Pairwise Document Classification for Relevance Feedback

Jonathan L. Elsas, Pinar Donmez, Jamie Callan, Jaime G. Carbonell
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
{jelsas,pinard,callan,jgc}@cs.cmu.edu

ABSTRACT

In this paper we present Carnegie Mellon University's submission to the TREC 2009 Relevance Feedback Track. In this submission we take a classification approach on document pairs to using relevance feedback information. We explore using textual and non-textual document-pair features to classify unjudged documents as relevant or non-relevant, and use this prediction to re-rank a baseline document retrieval. These features include co-citation measures, URL similarities, as well as features often used in machine learning systems for document ranking such as the difference in scores assigned by the baseline retrieval system.

1. INTRODUCTION

Retrieval systems employing relevance feedback techniques typically focus on augmenting the representation of the information need in order to improve performance. This is typically done through adding or re-weighting terms in the query representation, and have been shown to be effective techniques in the past [4, 7, 8, 13]. These techniques, however, are typically limited to the information need representation used in the baseline retrieval system and generally don't utilize information beyond the word distributions in the feedback documents to modify the query model.

This paper describes the CMU submission to the TREC 2009 Relevance Feedback Track. With this submission, our goal is to explore techniques beyond query term re-weighting and other traditional approaches to query expansion. Our approach constructs pairwise features between judged-relevant feedback documents and unjudged documents, and then applies a learned classifier to identify those unjudged documents likely to be relevant. The output of this classification is then used to re-rank an initial document ranking, favoring those documents predicted to be relevant to the query.

2. SYSTEM DESCRIPTION

The CMU submission system consists of four main components: baseline retrieval, document selection, relevance classification and document re-ranking. The document selection and relevance classification components of the system take a machine learning approach, using a feature space derived from document pairs.

This section describes these four components in the CMU relevance feedback track submission, as well as this feature-based document-pair representation.

2.1 Baseline Retrieval

For these experiments, we use Indri for our baseline ranking[1]. Indri has been shown to perform well in ad-hoc retrieval tasks at TREC in previous years [8, 10]. For these experiments we made use of a small standard stop-word list and applied the Krovetz stemmer. We constructed full-dependence model queries from the query text [9]. Smoothing parameters were taken directly from previously published TREC configurations[2].

Initial informal experiments with pseudo-relevance feedback (PRF) with relevance models [7] indicated that traditional approaches to query expansion may be less effective on the ClueWeb09 collection due to the susceptibility of those techniques to the web-spam present in the collection. For this reason we did not use PRF in our baseline run.

2.2 Document Representation

We take a machine learning approach to the document selection and relevance classification components of our system. These components use a common document representation scheme, described below.

2.2.1 *Pairwise Representation*

Our feature-based representation constructs feature vectors for each *pair* of documents retrieved by the baseline retrieval for a given query.

$$D_q = \{d_{q1}, d_{q2}, \ldots, d_{qR}\}$$
$$P_q = \{\mathbf{f}(d_{qi}, d_{qj}) \,|\, i, j \in \{1, \ldots, R\}, i \neq j\}$$

D_q are the R documents retrieved for query q, P_q are the document pair vectors defined by $\mathbf{f} : D_q \times D_q \rightarrow \mathbb{R}^M$, a vector feature function over document pairs:

$$\mathbf{f}(d_i, d_j) = \langle f_0(d_i, d_j), f_1(d_i, d_j), \ldots, f_M(d_i, d_j) \rangle$$

where each f_k are instantiations of individual features derived from the document pairs.

This representation allows use of some features that can be difficult to integrate into traditional retrieval systems that exclusively use term-weighting for estimating relevance. As we describe below, many of our features cannot be modeled with a bag-of-words document representation. Using a pairwise representation also allows a "query by example" approach to leveraging the feedback information. We make the assumption that relevant documents tend to be similar to each other, viz. the *cluster hypothesis* [12]. Thus, using pairwise features that describe document similarities

[1] `http://www.lemurproject.org/indri`

[2] `http://ciir.cs.umass.edu/ metzler/indri-tb05.tgz`

(or dissimilarities), the goal of our approach is to find other relevant documents similar to those that have been judged.

2.2.2 *Features*

The fourteen document-pair feature functions ($f_k(d_i, d_j)$) used in these experiments are described below. These features are generally intended to capture different types of similarity (or dissimilarity) between two documents. Many of these features are computed with the Jaccard coefficient, a measure of similarity of two sets of objects. The Jaccard coefficient of two sets A and B is given by:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

1. Document features
   - (a) **Length**: The absolute value of the difference in the lengths of d_i and d_j.
2. URL features
   - (a) **URL Depth**: The absolute value of the difference in the depth (number of occurrences of '/') in the URLs of d_i and d_j.
   - (b) **URL Host**: The Jaccard coefficient computed over overlapping character 4-grams in the URL hostnames of d_i and d_j.
   - (c) **URL Path**: The Jaccard coefficient computed over overlapping character 4-grams in the URL paths of d_i and d_j.
3. Webgraph features[3]
   - (a) **In-link**: The absolute value of the difference in the number of in-links to d_i and d_j.
   - (b) **Out-link**: The absolute value of the difference in the number of out-links from d_i and d_j.
   - (c) **Co-citation**: The Jaccard coefficient computed over the set of documents that link to d_i and d_j.
   - (d) **References**: The Jaccard coefficient computed over the set of documents that d_i and d_j link to.
4. Query-derived features
   - (a) **Unigram count**: The absolute value of the difference in the count of query tokens in d_i and d_j.
   - (b) **Ordered bigram count**: The absolute value of the difference in the count of ordered query bigrams in d_i and d_j.
   - (c) **Unordered bigram count**: The absolute value of the difference in the count of unordered query bigrams in d_i and d_j.
   - (d) **Unigram score**: The absolute value of the difference in Indri score of the unigram component of the baseline dependence model query.
   - (e) **Ordered window score**: The absolute value of the difference in Indri score of the ordered window component of the baseline dependence model query.
   - (f) **Unordered window score**: The absolute value of the difference in Indri score of the unordered window component of the baseline dependence model query.

All features are normalized to have zero-mean unit-variance per query prior to training and testing.

2.3 Relevance Classification

We can use the above document pair representation scheme to train a classifier that predicts whether unjudged documents are relevant or non-relevant given some judged documents. We make the assumption that relevant documents are likely to be similar to each other, and dissimilar to non-relevant documents with respect to the features defined in Section 2.2.2. In contrast, we make no assumption about the similarity of non-relevant documents to each other.

We train this classifier on a set of queries with known relevant and non-relevant documents. Let the set of (binary) judgements for a given training query, q be:

$$J_q = \{(d_{qi}, r_{qi}) \,|\, r_{qi} \in \{0, 1\}\}$$

where $r_{qi} = 1$ indicates the document d_{qi} is relevant for query q, and $r_{qi} = 0$ indicates the document is non-relevant.

We train a logistic regression classifier on judged document pairs, letting $y_{qij} \in \{0, 1\}$ indicate the class label of the pair (d_{qi}, d_{qj}). This training set is constructed as follows:

$$JP_q = \{(\mathbf{f}(d_{qi}, d_{qj}), y_{qij}) \,|\, r_{qi} = 1; y_{qij} = r_{qj}\}$$

so that each pair of training examples has at least one judged relevant document (d_{qi}). The judgement on the other document (d_{qj}) indicates whether this pair is a positive or negative training example. Thus, the classifier is trained to assign a positive (1) classification to relevant/relevant document pairs, and a negative (0) classification to relevant/non-relevant pairs. The result of this training produces a classification function $h : D_q \times D_q \rightarrow [0, 1]$, where a value close to 1 indicates a positive classification, and a value close to 0 indicates a negative classification.

After feedback judgements are collected, assuming some of the feedback documents are relevant, we can apply the learned classifier to predict whether or not unjudged documents are relevant or non-relevant. For each unjudged document d_{qj}, we make a relevance prediction given all the judged relevant documents: $\{h(d_{qi}, d_{qj}) \,\forall\, d_{qi} \text{ s.t. } r_{qi} = 1\}$. This set of predictions can be combined in several ways to form a final relevance classification, for example taking the *mean*, *minimum*, or *maximum* value across the predictions. Preliminary experiments with the TREC 2009 Relevance Feedback Track data showed that taking the *maximum* prediction value across all the judged relevant documents generally yielded the best performance. Thus, we define our final prediction for an unjudged document as follows:

$$\pi(d_{qj}) = \max_{d_{qi} \in J_q; r_{qi} = 1} h(d_{qi}, d_{qj})$$

This relevance prediction effectively classifies unjudged documents based on their similarity to the *closest* judged relevant feedback document with respect to the feature space defined above. Because of this, it is critical to collect relevance judgements on a *diverse* set of documents in order to maximize the chance of identifying relevant documents similar to possibly relevant but unjudged documents.

[3] All webgraph features were computed with the use of the WebGraph software package, available from `http://webgraph.dsi.unimi.it/` [3].

Note that judged non-relevant documents are used for training the model, but are not used at prediction time after collecting feedback judgements. Methods of using these non-relevant feedback documents is an area for future refinement of the models presented here.

2.4 Document Re-Ranking

We use the output of the above relevance classifier π to re-rank the documents retrieved with the baseline ranking algorithm. Due to the difficulty of re-scaling Indri's language modeling score and the output of a logistic regression classifier, we chose to combine scores using a rank-based voting method, Borda Count [1]. Rather than combining the *scores* of the baseline ranker and the logistic regression, Borda Count linearly combines the *ranks* of the documents from each of these components. Although this method ignores the magnitude of the confidence of the prediction output, it avoids the need to re-scale the scores to be comparable.

We use a weighted version of Borda Count in these experiments to adjust the relative influence of the baseline ranking score and the relevance prediction output. This weight is selected to maximize Mean Average Precision via a grid search on the same training data used to train the relevance classifier. For these experiments, we selected a weight of 0.3 on the relevance classifier and 0.7 on the baseline ranking.

2.5 Document Selection

The final component of our system is the document selection system. As pointed out earlier, diversity is a critical factor underlying our document selection approach. The classification method in Section 2.3 gives a probabilistic measure of the relevance of an unjudged document paired with a judged relevant document. The final relevance score of an unjudged document is then the maximum value assigned across all the judged relevant documents for that query. Having similar judged relevant documents agree on the relevance of an unjudged document is not as effective as having agreement across a diverse committee. Thus, this is the main focus of our selection mechanism.

The most naïve approach is to select the top 5 documents for feedback. However, it is often the case that top documents are similar to each other. Learning the relevance level of similar documents might improve the ranking for additional similar documents, but it might not generalize to a larger set of documents. The diversity factor has been investigated in the active learning literature [5, 11]. It is indicated that choosing the unlabeled examples which are representative of the underlying data distribution boosts the performance. Hence, we focus in this section to select documents that are likely to be relevant and also different from each other. Specifically, we adopted a clustering framework where we cluster the unjudged documents using the Fuzzy Clustering algorithm [2, 6].

The objective of fuzzy clustering is to spread out each example into various clusters. In other words, each example has a degree of belonging to clusters, rather than completely belonging a single cluster. Hence, it is a soft clustering method instead of hard clustering. For each point x, there is a corresponding coefficient indicating the degree of belonging to the k^{th} cluster; i.e. $u_k(x)$. However, the sum of the coefficients for any given point x is equal to 1.

$$\sum_{k=1}^{K} u_k(x) = 1 \forall x \tag{2}$$

Furthermore, the degree of belonging $u_k(x)$ (or the membership coefficient) is inversely related to the distance of the point to the cluster center $center_k$:

$$u_k(x) = \frac{1}{d(center_k, x)} \tag{3}$$

Hence, points further away from the center of the cluster have a lower degree of belonging than the points closer to the center. The cluster center is calculated using the mean of all points, weighted by their membership coefficients:

$$center_k = \frac{\sum_x u_k(x)^f x}{\sum_x u_k(x)^f} \tag{4}$$

where $f > 1$ is a predefined parameter that controls the fuzzyness. For instance, increasing f leads to crisper clusterings whereas f close to 1 resembles the k-means algorithm. Finally, the fuzzy clustering tries to minimize the following objective function

$$\sum_{k=1,\ldots,K} \frac{\sum_{i,j} u_k(i)^f u_k(j)^f d(i,j)}{2\sum_j u_k(j)^f} \tag{5}$$

where $d(i, j)$ is the distance between two documents d_i and d_j. The algorithm tries to minimize the inter-cluster similarity while minimizing the intra-cluster variance. It converges to a locally optimal solution [2].

We use the output of our trained logistic regression classifier on the document-pair features, as described above, to approximate this distance metric, $d(i, j)$. Although this is not a proper *metric* in the mathematical sense, it can be used by the presented clustering algorithm and it does capture the feature-weighted similarity used in the relevance classification component of our system.

Because our re-ranking system does not use non-relevant feedback documents, we want to select documents that are likely to be relevant as well as diverse. The classification scheme described in Section 2.3 requires judged relevant documents to make predictions on the unjudged documents during testing. Initial investigation with the TREC 2008 Relevance Feedback data indicated that increasing the number of judged relevant documents is quite beneficial to the final re-ranking performance. Therefore, our aim is to identify the potentially relevant documents while maintaining a degree of diversity among them. Assuming the baseline indri ranking is well-tuned and relatively accurate, it is reasonable to consider the top documents to be judged. After we build the clusters among unjudged documents, we choose the top ranked document in each cluster to be judged. This simple method has the two characteristics we require: 1) it consists of top ranked documents that are likely to be relevant, and 2) it is a diverse set that leverages the underlying relevance distribution.

3. EXPERIMENTS

This section describes the experiments conducted for the TREC 2009 relevance feedback track.

3.1 Training

The document selection and relevance classification components require training data in order to learn weights on the features described in Section 2.2.2 for use in the logistic regression relevance classifier (Section 2.3) and the clustering algorithm (Section 2.5). Because previous queries and relevance judgements do not exist on the ClueWeb09 dataset, we built our training data from previous years' TREC ad-hoc tasks using the GOV2 collection. This training set includes all relevance judgements for queries 701-850 excluding those queries with no relevant documents. The final constructed training set includes 1.8 million document pairs, with 31% positive examples (relevant/relevant pairs) and 69% negative examples (relevant/non-relevant pairs). Although these two document collections are somewhat different, the feature set described above can be generated on both collections. We make the assumption for these experiments that the feature weights learned on the GOV2 collection are similarly effective on the ClueWeb09 collection.

3.1.1 Features Weights

Sections 2.2 and 2.3 describe the pairwise document representation and how we use this representation in a logistic regression classifier to predict the relevance level of an unjudged document given a judged relevant document. It is informative to inspect the learned logistic regression weights for each of the features used in our model, as the larger magnitude weights indicates a more influential feature. Figure 1 shows the absolute weights of all the features learned in the logistic regression model. We can see that the most influential features in our model are the URL-based features, particularly the similarity of the host name and path portions of the URL. The next most powerful features are the components of the baseline Dependence Model query — the ordered and unordered window scores assigned by Indri. The **Out-link count** feature is the only webgraph feature that is at all influential in the model. This feature is derived exclusively from the content of the page (just the count of anchors), rather than relations between documents in the collection. This may be an indication that the GOV2 webgraph used for training may be too sparse to effectively estimate the other webgraph features which rely on linking among documents in the collection.

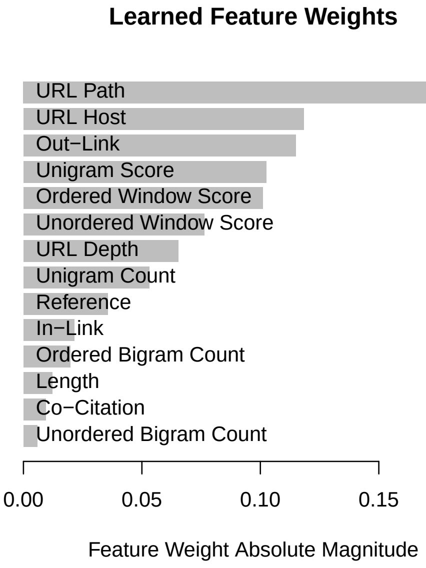

Figure 1: Learned Feature Weights.

3.1.2 Document selection

In this section, we analyze the quality of our document selection mechanism across queries. First, looking at the distribution of ranks in our baseline retrieval selected for judgement, we can see a strong skew towards the top-ranked documents to be selected for judgement. We also see that we do a reasonably good job of finding relevant documents not only at high ranks but also at lower ranks, though with decreasing frequency. This is especially useful since it detects the relevant documents the baseline ranker misjudged by putting in lower ranks. Incorporating such documents to the rank learner is likely to lead to improvements.

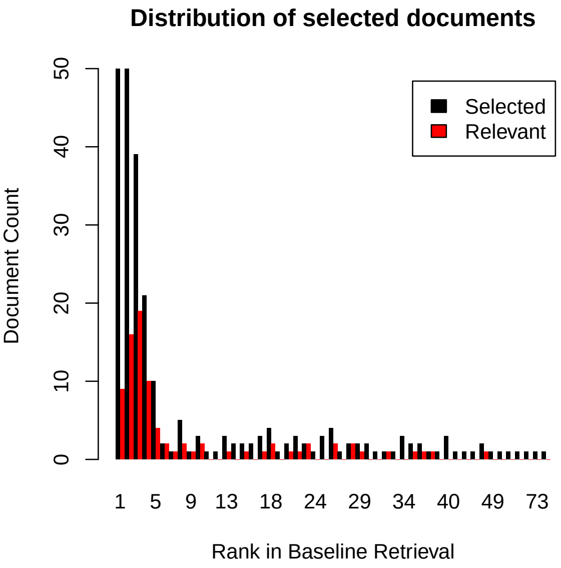

Figure 2: Rank distribution of selected documents, and judged relevant documents.

To evaluate the quality of our phase-1 document selection (CMU.1), we primarily consider the fraction of *other* inputs that our phase-1 input performed better than, which we refer to as the *score* here. (This score was computed and distributed by the track organizers.) The *score* value is intended to measure the general quality of the selected documents across a variety of systems that use this feedback as input. A higher value indicates the documents selected by our phase 1 system tended to be more useful that document selected by other phase 1 systems. The *score* is calculated on a per-query basis, and we evaluate the correlation across queries with various other measures. These measures are described below:

1. **Mean Rank**: The mean rank in our baseline ranking of the documents selected in our phase 1 selection (CMU.1).

2. **Max Rank**: The max rank in our baseline ranking of the documents selected in our phase 1 selection (CMU.1).

3. **Num. Relevant**: The number of documents selected by CMU.1 judged relevant for the query.

Table 1 shows the mean and the standard deviation of these measures and their correlations with the *score*, all computed across queries. There is not a strong correlation between the *score* value and any of the other performance measures computed over our document selection set.

Measure	Mean	Std.	Correlation with *score*
score	0.525	0.152	—
Mean Rank	10.24	5.85	0.139
Max Rank	30.0	19.59	0.115
Num Relevant	2.42	1.26	-0.030

Table 1: Document selection statistics and correlations with the *score*

3.1.3 Phase 2 Performance

Our document selection component was designed to identify documents useful for our relevance classifier and reranking components. For this reason, another appropriate method of evaluating the quality of our phase 1 input is to compare the relative improvement in phase 2 performance using our phase 1 input and other phase 1 inputs. Figure 3 shows this relative improvement as a function of the total number of relevant documents selected by that phase 1 input. For each input set, we compute the statMAP on the baseline and phase 2 run excluding those documents in the input set from each evaluation (i.e. *residual performance*). The relative improvement of a phase-2 run over the baseline is referred to as the *relative residual performance improvement* and is used as our primary measure to evaluate phase-2 performance.

There is a strong correlation between the number of relevant documents selected and the relative improvement in statMAP (Pearsons's correlation of 0.926). This is likely due to our phase 2 system ignoring non-relevant feedback documents, and suggests that focusing only on relevant feedback is not always an appropriate strategy.

We also see that, although our phase 1 selection system is moderately coupled with the phase 2 re-ranking system, it doesn't yield the best relative improvement in statMAP. These results clearly indicate that for our phase 2 system, increasing the number of relevant documents selected for feedback is an effective strategy for improving performance.

Looking deeper at the robustness of our phase-2 performance as a function of feedback documents, we evaluate the relative residual performance for all input sets as we vary the wight given to the feedback documents. Figure 4 shows the relative residual performance for each of our system's input sets as the weight on feedback documents varies from 0 to 1. The vertical line in this figure indicates the weight we used in our TREC submission (0.3) and the values along this vertical line correspond to those plotted in Figure 3. We can see that the weight selected based on our training data is not optimal for all of the input sets, but does represent a reasonable tradeoff across the different inputs. The best performing input set (CMIC.1) could have achieved almost a 13% improvement in residual statMAP had we selected a lower weight, but for most input sets the selected value is within 2% relative residual performance of the optimal weight.

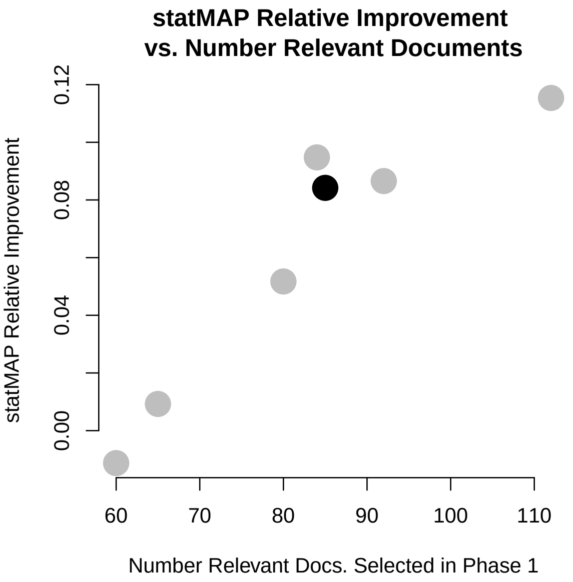

Figure 3: Relative residual performance improvement in statMAP over our baseline vs. number of relevant documents found in the input set. Each point represents a unique input set, and our phase-1 input (CMU.1) is shown in black.

Interestingly, the CMIC.1 input set, which yielded our best relative increase in statMAP, almost exclusively consists of documents from Wikipedia[4], whereas all of the other input sets consist of less than 5% Wikipedia documents. Although documents from Wikipedia may tend to be of higher general quality with less spam, these documents may be less diverse especially with regard to our link-based and URL-based document pair features. This result is somewhat contrary to the hypothesis that drove our document selection algorithm, that a diverse set of documents with respect to our feature space woud be most beneficial in final re-ranking performance.

4. CONCLUSION

In this year's submission to the TREC Relevance Feedback track, we took a machine learning approach to both the phase 1 (document selection) and phase 2 (document reranking) components of our system. These two systems use a shared feature space to represent pairs of documents. Our system specifically tried to leverage non-textual information such as webgraph features and URL similarity features, as well as textual features such as scores generated from different components of the baseline query. The shared representation moderately couples our selection and re-ranking systems, enabling us to select a set of documents specifically deemed to be useful for the down-stream re-ranking

[4] `http://en.wikipedia.org`

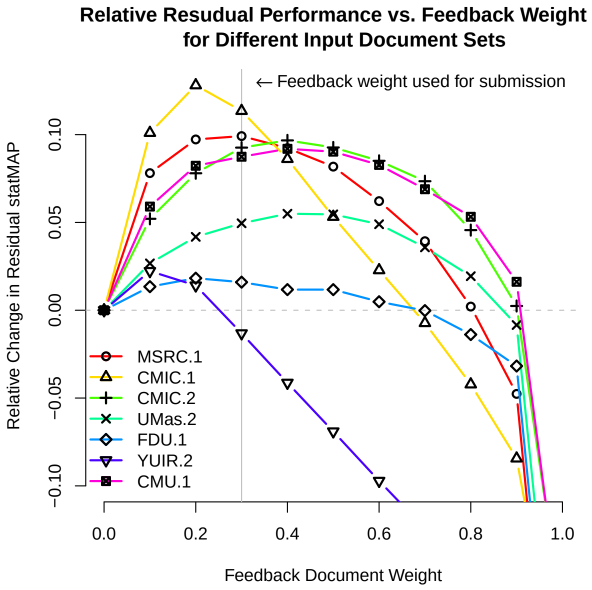

Figure 4: Relative residual statMAP for each input set as feedback document weight increases.

component.

Initial analysis suggests that phase 1 selection algorithms that identify more relevant documents yield a higher relative increase in performance for our phase 2 re-ranking system. Although our phase 1 selection system performed well, yielding almost an 8.5% relative improvement in statMAP, higher relative improvement was achieved by several other phase 1 inputs which did not share the same feature space. For this reason, it is not clear that coupling the representation used in our phase 1 and phase 2 systems yielded a significant performance boost. Further analysis is necessary to understand the effect of coupling these two systems.

One of the goals of the phase 1 selection system was to identify a diverse set of relevant documents by clustering the top-ranked documents from the baseline retrieval. This clustering was performed in the same feature space used by the relevance classification component (Section 2.3) in an effort to couple the two systems. To evaluate the effect of this coupling, future work should assess the performance of other selection mechanisms that aim to identify diverse documents, but not necessarily within the same feature space.

5. REFERENCES

[1] J. Aslam and M. Montague. Models for metasearch. In *SIGIR '01*, pages 276–284. ACM New York, NY, USA, 2001.

[2] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 1981.

[3] P. Boldi and S. Vigna. The webgraph framework I: compression techniques. In *WWW '04*, pages 595–602. ACM New York, NY, USA, 2004.

[4] K. Collins-Thompson and J. Callan. Query expansion using random walk models. In *CIKM '05*, page 711. ACM, 2005.

[5] P. Donmez and J. Carbonell. Paired sampling in density-sensitive active learning. In *ISAIM '08*, 2008.

[6] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley-Interscience, March 1990.

[7] V. Lavrenko and W. B. Croft. Relevance based language models. In *SIGIR '01*, pages 120–127, New York, NY, USA, 2001. ACM.

[8] M. Lease. Incorporating Relevance and Psuedo-relevance Feedback in the Markov Random Field Model: Brown at the TREC'08 Relevance Feedback Track. In *TREC '08*, 2008. Best results in track. This paper supersedes an earlier version appearing in conference's Working Notes.

[9] D. Metzler and W. B. Croft. A markov random field model for term dependencies. In *SIGIR '05*, pages 472–479, New York, NY, USA, 2005. ACM.

[10] D. Metzler, T. Strohman, and B. Croft. Indri TREC Notebook 2006: Lessons learned from Three Terabyte Tracks. In *TREC '06*, 2006.

[11] H. Nguyen and A. Smeulders. Active learning using pre-clustering. In *ICML '04*, pages 623–630, 2004.

[12] C. J. V. Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 1979.

[13] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Readings in information retrieval*, pages 355–364, 1997.