W^T$ and $\lambda^T$
4: 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
5: while not converged(| $b_q - \tilde{C}_q r_q$ |) do
6:     $b_q$= compute relevance matrix using $W^T$ and $I_q^+$
7:     find $\lambda^T$ so that $D\left(b_q, \sum_f \lambda_f \tilde{C}_f r_q\right)$ is minimized
                                                  ▷ $W^T$ is fixed
8:     $p_q = \sum_f \lambda_f \tilde{C}_f r_q$
9:     find $W^T$ so that $D\left(\frac{1}{1+e^{-\sum_f w_f N_f}}, p_q\right)$ is minimized
                                                  ▷ $\lambda^T$ is fixed
10: end while
    return $\left(W^T, \lambda^T\right)$
```

Algorithm 2. Inference

```
1: Input: $q, \mathcal{C}, \lambda^T, W^T, N_f, C_f$
2: Output: $K$ diverse interpretations
3: Generate $\mathcal{S} = \mathcal{H}(q, C)$ and build a graph using entities in $\mathcal{S} = \{e_1, .., e_n\}$
4: Compute $b_q$ using $W^T$ and node features $N_{1..F_E}(q, \mathcal{S})$
5: Compute $C_q = \sum_f \lambda_f C_f(\mathcal{S})$ and normalize as in Proposition 1
6: $R = \{$ï¿œ$\}$                      ▷ set of selected indices
7: $Q = \{i_1, .., i_n\}$               ▷ indices to select
8: for $i = 1$ to $K$ do
9:     $\underset{c_k \in Q/R}{argmin} \left\{ D\left(b_q, \sum_{r \in R \cup \{c_k\}} \left(C_q^r\right)\right) \times \left(1 - \frac{1}{Z} min\left(D\left(C_q^{R_1}, C_q^{c_k}\right), ..., D\left(C_q^{R_{|R|}}, C_q^{c_k}\right)\right)\right) \right\}$
       ▷ (query match) × (dissimilar to selected), $z$ is normalizer
10:    $R = R \cup \{c_k\}$
11: end for
    return $K$ interpretations representing $K$ columns $R_1, ..., R_{|R|}$
```

3 Experimental Evaluation

We apply our Rel-Div technique to generate interpretations to a short and/or ambiguous user query (*e.g. Beagle, Sony Camera*, etc) using Wikipedia. We do not support queries which are highly rich in semantics like *Who invented music* or very specific in nature like *DB2 error code 1064*. We used various signals from Wikipeida (such as title, hyperlink structure, infoboxes, category structure, etc) to define a bunch of node features and edge features. We evaluated our technique using the QRU dataset of SIGIR 2011 contains 100 TREC queries and the AMBIENT dataset which contains 40 one word queries.

3.1 Evaluation Methodology

The relevance of any interpretation to the query is measured using precision at different positions and the diversity is estimated using NDCG-IA [1]. For Recall, since it is practically not possible to manually inspect all Wikipedia entities for relevance to a query, we based our recall on the candidate interpretations generated.

In our experiments, we 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 [5] and Affinity Propagation [6, 7].

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 (as in GCD).

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

3.2 Comparisons of Approaches

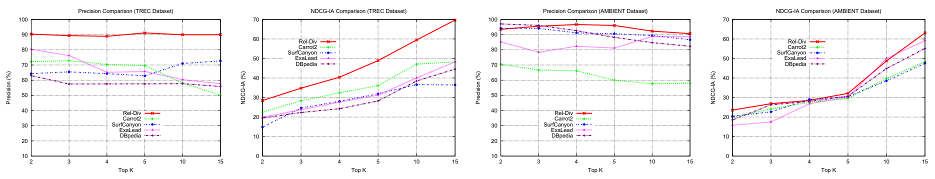
While experimenting with our proposed approach, we found best performance when D in div-step was chosen to be KL-divergence and D in rel-step was chosen as the Euclidean distance. In Table 1, we compare the proposed diversification algorithm against M-Div, M-Div-NI and AFP on precision, recall and NDCG-IA 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 also compare the diversity in interpretations using our approach against those from four other search engines: Carrot2, SurfCanyon, Exalead and DBPedia. Figure 1 confirms that our approach produces better diversity in the interpretations.

Table 1. Results of different approaches

		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



(a) Precision : TREC (b) NDCG-IA : TREC (c) precision : AMB (d) NDCG-IA : AMB

Fig. 1. Comparison with external systems

Examples of interetations generated from our system for many short queries can be found at `http://qassist.cse.iitb.ac.in/facetedsearcher/examples.pdf`

4 Conclusion

We presented techniques for generating the top K interpretations to a user query using some internet encyclopedia, (in particular, Wikipedia). Our approach caters 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. Our experimental evaluations, comparison to existing techniques and publicly accessible systems show that our approach performs well on both the fronts. 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

1. Agrawal, R., Gollapudi, S., Halverson, A., Ieong, S.: Diversifying search results. In: WSDM 2009, pp. 5–14. ACM, New York (2009)
2. Ben-Yitzhak, O., Golbandi, N., Har'El, N., Lempel, R., Neumann, A., Ofek-Koifman, S., Sheinwald, D., Shekita, E.J., Sznajder, B., Yogev, S.: Beyond basic faceted search. In: WSDM, pp. 33–44 (2008)
3. Brandt, C., Joachims, T., Yue, Y., Bank, J.: Dynamic ranked retrieval. In: WSDM, pp. 247–256 (2011)
4. Carbonell, J., Goldstein, J.: The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: Research and Development in Information Retrieval, pp. 335–336 (1998)
5. Dubey, A., Chakrabarti, S., Bhattacharyya, C.: Diversity in ranking via resistive graph centers. In: KDD 2011, pp. 78–86. ACM, New York (2011)
6. Frey, B., Dueck, D.: Mixture modeling by affinity propagation. In: Advances in Neural Information Processing Systems 18, pp. 379–386. MIT Press, Cambridge (2006)
7. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science 315 (2007)
8. Hahn, R., Bizer, C., Sahnwaldt, C., Herta, C., Robinson, S., Bürgle, M., Düwiger, H., Scheel, U.: Faceted wikipedia search. In: Abramowicz, W., Tolksdorf, R. (eds.) BIS 2010. LNBIP, vol. 47, pp. 1–11. Springer, Heidelberg (2010)
9. Hearst, M.A.: Clustering versus faceted categories for information exploration. Commun. ACM 49(4), 59–61 (2006)
10. Li, C., Yan, N., Roy, S.B., Lisham, L., Das, G.: Facetedpedia: dynamic generation of query-dependent faceted interfaces for wikipedia. In: WWW, pp. 651–660 (2010)
11. Ma, H., Lyu, M.R., King, I.: Diversifying query suggestion results. In: AAAI (2010)
12. Raman, K., Joachims, T., Shivaswamy, P.: Structured learning of two-level dynamic rankings. In: CIKM (2011)
13. Shen, Y., Yan, J., Yan, S., Ji, L., Liu, N., Chen, Z.: Sparse hidden-dynamics conditional random fields for user intent understanding. In: WWW 2011, pp. 7–16. ACM, New York (2011)
14. Swaminathan, A., Mathew, C.V., Kirovski, D.: Essential pages. Web Intelligence, 173–182 (2009)
15. Yan, L., Dodier, R.H., Mozer, M., Wolniewicz, R.H.: Optimizing classifier performance via an approximation to the wilcoxon-mann-whitney statistic. In: ICML, pp. 848–855 (2003)
16. Yue, Y., Joachims, T.: Predicting diverse subsets using structural SVMs. In: ICML, pp. 271–278 (2008)
17. Zhai, C., Cohen, W.W., Lafferty, J.: Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. In: SIGIR, pp. 10–17 (2003)