Using Hybrid Methods for Relevance Assessment in TREC Crowd'12

Christopher Harris[1] and Padmini Srinivasan[1,2]

[1]Informatics Program, The University of Iowa, Iowa City, IA 52242
[2]Computer Science Department, The University of Iowa, Iowa City, IA 52242

The University of Iowa (UIowaS) submitted three runs to the TRAT subtask of the 2012 TREC Crowdsourcing track. The task objective was to evaluate approaches to crowdsourcing high quality relevance judgments for a text document collection. We used this as an opportunity to examine three hybrid (combination of human-based and machine-based) approaches while simultaneously limiting time and cost. We create a training set from topics, which were previously assessed for relevance on the same document set, and use this training set to build strategies. We apply machine approaches, including clustering, to order documents for each topic, and then ask crowdworkers to provide relevance judgments for a subset of documents. One of our runs provides the best logistic average misclassification (LAM) rates of all submitted TRAT runs.

1. Approach

1.1 Overview

The TRAT (Text Relevance Assessing Task) objective was to explore new techniques to examine the ability to obtain relevance judgments on 18,260 documents using any method that did not make explicit use of the qrels for the test topics (i.e., TREC-8 topic IDs 401-450). Our approach also had the unstated objective of minimizing task time and cost. Available to us were the TREC-7 topics (topic IDs 351-400), which make use of the same data collection, the TREC-7 and TREC-8 ad hoc track submitted runs from previous years, and the TREC-7 ad hoc track qrels. Our TRAT strategy exploits information about document ranks in these previously submitted runs.

To make a training set, we randomly select ten of the fifty TREC-7 ad hoc track topics. For each of these topics, we evaluate the information from the 102 submitted TREC-7 ad hoc track runs, along with the TREC-7 ad hoc track qrels. The use of previous years' TREC submissions has been explored in other contexts (e.g., [4, 6]). We examine two types of counts, each of which represent different characteristics of the TREC-7 ad hoc track submitted runs, and then determine the appropriate weighting strategy that combines these two counts. This weighting is determined through our training runs. For each topic, we apply these weighing strategies to the TREC-8 ad hoc track submitted runs and calculate a score for each document. We subsequently rank all documents in decreasing order by this score.

From this point forward, the methods used for each of our three runs differ. For our first run (UIowaS01r), we take the list of test documents in rank score order and break them into three distinct groups: (1) definitely relevant, (2) possibly relevant and (3) definitely not relevant. We determine a fixed size for the first group of documents, which are all marked relevant. The second group is assigned to the crowd in descending rank score order for assessment. We use the crowd to determine a relevance threshold score for each topic. All documents lower in rank score than this relevance threshold are marked not relevant.

In our second run (UIowaS02r), we cluster the document collection using k-means clustering on the document headline and text fields. An appropriate k is determined (in our training runs). We order documents in rank score order within each cluster and divide them into the same three groups as we did with our first run. The overall top ranking documents across all clusters are marked as relevant. For the remaining unmarked documents, we calculate a mean rank score per cluster. The cluster with the highest mean rank score is given to the crowd to assess first. We provide the documents from this cluster to the crowd in decreasing rank score order until a relevance threshold is reached. The remaining documents in that cluster become our third group and are marked as non-relevant.

Our third run (UIowaS03r), we use the same clustering technique as we did in the second run, ranking the clusters according to the highest rank score for relevant documents for each topic, and evaluating each cluster in mean rank score order. After we mark the top 10 documents overall as relevant, we mark all documents that fall below the 40th percentile in each cluster as not relevant. For the remaining documents, which comprise the top 60% of each cluster, we randomly sample documents from the selected cluster until a relevance threshold is reached. Once that threshold is reached, the remaining documents in that cluster are marked as non-relevant and we evaluate the cluster that has the next-highest mean rank score.

1.2 Training

Ten topics from the TREC-7 ad hoc collection (topic IDs between 351 and 400) were randomly-selected and used to construct a training set. The topic IDs randomly selected were: 351, 358, 364, 369, 374, 375, 379, 388, 395, and 396.

Using these topics, we gathered the submission files. These 102 submission files represented the submitted TREC-7 ad hoc runs using a variety of methods and from different research groups, containing the topic ID, retrieved document name and a binary relevance score. A total of 14,307 unique documents referred to in these submission runs for these 10 topics.

Using these submission documents, we compute two scores for each topic:

- A *simple count*, C_S, indicates the count of submitted runs (out of 102) that included a given document.
- A *Borda count*, C_B, takes into account the rank in each submitted run for a given document.

This represents an approach similar to the one used in [1]. This Borda count is calculated as *(n-r)*, where *n* is the number of documents retrieved for a topic in a single submission file, and *r* represents the document's rank within the list (i.e., the top-ranking document in a list of 1000 documents will receive a score of (1000-1) = 999). We then sum the borda count for all All TREC-7 submissions provide 1000 or fewer documents per topic, so for each of our 102 submissions, the Borda count is in the range (0, 999).

We use both counts since they represent different properties of each training document. C_S measures the number of submissions that include that document for a topic, but does not consider its rank; C_B examines the documents rank but does not consider how many of the 102 submitted runs the document appears. For example, for a given topic, if a document exists in all 102 lists, it would receive a C_S of 102. However, if that document was ranked at the bottom of each list, the document is not likely to be relevant. Conversely, if a document was listed in only 10 of the 102 lists, but ranked at or near the top of each, C_B would be relatively high. The *count ratio coefficient*, α, represented by a value in the range (0,1), is the relative balance between these two counts for a data collection. Using these two counts (C_S and C_B) and applying the count ratio coefficient, α, we calculate a *weighted rank coefficient*, $C(d)_W$, for each document using these two separate counts for each individual document, d. A document will have a different weighted rank coefficient for each topic examined.

$$C(d)_W = \alpha\, C(d)_S + (1-\alpha)\, C(d)_B$$

A merged listing of documents was created ranked by $C(d)_W$, from highest to lowest for each topic.

We then experimented with various values of α, from 0 to 1, in increments of 0.05. A *relevant document score at α*, S_α, was determined for each topic:

$$S_\alpha = \frac{\sum_{n=1}^{d} rel(n) * C(n)_{W_\alpha}}{\sum_{n=1}^{d} rel(n)}$$

Where *rel(n)* is the binary relevance for document *n* and $C(n)_{W_\alpha}$ is the weighted sum for document *n* for a given α for one topic. S_α indicates the weighted rank of all relevant documents for a single topic for a given α; we obtain the average S_α across all ten training topics. If we rank our list by $C(d)_W$ in decreasing order and the resulting S_α is large (i.e., documents appearing at the top are relevant), it indicates the selected α bunches the relevant documents closer to the top of our list. The effect of different values of α on average relative document rank is provided graphically in Figure 2. Empirically, we determined that α = 0.8 provided the highest S_α across all training set topics. We therefore use this value for calculating our *document score*. Table 1 provides additional information about each topic used in our training set.

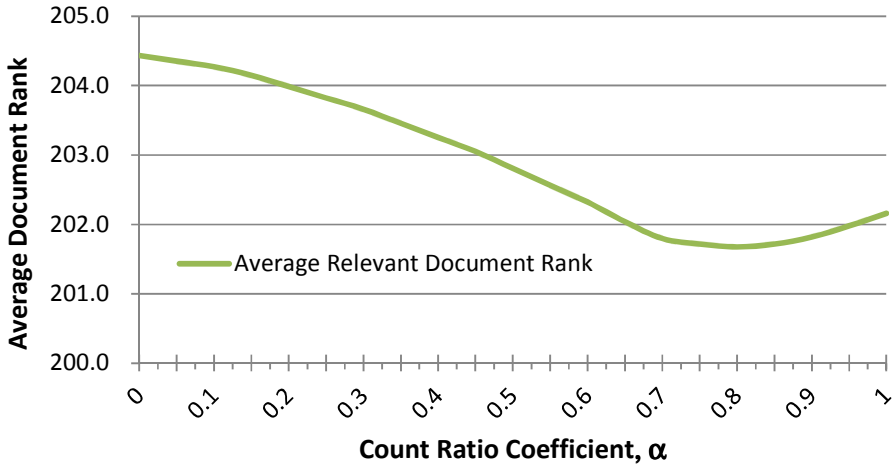

Figure 1. Relation between the document rank for 10 training topics and α

Table 1. Overview of the 10 training topics using a ranked list approach at α = 0.8

topic ID	# of unique documents for that topic ID	# of relevant documents for that topic ID	Percent relevant overall	# of relevant docs in top 10 ranked docs	Position of highest ranked relevant document	Position of lowest ranked relevant document	Average rank of relevant documents
351	1029	48	4.66	9	1	723	128.61
358	1121	51	4.55	6	3	609	122.33
364	1513	35	2.31	9	1	182	40.46
369	1706	13	0.76	5	1	102	176.43
374	1107	203	18.34	5	1	886	352.11
375	1140	80	7.02	7	1	719	250.92
379	2196	16	0.73	4	1	660	32.38
388	1467	51	3.48	7	2	1111	253.84
395	1492	213	14.28	6	1	1483	431.06
396	1536	59	3.84	9	1	1022	228.70
ALL	**Sum: 14307**	**Sum: 769**	**Avg: 6.00**	**Avg: 6.7**	**Avg: 1**	**Avg: 750**	**Avg: 201.68**

For the 10 topics in this training set, we observe that 6.0% of all documents are considered relevant by the TREC assessors. On average, 7 of the top 10 documents (as ranked by $C(d)_W$ in decreasing order) for each topic are found to be relevant. For these 10 topics, the mean lowest ranked relevant document position was 750. The mean rank for relevant documents for these 10 topics was 201.68.

2. Run UIowaS01r: Single Ranked List Method

For each topic in our training set, we ranked each document by $C(d)_W$ in decreasing order. We divided this single ranked list into batches. Our objective was to assign these documents into one of three groups:

1. All documents are relevant
2. Documents are possibly relevant
3. All documents are non-relevant

We make a manual examination of our documents, beginning with those with the highest $C(d)_W$. We determine the most appropriate size for our first group is 10 documents; that is, we mark the top 10 documents as relevant.

Next, we examine the distribution of the remaining documents. These unmarked documents make up the second and third group of documents. We empirically determine that the most appropriate batch size is 20. These are the documents we submit to the crowd for relevance assessment. To determine the threshold between the second and third document groups, we establish a rule: We submit the document batches to the crowd in descending rank order. If the crowd judges 2 consecutive batches (40 consecutive documents) as not relevant, this marks our relevance threshold and the start of our third group. We mark all the batches below this relevance threshold as not relevant.

Table 2 shows the potential merits with regard to time and cost on our training set: we are able to achieve a 92.3% recall by examining only 26.6% of documents per topic. By starting with *p* batches for each topic, we are able to assign the initial batches of documents in parallel, reducing crowd assessment time. We simulate crowd assessment in our training runs as we have access to the qrels.

Table 2. Values obtained using the simulation and fixed batch sizes.

topic ID	# of unique documents for that topic ID	# of relevant documents for that topic ID	Lowest ranked document examined	# of relevant documents found using this approach	Percent of documents examined	Percent of relevant documents found
351	1029	48	230	39	22.4	81.3
358	1121	51	310	49	27.7	96.1
364	1513	35	90	33	5.9	94.3
369	1706	13	110	13	6.4	100.0
374	1107	203	810	202	73.2	99.5
375	1140	80	630	77	55.3	96.3
379	2196	16	110	10	5.0	62.5
388	1467	51	190	38	13.0	74.5
395	1492	213	950	204	63.7	95.8
396	1536	59	370	45	24.1	76.3
ALL	**Sum:14307**	**Sum:769**	**Sum:3800**	**Sum: 710**	**Avg: 26.6**	**Avg: 92.3**

For our training set, we send 190 batches for crowd assessment. At a cost of $0.20 per batch of 20 documents, this involves a total cost to us of $40.70 for this run, including the 10% Amazon Mechanical Turk fees.

3. Runs UIowaS02r and UIowaS03r: k-means Clustering Method using Document Ranks

We used another machine-based method to divide the 14,307 documents into the three groups (relevant, maybe relevant, and non-relevant) by topic. In this method, which we use for our second and third runs, we cluster documents based on the similarity in text. We cluster using k-means clustering, a method used to partition *n* documents into *k* clusters in which each document belongs to the cluster with the nearest mean. Using Weka 3.6, we clustered using the headline and text (body) sections of each document, placing an equal weight on each of these two sections. Our initial step was to determine a value for the number of clusters, *k*. To do so, we take the following approach for each topic:

First, we rank the documents by average $C(d)_W$ in decreasing order. We mark the 10 documents with highest overall weighted counts across all clusters as *relevant*. Note that this relevancy determination is independent of our clustering approach. Next, examine the average $C(d)_W$ for each cluster. Our goal is to have a few clusters that contain a majority of the relevant documents and other clusters that contain very few or no relevant documents. Therefore, we examine the variance of average cluster $C(d)_W$ for each value of k. With a fixed number of relevant documents, a large variance implies more variation, in terms of relevant documents, across clusters. We evaluated ranges of k between 5 and 25 and empirically determine that k=18 provides the highest average cluster variance and we use this value for k.

For run UIowaS02r, we begin with the cluster with the largest average $C(d)_W$. Within this cluster, we ask the crowd to assess relevance in batches of 20 documents, stopping once the crowd indicates 2 consecutive batches (40 documents) are all non-relevant. We mark the remaining documents in that cluster (in order of $C(d)_W$) as non-relevant. We then move to the cluster with the next-largest average $C(d)_W$ and repeat the process for that cluster.

For run UIowaS03r, we also begin with the cluster with the largest average $C(d)_W$. We mark all documents that fall below the 40th percentile in each cluster as not relevant. We choose to evaluate the top 60% of documents as an empirical evaluation of our training set found very few relevant documents appeared in the bottom 40%. For the remaining unmarked documents, which comprise the top 60% of each cluster, we randomly sample documents from the selected cluster and provide them to the crowd for assessment. We continue with documents from this same cluster until two consecutive batches (40 documents) are found to have no relevant documents. Once that threshold is reached, the remaining documents in that cluster are marked as non-relevant. We then move to the cluster with the next-largest average $C(d)_W$ and repeat the process for that cluster.

Table 3 (next page) provides the cluster sizes and some basic information on the test set averages obtained for each of the 18 clusters across all topics. This information was used for Runs UIowaS02r and UIowaS03r.

4. Results

The logistic average misclassification rate (LAM) is used as the evaluation metric. LAM is defined as

$$LAM = logit^{-1}\left(\frac{logit\,(fnr) + logit\,(fpr)}{2}\right)$$

where *fnr* is the smoothed false negative rate and the *fpr* is the smoothed false positive rate.

$$fpr = \frac{|FP| + 0.5}{|FP| + |TN| + 1}$$

$$fnr = \frac{|FN| + 0.5}{|FN| + |TP| + 1}$$

The logit function (and its inverse) are defined as:

$$logit\ (p) = \log\frac{p}{1-p}$$

$$logit^{-1} = \frac{e^x}{1-\ e^x}$$

Thus, lower values of LAM are desirable. We provide our results for run UIowaS01r (Table 4), run UIowaS02r (Table 5) and run UIowaS03r (Table 6). We achieve our best results (and the best results for all TRAT submissions overall) with our second run, which combines ensemble methods (scoring submissions from a number of different techniques), human computation methods (crowdsourcing relevance judgments) and machine methods (clustering).

Table 3. Number of batches evaluated for each Topic ID. Note each run had an initial top 10 ranked documents judged.

.Topic ID	UIowaS01r	UIowaS02r	UIowaS03r
411	6	7	6
416	11	15	16
417	15	19	12
420	10	15	16
427	8	14	16
432	7	14	10
438	25	29	24
445	22	27	24
446	22	30	28
447	5	6	5
ALL	**Sum: 133**	**Sum: 176**	**Sum: 157**

Table 4. Results per topic ID for UIowaS01r.

Test Topic ID	Documents in Collection	Marked as Relevant (TP + FP)	Correctly Identified (TP)	All Relevant (TP + FN)	Run1 Results (LAM)	TRAT Task Mean (LAM)
411	2056	22	15	27	0.052	0.15
416	1235	48	41	45	0.026	0.16
417	2992	60	52	75	0.035	0.20
420	1136	37	27	37	0.057	0.17
427	1528	39	24	37	0.071	0.18
432	2503	17	1	22	0.236	0.27
438	1798	102	94	162	0.058	0.26
445	1404	61	48	60	0.049	0.19
446	2020	108	93	156	0.070	0.21
447	1588	29	14	16	0.040	0.08
ALL	**Sum: 18260**	**Sum: 523**	**Sum: 409**	**Sum: 637**	**Avg: 0.069**	**Avg: 0.187**

Table 5. Results per Topic ID for UIowaS02r.

Test Topic ID	Documents in Collection	Marked as Relevant (TP + FP)	Correctly Identified (TP)	All Relevant (TP + FN)	Run 2 Results (LAM)	TRAT Task Mean (LAM)
411	2056	26	20	27	0.033	0.15
416	1235	54	43	45	0.023	0.16
417	2992	72	62	75	0.027	0.20
420	1136	53	31	37	0.062	0.17
427	1528	54	32	37	0.048	0.18
432	2503	26	8	22	0.102	0.27
438	1798	128	109	162	0.071	0.26
445	1404	62	53	60	0.031	0.19
446	2020	115	101	156	0.061	0.21
447	1588	27	16	16	0.015	0.08
ALL	**Sum: 18260**	**Sum: 617**	**Sum: 475**	**Sum: 637**	**Avg: 0.047**	**Avg: 0.187**

Table 6. Results per Topic ID for UIowaS03r.

Test Topic ID	Documents in Collection	Marked as Relevant (TP + FP)	Correctly Identified (TP)	All Relevant (TP + FN)	Run 3 Results (LAM)	TRAT Task Mean (LAM)
411	2056	26	20	27	0.033	0.15
416	1235	53	42	45	0.028	0.16
417	2992	64	55	75	0.034	0.20
420	1136	41	26	37	0.073	0.17
427	1528	51	29	37	0.062	0.18
432	2503	18	2	22	0.190	0.27
438	1798	84	70	162	0.098	0.26
445	1404	46	39	60	0.052	0.19
446	2020	86	75	156	0.076	0.21
447	1588	26	16	16	0.014	0.08
ALL	**Sum: 18260**	**Sum: 495**	**Sum: 374**	**Sum: 637**	**Avg: 0.066**	**Avg: 0.187**

We acknowledge there are a number of crowdsourcing techniques that would likely improve our results, such as spam detection, incentives, or having overlapping assessments made by different crowdworkers and applying a voting method. We believe that not having multiple assessments negatively impacted worker accuracy.

Like other groups, we struggled with several of the topics, particularly topic 432 ("Do police departments use 'profiling' to stop motorists?"). This may be due to the difficulty of crowdworkers who reside outside of the United States to understand the nature of this information need, since profiling is not a universally-known concept of police methods. Other information needs that have greater concept transferability did better, such as topic 416 ("What is the status of The Three Gorges Project?"), which clearly refers to a massive construction project in China that is well-documented in the press, and is well-known globally.

5. Judgment Cost and Time

We performed all of our relevance judgments using Amazon Mechanical Turk. We paid $0.20 for each batch of 20 document assessments, or $0.01 per assessment, which is a customary rate for a binary relevance assessment. Of the 112 different crowdworkers who performed judgments on our three runs, we had 31 crowdworkers (27.7%) evaluate more than a single batch. Table 7 examines the cost for each of our three runs.

Table 7. Analysis of cost for each of our runs.

Run	Documents Assessed	Percent of Docs Assessed	Cost (including AMT service fee)	LAM	Cost per Relevant Document
UIowaS01r	2660	14.6	$29.26	0.069	$0.072
UIowaS02r	3520	19.3	$38.72	0.047	$0.082
UIowaS03r	3140	17.2	$34.54	0.066	$0.092

The cost for UIowaS02r is slightly higher per relevant document found than for UIowaS01r, but all are cheaper than UIowaS03r, which uses random sampling. Overall, since we expected to outsource roughly 26% of the documents, the results in Table 7, particularly column indicating the percent of documents assessed, shows we overestimated the number of documents we believed the assessors would evaluate from our training set. Roughly 17% of documents were sent to the crowd for assessment. This is likely due to a different distribution of relevant documents from our training set or that our crowd assessors were excessively conservative in their assessment of relevant documents.

In Table 8, we evaluate the time taken for each of our runs.

Table 8. Analysis of time taken for each of our runs.

Run	# Docs Assessed	Task Time taken (hours)	Time taken per batch of 20 (min)	LAM	Time per Relevant Document (min)
UIowaS01r	2660	42	9.8	0.069	6.161
UIowaS02r	3520	51	10.3	0.047	6.442
UIowaS03r	3140	47	10.1	0.066	7.540

From Table 8, we see that the time taken for our second run, with more assessments, takes the longest to complete. The tradeoff between time taken and the improvement in the LAM rate for our runs indicates the method used has an impact. The methods that apply our ranking approach (runs UIowaS01r and UIowaS02r) are more efficient than the run that uses random samples (UIowaS03r).

These time and cost numbers do not count the real cost of obtaining the submitted runs for our test set, since these represent important inputs to our process. The run that is least dependent on submitted run information (UIowaS03r) had the highest LAM rate, indicating the power of the ensemble method to reduce LAM rates.

6. Limitations of our Methods

We identify some of the limitations of our methods. They are as follows:

1. There is a difference between the test and training sets. The distribution of the data between the test and training set topics might be different. A preliminary assessment of the test topics indicates their underlying distribution is quite different for 4 of the 10 topics.

2. Reliance on the crowd to set relevance thresholds. The crowd determines which documents are relevant and also when we stop our assessment. One careless crowdworker could grab two consecutive batches for a topic, mark them all non-relevant, and we would subsequently miss many important relevance judgments for that topic that appear lower in our ranked list.
3. Lack of anti-spam crowdsourcing techniques. Due to time constraints on our part, we did not integrate any of the voting techniques, honey pots, or other types of crowdsourcing quality checks that are now commonplace, which affected the precision of our results. This is easily addressed through the application of voting mechanisms and incentives, as has been discussed in [2, 3, 5]. We also did not limit the number of batches a single crowdworker could assess, which may potentially bias our results.

7. Summary

For the TREC Crowd'12 TRAT task, we used a hybrid method for each of our three runs; however the hybrid method used differs slightly for each run. Central to each method is a calculation of a weighted score for each document for each topic. We rank our list of documents using this score. We then set out to break the list into three groups of documents for that topic – definitely relevant, possibly relevant, and definitely non-relevant. We take the top 10 ranked documents as our definitely relevant group. The possibly relevant and definitely non-relevant groups were submitted to the crowd for relevance assessment.

For run UIowaS01r, we score our documents and then create a single ranked list. We send documents to the crowd in decreasing rank score order. The crowd continued to assess these documents and if there was at least 1 relevant document out of 40 consecutive documents, we continued to send documents to the crowd. Our preliminary assessment using training data from the TREC 7 ad hoc track showed we could assess 92.3% of relevant documents only using 26.6% of the document set. In our test set, we used only 17% of the document set, but we did not find as many relevant documents. Thus, our method is dependent on test and training sets have a very similar distribution.

For runs UIowasS02r and UIowaS03r, we performed k-means clustering of all the documents. Using 18 clusters and the ranked list of each document for each topic, we applied two slightly different methods of obtaining documents form these clusters to send to the crowd for assessment. Run UIowaS02r results, which made better use of our ranking methods, performed better than our third run, UIowaS03r, which instead used a random sampling method. All three of our runs were above the mean LAM rate for all submitted runs. Although significance testing was not performed, our best results we obtained using a combination of methods, which illustrates the merits of this hybrid approach.

We believe much of our strong performance is due to an ensemble method. We use an ensemble method to establish ranking and weights, machine approaches (clustering), and a human computation approach (crowd-based relevance assessment). We derive our weights using the submissions for 129 training runs from many different participants. As with crowdsourcing in general, the use of a number of different submission approaches provides allows us to obtain ensemble-like results, which reduce the risk of applying a single method. However, we note that distributional differences between our training and test results may present biases that skew the document weights upon which we rely, undermining many aspects of our experiment. In most experimental settings, submission run data is not typically available, limiting our ability to extrapolate these results to real-world scenarios.

Although not a stated TRAT task objective, we also examine how the methods used for our three runs vary in terms of quality (in terms of LAM) relative to the time and cost needed. Our first run (UIowaS01r) requires the fewest number of documents to be evaluated (and the lowest time and costs); our second run (UIowaS02r) requires the most. Although further analysis needs to be performed, our initial observation is that the slight extra time and cost required for UIowaS02r results in a much larger improvement in the LAM rate.

Last, we highlight some limitations of our approach. Among these are a dependency on a similar distribution of relevant documents between our training and our test sets, a heavy reliance on the crowd to determine when we stop our assessment for a given topic, and a lack of employing anti-spam crowdsourcing techniques, which could permit sloppy or malevolent behavior to occur in our assessment tasks. These represent directions of future work in this area.

References

1. Almquist, B., Mejova, Y., Ha-Thuc, V., & Srinivasan, P. (2008). *University of Iowa at TREC 2008 Legal and Relevance Feedback Tracks*. In *Proceedings of the 17th Text Retrieval Conference*, (TREC'08) Gaithersburg, MD.
2. Alonso, O., & Mizzaro, S. (2009). Can we get rid of TREC assessors? Using Mechanical Turk for relevance assessment. In *Proceedings of the SIGIR 2009 Workshop on the Future of IR Evaluation* (pp. 15-16).
3. Alonso, O., & Mizzaro, S. (2012). Using crowdsourcing for TREC relevance assessment. *Information Processing & Management*. 48(6). (pp. 1053-1066).
4. Chu-Carroll, J., Czuba, K., Prager, J., Ittycheriah, A., & Blair-Goldensohn, S. (2004). IBM's PIQUANT II in trec 2004. *Proceedings of the 13th Text Retrieval Conference*, (TREC'04) Gaithersburg, MD.
5. Eickhoff, C., Harris, C. G., de Vries, A. P., & Srinivasan, P. (2012). Quality through flow and immersion: gamifying crowdsourced relevance assessments. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval* (pp. 871-880). ACM.
6. Sanderson, M., & Zobel, J. (2005). Information retrieval system evaluation: effort, sensitivity, and reliability. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 162-169). ACM.