The University Carlos III of Madrid at TREC 2011 Crowdsourcing Track

Julián Urbano, Mónica Marrero, Diego Martín, Jorge Morato, Karina Robles and Juan Lloréns

University Carlos III of Madrid · Department of Computer Science
{jurbano, mmarrero, dmandres, jmorato, krobles, llorens}@inf.uc3m.es

Abstract. This paper describes the participation of the uc3m team in both tasks of the TREC 2011 Crowdsourcing Track. For the first task we submitted three runs that used Amazon Mechanical Turk: one where workers made relevance judgments based on a 3-point scale, and two similar runs where workers provided an explicit ranking of documents. All three runs implemented a quality control mechanism at the task level based on a simple reading comprehension test. For the second task we also submitted three runs: one with a stepwise execution of the GetAnotherLabel algorithm and two others with a rule-based and a SVM-based model. According to the NIST gold labels, our runs performed very well in both tasks, ranking at the top for most measures.

1 Introduction

The TREC 2011 Crowdsourcing Track was designed to investigate how to better use crowdsourcing platforms to evaluate Information Retrieval systems. The Track was divided in two tasks: obtaining topical relevance judgments from individual workers and computing consensus judgments from several workers. The Knowledge Reuse research group at the University Carlos III of Madrid put together a team of six people to participate in both tasks. We submitted three runs for each task, although most of our work was devoted to the first one. We focused on designing a practical and effective task template for gathering unconventional relevance judgments through Amazon Mechanical Turk, while studying quality control mechanisms at the task level for such heterogeneous documents as arbitrary HTML pages from the Web. In the second task, two of our runs used a rule-based and an SVM Machine Learning model, while the other one followed a stepwise execution of the GetAnotherLabel algorithm by Ipeirotis et al. [2010].

The rest of the paper is organized as follows. Section 2 describes our submissions for the first task, detailing the HIT design, document preprocessing and quality control mechanism. Section 3 describes our submissions for the second task, and Section 4 summarizes the results in both tasks. Section 5 concludes with final remarks and lines for further work.

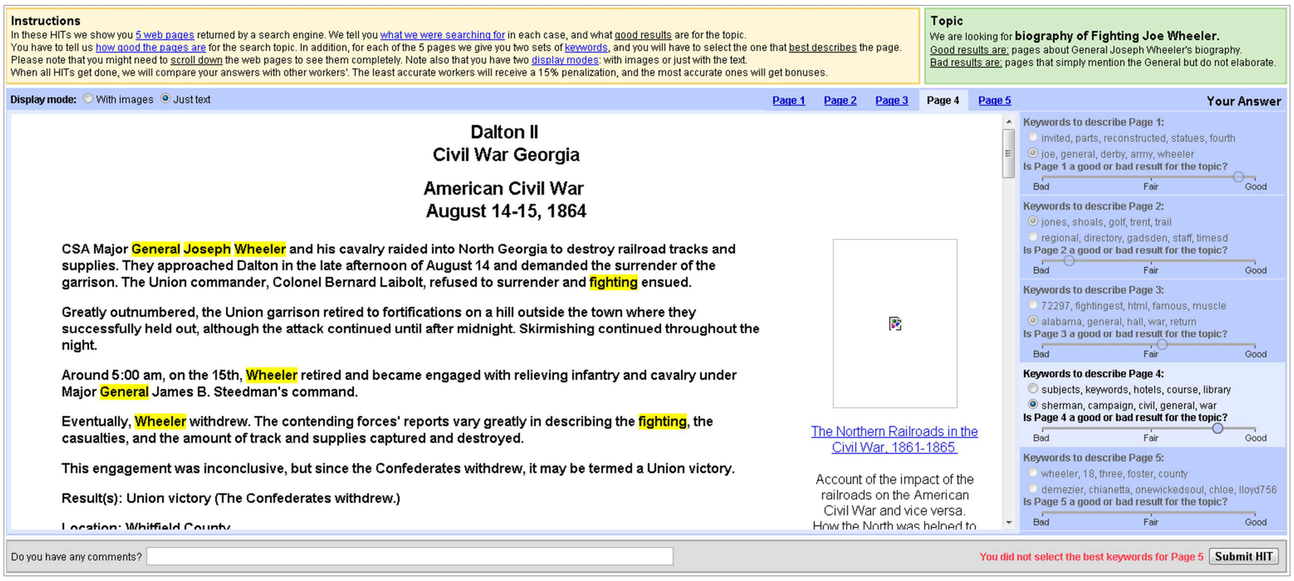
2 Task I: Crowdsourcing Individual Judgments

For the first task we submitted three runs (see Table 1), all of which used Amazon Mechanical Turk (AMT) as the crowdsourcing platform. The task was implemented with external HITs, that is, we hosted the templates and data in our own server, communicating with AMT via the API. This allowed us to have more control over the whole process, besides the possibility of gathering some additional data such as knowing when and for how long workers previewed our HITs or where they came from.

Run	uc3m.graded	uc3m.slider	uc3m.hterms
Uploaded	Sep 12th, 19:22 CEST	Sep 13th, 17:48 CEST	Sep 14th, 18:44 CEST
Hours to complete	8.5	38	20.5
HITs submitted (overhead)	438 (+1%)	530 (+22%)	448 (+3%)
Workers who submitted (just previewers)	29 (80)	86 (354)	33 (175)
Average judgments per worker	76	32	75
Cost (fees)[1]	$87 ($8.7)	$87 ($8.7)	$87 ($8.7)

Table 1. Summary of the runs submitted for Task I.

[1] This is the total cost if rejected work were not paid. See Section 2.4.4.



Instructions
In these HITs we show you 5 web pages returned by a search engine. We tell you what we were searching for in each case, and what good results are for the topic.
You have to tell us how good the pages are for the search topic. In addition, for each of the 5 pages we give you two sets of keywords, and you will have to select the one that best describes the page.
Please note that you might need to scroll down the web pages to see them completely. Note also that you have two display modes: with images or just with the text.
When all HITs get done, we will compare your answers with other workers'. The least accurate workers will receive a 15% penalization, and the most accurate ones will get bonuses.

Topic
We are looking for **biography of Fighting Joe Wheeler.**
Good results are: pages about General Joseph Wheeler's biography.
Bad results are: pages that simply mention the General but do not elaborate.

Display mode: With images [✓ Just text]

Page 1 Page 2 Page 3 Page 4 Page 5

Dalton II
Civil War Georgia

American Civil War
August 14-15, 1864

CSA Major General Joseph Wheeler and his cavalry raided into North Georgia to destroy railroad tracks and supplies. They approached Dalton in the late afternoon of August 14 and demanded the surrender of the garrison. The Union commander, Colonel Bernard Laibolt, refused to surrender and fighting ensued.

Greatly outnumbered, the Union garrison retired to fortifications on a hill outside the town where they successfully held out, although the attack continued until after midnight. Skirmishing continued throughout the night.

Around 5:00 am, on the 15th, Wheeler retired and became engaged with relieving infantry and cavalry under Major General James B. Steedman's command.

Eventually, Wheeler withdrew. The contending forces' reports vary greatly in describing the fighting, the casualties, and the amount of track and supplies captured and destroyed.

This engagement was inconclusive, but since the Confederates withdrew, it may be termed a Union victory.

Result(s): Union victory (The Confederates withdrew.)

Location: Whitfield County

The Northern Railroads in the Civil War, 1861-1865

Account of the impact of the railroads on the American Civil War and vice versa. How the North was helped to

Your Answer

Keywords to describe Page 1:
invited, parts, reconstructed, statues, fourth
[✓ joe, general, derby, army, wheeler]
Is Page 1 a good or bad result for the topic?
Bad Fair Good

Keywords to describe Page 2:
[✓ jones, shoals, golf, trent, trail]
regional, directory, gadsden, staff, timesd
Is Page 2 a good or bad result for the topic?
Bad Fair Good

Keywords to describe Page 3:
72297, fightingest, html, famous, muscle
[✓ alabama, general, hall, war, return]
Is Page 3 a good or bad result for the topic?
Bad Fair Good

Keywords to describe Page 4:
subjects, keywords, hotels, course, library
[✓ sherman, campaign, civil, general, war]
Is Page 4 a good or bad result for the topic?
Bad Fair Good

Keywords to describe Page 5:
wheeler, 18, three, foster, county
demezier, chianetta, onewickedsoul, chloe, lloyd756
Is Page 5 a good or bad result for the topic?
Bad Fair Good

Do you have any comments?

You did not select the best keywords for Page 5 Submit HIT

Figure 1. HIT design for run uc3m.hterms.

2.1 HIT Design[2]

We worked on the HIT template to make it as simple, functional and self-contained as possible. Given that all five documents in a set had to be judged by the same worker, we decided to include them all in a single HIT to make the assignment process easier and allow workers to see all five documents at once to make the ranking judgments easier. Thus, we had a total of 435 HITs for a total of 2175 judgments. We paid \$0.20 plus fees per HIT, which makes it \$0.04 per document. It was in our plans to try paying a little less, but given the tight deadlines we decided to stick with the \$0.04 wage for the sake of time [Mason et al., 2009]: workers seemed to get interest in our task, and in previous trials we asked them if they thought the payment was fair, and they mostly thought so.

All three runs shared the same template except for the HIT description, instructions and the part where the actual relevance questions were asked (see Figure 1). The top left box contained the task instructions. For the uc3m.graded and uc3m.slider runs the focus of the task was on the quality control questions (see Section 2.4.3), making the relevance question secondary (see Section 2.3). For the uc3m.hterms run, we made it the other way around. We informed that the most accurate workers would get a bonus, while the least accurate would receive a 15% penalization [Shaw et al., 2011]. However, due to mere programming difficulties, we ended up paying bad workers too to avoid unnecessary conflicts for the time being. Some fragments of the instructions were rendered as hyperlinks (see Figure 1), and when clicked upon they triggered a highlighting effect on the part of the template they referred to, so that workers could immediately follow up while reading. For instance, when clicking on “what we were searching for” the box with the topic description would be highlighted blinking thrice.

The top right box contained the topic description, with the title in a bigger bold face. Underneath, two lines briefly described what good and bad results were considered for the topic. These descriptions were not directly taken as in the files distributed for the track; we made a simpler version, explaining concepts when appropriate.

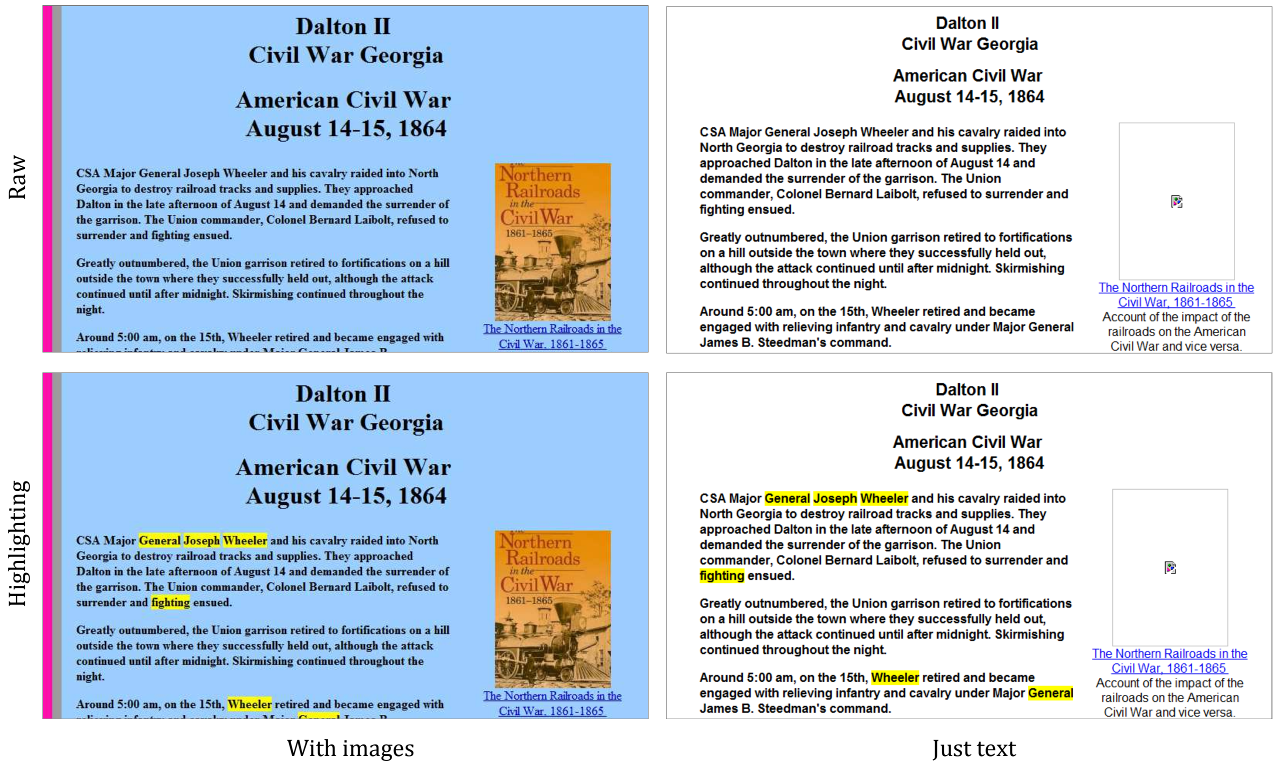
The middle right area of the template contained five stacked boxes where workers had to give their answers, one box per document. First, we asked for the answer to the quality control question (see Section 2.4), which consisted in selecting one correct answer out of two options. Next, we asked the actual relevance question for the document selected. Unlike the quality control question, the relevance question was different in all three runs (see Section 2.3).

The five documents in the set were displayed below the instructions and topic description using a tabbed design where workers could display one document or another by clicking on the corresponding tab header. This way, there is no need to follow external links or do long scrolls throughout the HIT to go from document to document. The tab and answer boxes corresponding to the document currently being displayed were rendered with a lighter color, while the others' where darker and with all options disabled, so that workers could easily know which of the five answer boxes belonged to the document and make no mistake. Tab headers were located in the upper right corner, while the upper left corner offered two different options to display documents:

[2] All HIT templates and gathered data can be downloaded at http://www.kr.inf.uc3m.es/trec2011/.

- With images. This mode kept CSS styles, layout, images, etc. However, we removed all scripts, embedded objects and all HTML elements not related with the page rendering (see Section 2.2). Therefore, workers saw the documents nearly as they would if surfing the Web themselves.
- Just text. This mode rendered documents as in the images mode, but removing all images, colors, custom formatting and all other elements not related with the page layout. This results in a simple black and white view, while still maintaining the headers and layout scheme of the original page.

In early trials we asked workers which of the two display modes they preferred. Most of the times they chose the mode with images, so we made it the default one. However, some workers always selected the just text mode, so we decided to keep it anyway and let workers decide. The bottom of the HIT contained a simple box with a textbox to provide optional feedback (they rarely did), and in the bottom right corner we placed the submission button. When clicking this button, a script checked that workers answered all five relevance questions and all five quality control questions. If not, they were informed with a message displayed in red font right before the submission button, keeping the browser in the same page.



Dalton II
Civil War Georgia

American Civil War
August 14-15, 1864

CSA Major General Joseph Wheeler and his cavalry raided into North Georgia to destroy railroad tracks and supplies. They approached Dalton in the late afternoon of August 14 and demanded the surrender of the garrison. The Union commander, Colonel Bernard Laibolt, refused to surrender and fighting ensued.

Greatly outnumbered, the Union garrison retired to fortifications on a hill outside the town where they successfully held out, although the attack continued until after midnight. Skirmishing continued throughout the night.

Around 5:00 am, on the 15th, Wheeler retired and became engaged with relieving infantry and cavalry under Major General James B. Steedman's command.

The Northern Railroads in the Civil War, 1861-1865
Account of the impact of the railroads on the American Civil War and vice versa.

Figure 2. Document display modes for runs uc3m.graded and uc3m.slider (top) and uc3m.hterms (bottom).

2.2 Document Preprocessing

Documents displayed to workers were the result of a previous cleanup process to ensure smooth loading and safe rendering, embedding all contents within a single HTML file so no additional requests were made to any server. Documents displayed in the mode with images were processed as follows:

1. All hyperlinks were modified to point to "#", so that clicking on them would not trigger any page loading.
2. All CSS rules in external stylesheets were joined and put together in a single style tag within the document.
3. All CSS rules unrelated to style or layout were removed.
4. Unsafe or irrelevant HTML elements, such as scripts and applets, were removed too.
5. All HTML attributes unrelated to style or layout were removed.
6. All images were loaded and embedded within their a tag, using base64 encoding.

Documents displayed in the mode with just text underwent three more processing steps:

1. The source of all images was removed. That is, workers could see where images appeared in the document, but no image was shown at all (see Figure 2, right).
2. HTML attributes related to style were removed, keeping the ones related to layout.
3. Background colors were all set to white, font colors to black, and the font family set to Sans Serif of 11px.

Both document versions were modified for the uc3m.hterms run to have key terms highlighted. We removed stopwords from each topic description, and obtained the stems of the remaining words. Then, we compiled all documents to judge for that topic, and for every word with a matching stem we set the background color to yellow and font color to black, wrapping the word with an HTML span tag (see Figure 2, bottom).

All these document versions were stored in our server and sent on demand whenever a worker clicked on a document tab header. Runs uc3m.graded and uc3m.slider allowed workers to see documents with and without images, and run uc3m.hterms used those same two versions but with the topic key terms highlighted. Many workers preferred to use the just text version: 7 (24%) in uc3m.graded, 21 (24%) in uc3m.slider, and 12 (36%) in uc3m.hterms. Apparently, a larger proportion in the latter preferred this version, because the highlighted terms can be more easily glimpsed with a black and white document.

2.3 Relevance Question

We explored two lines of work for obtaining the actual relevance judgments (see Figure 3). In uc3m.graded we tried to optimize for the binary relevance label, while for runs uc3m.slider and uc3m.htrems we tried to optimize for the ranking labels.

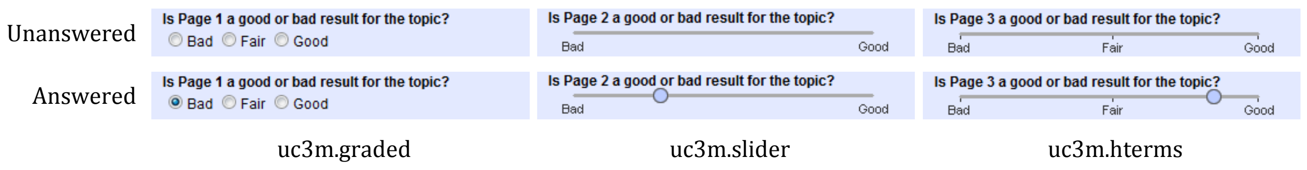

Unanswered
Answered
Is Page 1 a good or bad result for the topic?
Bad
Fair
Good
Is Page 2 a good or bad result for the topic?
Bad
Good
Is Page 3 a good or bad result for the topic?
Bad
Fair
Good
uc3m.graded
uc3m.slider
uc3m.hterms

Figure 3. Relevance question widgets.

2.3.1 uc3m.graded

For the first run we used a traditional 3-point graded relevance scale where documents could be judged as *bad*, *fair* or *good* for the search topic (see Figure 3, left). Even though the Track asks for binary labels, using three levels provides more information to adjust for a probability-based binary label as well as for the ranking labels.

For the probabilistic binary labels we set the *bad* documents as 0, the *good* documents as 1, and then studied different probabilities for the *fair* documents. We tried values from 0 to 1, with increments of 0.1, and the results with the known judgments supported the choice of 1 for the *fair* documents too (see Section 4.1). For the ranking labels, we ordered documents by their relevance level, as reported by workers. We broke ties ordering again by the number of failures in quality control (see Section 2.4) and then by the time spent on each document (descending). We thus assumed that, given two documents with the same relevance judgment, the one for which workers failed the quality control question or spent little time is less likely to be relevant. We tried other attributes and orderings, but they were marginally worse.

2.3.2 uc3m.slider

For the second run we investigated how to have workers provide an explicit ranking of the five documents in the set without much trouble. Preference judgments would make more sense for a relatively large number of documents, but for just five documents we did not consider it a viable option because of the additional cost. We also thought about using a 5-point relevance scale without ties, but such an option would not be very practical for little over just five documents, and still we would not be able to obtain a clear boundary between the relevant and not relevant documents for the binary scale.

Instead, we gave workers a slider widget they could use to give each document a certain amount of relevance (see Figure 3, middle), from 0 to 1 with increments of 0.01, though workers could not see the actual value set. This would give us a complete ranking of the documents and something like a relevance distance between them: documents similarly relevant to the topic would have their slider handles close together, and documents too different in terms of relevance would have them far apart. Intuitively, workers would move the slider handle to the right for the relevant documents and to the left for nonrelevant ones, even form little clusters with the slider handles for documents of similar relevance. Using a simple clustering algorithm we could infer the binary relevance label out of these clusters.

In early trials the slider handle was shown in the middle from the beginning, and workers had to move it in either direction, even if they wanted it to be exactly in the middle. This had a clear advantage for binary labels: if the slider handle were moved to the right side, even if just a little, we could assign a binary label of 1 because the right end of the slider was labeled as *good*. Similarly, if moved to the left side we could assign a label of 0. The downside was that some workers moved the slider handle just 5-10 pixels, being hardly informative for the ranking labels. Most probably, these were careless workers trying to make easy money, but they could as well be honest workers that just wanted to indicate mid-relevant documents. In contrast, some others tended to place the slider handles at the end points (see Figure 4, left), providing little information for the ranking labels too. Probably they just did not understand the slider widget and just clicked on the end labels. We finally decided to hide the slider handle at the beginning, as an indication that no value was set (see Figure 3, top), avoiding any initial bias too. Workers had to click on the slider bar to set an initial value.

For the ranking label we just used the slider positions provided by workers, and broke ties by the number of failures in quality control and then by the time spent on each document (descending), just like in the uc3m.graded run. For the binary label we studied several options:

- Range-Normalized. The probabilistic binary label is just the slider value from 0 to 1.
- Range-Normalized with Threshold. All documents with a slider value larger than or equal to a threshold *t* would be assigned a binary label of 1 and 0 otherwise. We tried threshold values from 0 to 1, with increments of 0.1.
- Set-Normalized. The slider value is normalized between the minimum and maximum judgments in the set of five documents. This option could correct for worker bias, although it might suffer learning effects and it assumes there is a highly relevant document and a highly nonrelevant document in the set of five.
- Set-Normalized with Threshold. The Set-Normalized value is computed, and if it is larger than or equal to a threshold *t* we assign a binary label of 1 and 0 otherwise. We tried values for *t* from 0 to 1 with increments of 0.1.
- Worker-Normalized. We compute the range of values given by the worker for all documents (i.e. her global minimum and maximum), and then normalize the slider values within that range. This option could correct for possible biases per worker, although it makes the similarly unjustified assumption that all workers judge at least one highly relevant document and one highly nonrelevant. In addition, it could clearly suffer from learning effects.
- Worker-Normalized with Threshold. The Worker-Normalized value is computed, and if it is larger than or equal to a threshold *t* we assign a binary label of 1 and 0 otherwise. Similarly, we tried values for *t* from 0 to 1 with increments of 0.1.
- Cluster. Given the slider values for all five documents in the set, we ran a k-means clustering algorithm for 2 clusters: documents in the left cluster are assigned a binary label of 0, and those in the right side are assigned a label of 1. This again assumes there are both relevant and nonrelevant documents in the set.

These options were evaluated with the known labels of the test set provided for the Track, and we chose the Set-Normalized labels with Threshold *t*=0.4 (see Section 4.1).

2.3.3 uc3m.hterms

For the third run we followed the same line as in the second run, but we decided to include another label in the slider widget marking the middle point as *fair* (see Figure 3, right). Our hope was that workers would better understand that they had a wide range of possible values to set, not just the *bad* and *good* extremes. However, they seemed to just set more values around the middle label as well (see Figure 4, right).

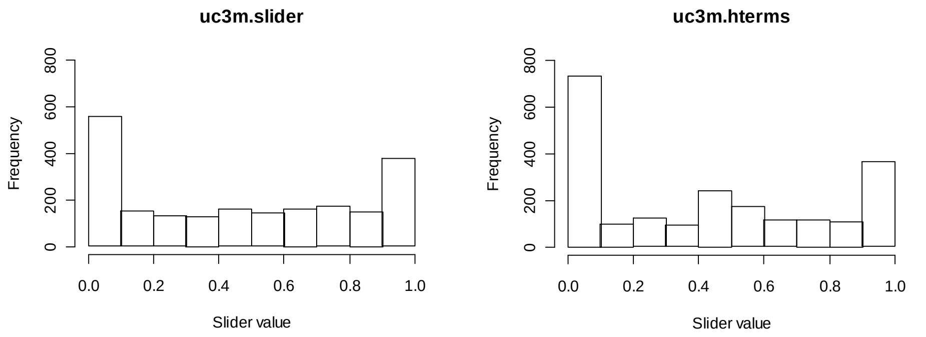


Figure 4. Distribution of slider values assigned by workers in runs uc3m.slider (left) and uc3m.hterms (right). It can be seen how they tended to click in the end points and, in the second case, also towards the center.

We tried the same methods to generate the binary labels, and the Set-Normalized method with Threshold *t*=0.4 yielded again the best results (see Section 4.1).

2.4 Quality Control

There is no warranty that the answers provided by workers are correct (to the extent there is an objective and correct answer at all in this task [Voorhees, 2000]). We thus decided to implement some quality control mechanisms, which work at different levels:

- Worker level, where only workers that meet certain demographic criteria are allowed to work in the task. In the case of Amazon Mechanical Turk, they can be filtered by their percent of work accepted, geographic location, etc.
- Task level, where the task template is modified to incorporate some additional information about the individual worker response that could indicate misbehavior. This information can be implicit, such as the time needed to complete the task [Zhu et al., 2010][Urbano et al., 2010] or behavioral patterns such as mouse clicks, scrolling and key strokes [Rzeszotarski et al., 2011]. It can also be gathered explicitly, actually asking additional secondary questions with quantitative, unbiased and automatically computable answers [Kittur et al., 2008]. If giving correct answers to these questions required as much effort as answering the main question, we could assume that if the secondary answer is correct then we can trust the answer to the main question.
- Process level, where the task assignment process is modified so that workers receive examples for which the correct answer to the main question is known beforehand (i.e. trap questions), allowing us to assess how well the worker responses fit the known answers [Sanderson et al., 2010] and follow a learning process accordingly [Le et al., 2010].

The subset of known labels provided in the test set suggests the use of trap questions as the quality control mechanism at the Process level. However, we decided not to pursue this line for several reasons. First, these known labels would need to be generated in real evaluation settings, with the problem of choosing what topics (if not all of them) and what documents to judge, increasing the workload in a real evaluation experiment via crowdsourcing. Second, the labels would not necessarily be balanced across relevance levels like in the data used in the Track, so that many judgments could be needed beforehand to get a meaningful gold set to use as trap questions. Third, using these questions involve an overhead cost, whether adding just one document per set or having workers judge all five documents in a known set. Fourth, the assignment process is more complex, and most importantly, trap questions may ensure that workers understood and paid attention to those particular topics and documents, but not necessarily to all others.

Instead, we decided to investigate quality control mechanisms at the Task level. As an implicit measure we used the work time per document, taking into account both the HIT previewing and working times; and as an explicit measure we used a very simple reading comprehension test based on a selection of the keywords that best describe the documents. We also used the Worker level control provided by AMT.

2.4.1 Worker Level

We used some of the worker filters provided by Amazon Mechanical Turk. At first, we considered submitting different runs filtering by country, but given the tight deadlines we discarded this option in favor of having more workers. We did require all them to have a minimum of 100 total approved HITs, and a minimum percentage of approved HITs of 95% for the uc3m.graded and uc3m.slider runs, 98% for the uc3m.hterms run. Based on the demographics of AMT [Ipeirotis, 2010][Mason et al., 2011], we implemented these restrictions to avoid sporadic workers. In addition, we limited workers to contribute a maximum of 50 sets (250 judgments) to avoid super-workers. For the uc3m.hterms run we also requested workers to have Amazon's Categorization Master Qualification, but these workers were not attracted at all by our task, so we immediately removed this filter and started over. However, there is no easy way of knowing if any of the workers in the three runs does have this qualification anyway.

2.4.2 Implicit Task Level: Work Time

Our HIT templates measured how much time workers spent on each document, rather than on all five altogether, summing the time spent with both display modes. As an implicit quality measure we counted the number of time failures, that is, the number of documents in the set for which workers spent less than 4.5 seconds. Previous trials with real AMT workers and colleagues indicated 4.5 seconds as a suitable threshold, even for documents that can be quickly identifiable as spam.

We did not use the work time in seconds reported by Amazon. In our experience, many workers like to preview HITs, and spend some time getting used to the particular contents of each one. Some workers solve HITs while previewing, and the time they spend after accepting the HIT is used just to select their answers and submit. As such, using only the work time can be very misleading, and some workers get very offended if their work is rejected on the basis of work time alone [Urbano et al., 2010]. Our HITs were implemented as external, that is, they were hosted in our own server. Thus, we knew when workers previewed HITs. We included a script in the HIT template to report back to our server, every 10 seconds, how much time was spent on each document so far. When computing the number of time failures, we added this preview time to the work time to have a better estimate of actual time.

2.4.3 Explicit Task Level: Reading Comprehension

Our first approach for explicit quality measures was to modify documents by inserting a paragraph with syntactically correct yet nonsensical sentences, asking workers to detect where they are. However, some documents can be very easily identifiable as relevant or nonrelevant, so asking workers to spot the nonsensical paragraph would just be a waste of time and it would deteriorate worker performance (for example in a large Wikipedia article). In addition, automatically finding a suitable location to embed these paragraphs is certainly not an easy task. Kittur et al. [2008] asked for the number of references, sections and images in Wikipedia articles, but in our case documents are very heterogeneous HTML documents that seldom have a common structure, so we discarded these questions as well.

Our choice was to implement a very simple reading comprehension test where workers had to tell which one of two sets of keywords better described the document (call this the positive keyword set). We designed a previous experiment where subjects were given documents and they had to provide a list with 5 to 10 keywords that they thought best described the document. We ran four trials with real AMT workers: one with no qualification required, with more than 95% approval, with more than 1,000 total HITs approved and 95% approval, and another one with the Categorization Master Qualification required. Another trial was run with four colleagues with varying expertise in Information Retrieval. We found that virtually all subjects in all five trials provided the most and/or the second most frequent terms in the document (excluding stopwords and considering two terms as equal if they had the same stem), and those who did not were clearly spammers.

Given that all subjects recognized the most frequent terms, we included in the positive keyword set the 3 most frequent terms plus 2 random terms from the next 5 most frequent ones; and for the negative keyword set we randomly picked 5 terms out of the 25 least frequent ones (see Figure 1, right). In addition, terms within keyword sets were shuffled. We decided to pick terms randomly to some degree because many document sets had duplicate documents or documents for which the most frequent terms were the same, which would have made the task very easy. As an explicit quality measure we counted the number of keyword failures, that is, the number of documents in the set for which workers did not select the positive keyword set.

2.4.4 Rejecting HITs and Blocking Workers

We used the number of time and keyword failures to decide whether to accept all five judgments in the HIT, and in case of rejecting it whether to block the worker or not. If a HIT was to be rejected, we extended it so that another worker could answer it, and when blocking a worker we did not reject all his previous work to be sure we met the deadline. Moreover, some documents might not have clearly important terms, so simply failing the keywords question would not be that rare.

Action	Failure	uc3m.graded	uc3m.slider	uc3m.hterms
Reject HIT	Keyword	1	0	1
	Time	2	1	1
Block Worker	Keyword	1	1	1
	Time	2	1	1
	Total HITs rejected	3 (1%)	103 (19%)	13 (3%)
	Total Workers blocked	0 (0%)	38 (44%)	4 (12%)

Table 2. Maximum number of keyword and time failures allowed per HIT and worker.

As Table 2 shows, the uc3m.graded run was the most permissive of all three, and uc3m.slider was the most restrictive. For instance, in the first case we rejected a HIT if there were more than 1 keyword failures or more than 2 time failures, and if a worker accumulated more than 1 HIT rejected because of keyword failures or more than 2 HITs rejected because of time failures, we would block him. One of the consequences was that the uc3m.slider run took much more time to complete, as much more work was being rejected. In fact, workers seemed to keep failing in 7 of the HITs from the uc3m.slider run, so we decided to stop extending those HITs when three workers failed in them. The labels for those HITs were computed by simply averaging the labels of their three workers.

3 Task II: Aggregating Multiple Judgments

The second task of the Track promoted research on quality control at the fourth level: Aggregation. These mechanisms modify the task assignment process so that questions are answered by multiple workers, resulting in redundant answers from which a consensus response can be computed [Snow et al., 2008]. There are simple mechanisms such as using the vote of the majority [Alonso et al., 2008] or comparing the distributions of possible answers [Urbano et al., 2010]; and more complex alternatives such as weighting answers by the worker trustworthiness [Le et al., 2010] or using statistical models that try to maximize the effectiveness of workers [Sheng et al., 2008][Snow et al., 2008][Ipeirotis et al., 2010]. We followed the last line.

3.1 uc3m.wordnet

For this submission we used the GetAnotherLabel algorithm[3] (GAL) to manage redundant answers [Ipeirotis et al., 2010], based on the expectation maximization algorithm proposed by Dawid and Skene [1979]. Given a set of examples with previously known answers and the labels given by workers to those and other examples, it can compute a certain probability for each unknown example to have a particular label based on the worker responses. In addition, it computes a confusion matrix and an overall expected quality for each worker. In early trials we found this algorithm to perform exceptionally well, but we did not make a submission just with it alone because it is not our work.

Instead, we decided to explore stepwise executions of the algorithm using various features. Let us assume a feature that can be used to categorize the topics. Our idea is that workers could be more reliable in documents from one category than another, so we ran the algorithm once per category, using only documents pertaining to it. We then compared the worker expected quality in each category with the overall expected quality using all documents, and if it was smaller we ignored that worker's judgments and executed the algorithm again. That is, we used only the best workers per category, and ran the GAL algorithm with them. The features we tried were:

- Topic category, as indicated in TREC's topic descriptions: closed or limited, advice, navigational, etc.
- Subject of the information need: politics, shopping, people, geography, etc.
- Rareness of the topic. We looked up in Wordnet the terms in the topic descriptions, and if a topic had terms that did not appear in any glossary we categorized it as rare. That is, we assumed that some workers might have more difficulty with rare topics.

The binary labels are directly provided by the algorithm as the probability of having the relevant label, and the ranking labels can be assigned just by ordering documents according to these probabilistic binary labels. We tried the three features, and the topic rareness was marginally better than the others, so that is the one we used for our submission. However, when computing the labels based on the answers of the good workers, we only used their labels for that topic category, rather than their labels for all topic categories. Therefore, we lost much information for computing the consensus labels, which surely affected the quality of the results (see Section 4.2).

3.2 uc3m.rule

For the second run we trained a rule-based model using several features regarding worker quality, with each topic-document as a different example. Besides other features, we used the output of the GAL algorithm using all topic-document pairs; in particular, the confusion matrix for each worker. These matrixes are defined with several variables of the form π_{ij}^{w}, that is, the probability that worker *w* assigned the label *j* to an example whose actual label is *i*. The complete set of features is as follows:

- Relevant to nonrelevant ratio of the labels received for the topic-document. Using only this feature would be like using majority voting, possibly with bias correction.
- For all workers that labeled the topic-document, the average correct to incorrect ratio when they assigned a relevant label.
- For all workers that labeled the topic-document, the average correct to incorrect ratio when they assigned a nonrelevant label.
- For all workers that labeled the example, the average posterior probability of it being relevant, that is, π_{11}^{w} if they said relevant (they were right) and π_{10}^{w} if they said nonrelevant (they were wrong).
- For all workers that labeled the example, the average posterior probability of it being nonrelevant, that is, π_{00}^{w} if they said nonrelevant (they were right) and π_{01}^{w} if they said relevant (they were wrong).

The model provided the binary labels out of the box, and the ranking labels were computed by the branch of the decision tree the topic-document fell under: the larger the positive to negative ratio of examples falling under that branch, the higher the ranking.

3.3 uc3m.svm

For the third run we trained a SVM model using the same features as in the uc3m.rule run. The model allowed us to easily rank documents and assign binary labels: positive scores indicated relevant documents (negative scores for nonrelevant), and the larger the score the more relevant the document.

[3] Actually, we implemented a C# port of the original Java software, available at http://code.google.com/p/get-another-label-dotnet/

4 Results

A much debated aspect of the Track was how to actually evaluate runs. This year, the Track followed a different methodology than usual, because NIST assessors did not provide new relevance judgments to evaluate runs. The alternative consisted in computing the labels for each topic-document example through the vote of the majority, and using those labels as ground truth. One problem with this method is that the vote of the majority is too simple and it results in low quality labels when compared to other methods [Sheng et al., 2008]. Also, it is not clear to us whether the best results will be achieved by the actually best runs or by the average ones. In our experience, workers are biased toward the relevant label, probably because of the topic terms appearing in the documents. This can be observed when comparing UWaterlooMDS's run with the consensus labels for recall and specificity[4] [Lease et al., 2011]. As a result, if teams do not correct this bias somehow the vote of the majority will be biased too. Indeed, it is striking how much the numbers change when looking at the results against the vote of the majority and against the known NIST labels. All results we describe below are computed as per the available NIST gold, not the consensus gold. Also, we show our results only for the accepted work, ignoring rejected labels.

4.1 Task I

Figure 5 shows the expected pattern for different probabilistic labels for the *fair* labels in run uc3m.graded: the larger the value, the higher the recall and the lower the specificity. All measures vary between 0.6 and 0.8, but we decided to maximize recall and accuracy and let all *fair* judgments have a label of 1 for simplicity.

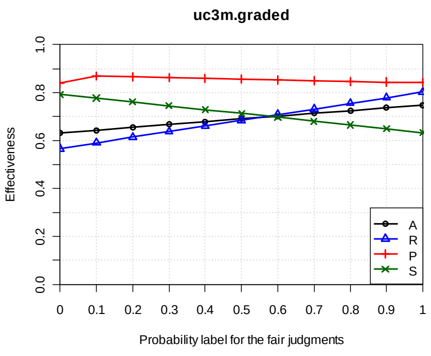


Figure 5. Accuracy, Recall, Precision and Specificity of the uc3m.graded run for various probabilistic label assignments to the *fair* judgments.

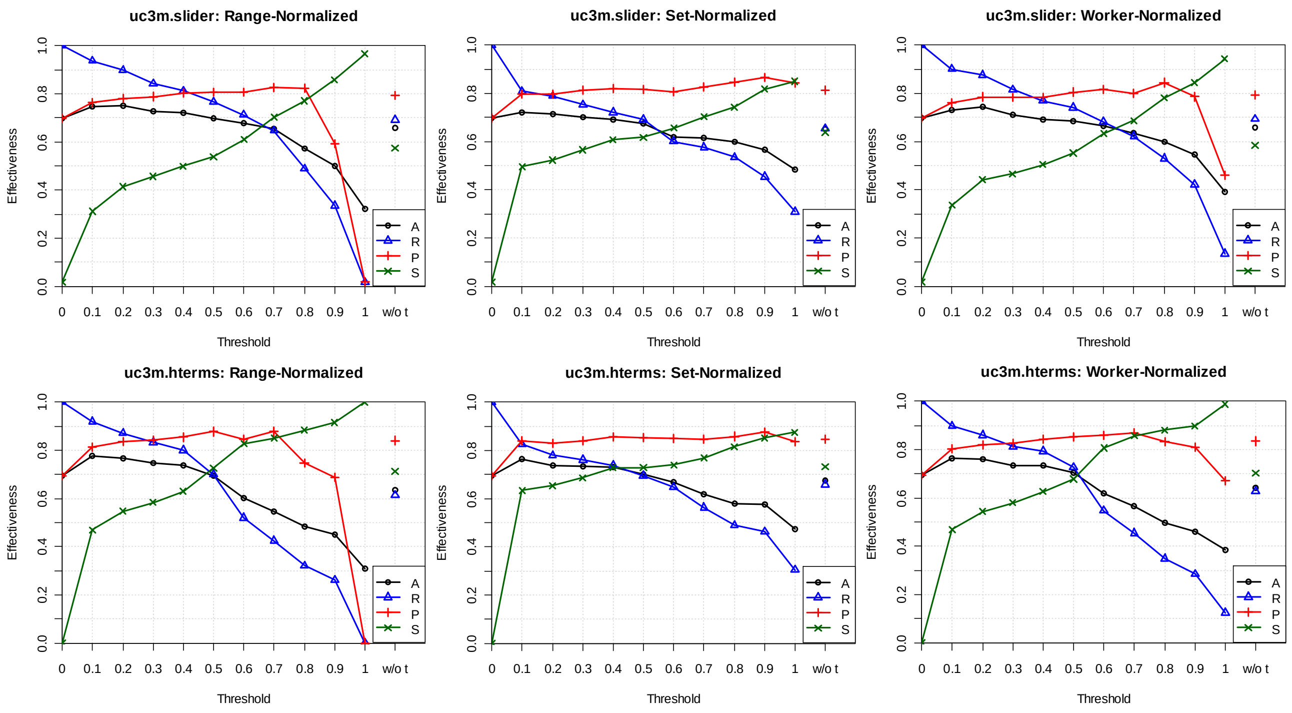


Figure 6. Accuracy, Recall, Precision and Specificity of the uc3m.slider (top) and uc3m.hterms (bottom) runs for the different methods to compute binary labels: Range-Normalized (left), Set-Normalized (center) and Worker-Normalized (right). The points in the right side of each plot mark the scores without using a threshold.

[4] Their run consisted in the judgments of one of the participants himself, which could somehow be considered as a NIST assessor judgments.

In the case of the uc3m.slider and uc3m.hterms runs the patterns are more complex (see Figure 6). The expected result is again clear: the larger the threshold *t*, the lower the recall and the higher the specificity. As the sliders need to be more and more to the right for the document to be considered relevant (high values of *t*), the fewer relevant documents we will have altogether, missing actually relevant documents (low recall) and hitting actually nonrelevant ones (high specificity). Indeed, in the Range-Normalized plot (left), for *t*=0 recall is 1 and specificity is 0, reversed for *t*=1. The steep decay in precision is caused by a couple of documents with the sliders set at the very end, but that turned out to be nonrelevant.

In general, without thresholds involved, precision scores are a little over 0.8; and accuracy, recall and specificity move between 0.6 and 0.7. In addition, we can see that using thresholds can indeed improve the results over the base cases. In general, thresholds between 0.5 and 0.6 seem to work best, improving both recall and accuracy at the cost of lowering specificity. Another clear pattern can be observed with the slopes of the curves: the Set-Normalized curves have less variation, ensuring better results around the chosen threshold. We chose the Set-Normalized method with Threshold *t*=0.4, because it yielded the seemingly best results for uc3m.hterms and very good results for uc3m.slider without approaching steep decays and maximizing more measures. The cluster method, not shown here, resulted in worse results overall.

Table 3 summarizes the results of our runs compared to the best and median per measure. The median scores are computed between the other teams runs (according to the notebook overview paper [Lease et al., 2011]) and our main run (uc3m.hterms). The top scores are computed over all runs. In addition, we do not include the BUPT-WILDCAT team results because "workers could review the reference answers of the gold set as instructions for further HITs", yielding their exceptionally high overfitted scores (above 0.9) [Lease et al., 2011].

	Accuracy	Recall	Precision	Specificity	MAP	NDCG
Best	.75	.80	.86	.73	-	-
Median	.62	.74	.77	.54	-	-
uc3m.graded	**.75**	**.80**	.84	.63	.73	.78
uc3m.slider	.69	.72	.82	.61	.76	.80
uc3m.hterms	.73	.74	**.86**	**.73**	**.81**	**.84**

Table 3. Results of our three runs for Task I compared to the best and median per measure.

As the table shows, our runs performed better than the median in all cases, except for uc3m.slider in recall. In fact, the best overall results were achieved by our uc3m.graded and uc3m.hterms runs. In addition, we can see that the slider widget did indeed result in a better ranking of the documents in runs uc3m.slider and uc3m.hterms. We note that we *did not* use these known labels to train workers or as trap questions. However, we did use them to tune the threshold values. Nonetheless, the results ignoring the thresholds, and thus the known labels altogether, would have still been above the median for all measures and at the top most of the times. We thus discard overfitting effects.

4.2 Task II

Table 4 summarizes the results of our runs compared to the best and median per measure. Again, the median scores are computed between the other teams runs (according to the notebook overview paper) and our main run (uc3m.rule). The top scores are computed over all runs.

	Accuracy	Recall	Precision	Specificity	MAP	NDCG
Best	.71	.91	.70	.68	.17	.42
Median	.65	.77	.63	.57	.11	.36
uc3m.rule	0.70	0.75	0.68	0.64	**0.17**	**0.42**
uc3m.svm	**0.71**	0.75	**0.70**	**0.68**	0.08	0.33
uc3m.wordnet	0.57	0.66	0.56	0.48	0.06	0.30

Table 4. Results of our three runs for Task II compared to the best and median per measure.

We can see how the uc3m.wordnet run did indeed perform poorly, probably because of our mistake when using only within-category judgments to compute the labels (see Section 3.1). The other two runs performed above the median except for recall and the uc3m.svm run for the ranking measures. In fact, the best overall scores, except for recall, were achieved by our runs. We again discard any overfitting effects because these true labels were not known to participants.

5 Conclusions

This paper described the participation of the University Carlos III of Madrid (uc3m team) in the TREC 2011 Crowdsourcing Track. For the first task we submitted three runs that used Amazon Mechanical Turk: in one case we asked workers for typical 3-point judgments, and in the other two cases we asked for unconventional judgments based on a slider widget, thus focusing on the ranking of documents. We also worked on a simple task design, document rendering, and quality control based on a simple reading comprehension test. In general, our runs ranked at the top for most measures. In any case, all teams performed reasonably well: the median scores are about 0.75 for precision and recall. These numbers agree exceptionally well with Ellen Voorhees finding that NIST assessors agreed with each other to about 0.65 precision at 0.65 recall in the old TREC ad hoc tasks [Voorhees, 2000]. The results of the Crowdsourcing Track do therefore support the use of evaluation methodologies based on crowdsourcing.

For the second task we also submitted three runs: two of them were based on trained machine learning models, and the other one consisted in a stepwise execution of an expectation maximization algorithm. Our hypothesis was that bad workers are not necessarily bad for all kinds of topics. For instance, a worker could perform poorly with sports-related topics, but good with politics. However, we could not really study this hypothesis because of a programming bug. Nonetheless, the other two runs ranked at the top of the results for all measures but recall. Again, the average results across teams passed the 0.65 precision at 0.65 recall threshold.

A clear line for further work is the quality control mechanism at the explicit task level for arbitrary Web documents. Our results with and without rejected work do not differ significantly, so we tend to believe that our reading comprehension test was probably too easy. Two immediate changes are the maximum number of allowed mistakes and the frequency of the terms included in each set: perhaps having the 3 most frequent terms is too obvious, or maybe having five documents together for the same topic makes the test inherently easier.

Acknowledgements

We acknowledge the support received by Amazon for the accomplishment of these experiments.

References

O. Alonso, D.E. Rose, and B. Stewart, "Crowdsourcing for Relevance Evaluation", *ACM SIGIR Forum*, vol. 42, no. 2, pp. 9-15, 2008.

A.P. Dawid and A.M. Skene, "Maximum Likelihood Estimation of Observer Error-Rates Using the EM Algorithm", *Journal of the Royal Statistical Society*, vol. 28, no. 1, pp. 20-28, 1979.

P.G. Ipeirotis, F. Provost, and J. Wang, "Quality Management on Amazon Mechanical Turk", *ACM SIGKDD Workshop on Human Computation*, pp. 64-67, 2010.

P.G. Ipeirotis, "The Demographics of Mechanical Turk", 2010, available at: http://archive.nyu.edu/handle/2451/29585.

Kittur, E.H. Chi, and B. Suh, "Crowdsourcing User Studies With Mechanical Turk", *Annual ACM SIGCHI Conference on Human Factors in Computing Systems*, pp. 453-456, 2008.

J. Le, A. Edmonds, V. Hester, and L. Biewald, "Ensuring Quality in Crowdsourced Search Relevance Evaluation: The Effects of Training Question Distribution", *ACM SIGIR Workshop on Crowdsourcing for Search Evaluation*, pp. 17-20, 2010.

M. Lease and G. Kazai, "Overview of the TREC 2011 Crowdsourcing Track (Conference Notebook)", *Text REtrieval Conference Notebook*, 2011.

W. Mason and D.J. Watts, "Financial Incentives and the "Performance of Crowds"", *ACM SIGKDD Workshop on Human Computation*, pp. 77-85, 2009.

W. Mason and S. Suri, "Conducting Behavioral Research on Amazon's Mechanical Turk", *Social Science Research Network*, 2011.

J. Rzeszotarski and A. Kittur, "Instrumenting the Crowd: Using Implicit Behavioral Measures to Predict Task Performance", *ACM Symposium on User Interface Software and Technology*, 2011.

M. Sanderson, M.L. Paramita, P. Clough, and E. Kanoulas, "Do User Preferences and Evaluation Measures Line Up?", *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 555-562, 2010.

A.D. Shaw, J.J. Horton, and D.L. Chen, "Designing Incentives for Inexpert Human Raters", *ACM Conference on Computer Supported Cooperative Work*, pp. 275-284, 2011.

V.S. Sheng, F. Provost, and P.G. Ipeirotis, "Get Another Label? Improving Data Quality and Data Mining Using Multiple, Noisy Labelers", *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 614-622, 2008.

R. Snow, B. O'Connor, D. Jurafsky, and A.Y. Ng, "Cheap and Fast—But is it Good? Evaluating Non-Expert Annotations for Natural Language Tasks", *Conference on Empirical Methods in Natural Language Processing*, pp. 254-263, 2008.

J. Urbano, J. Morato, M. Marrero, and D. Martín, "Crowdsourcing Preference Judgments for Evaluation of Music Similarity Tasks", *ACM SIGIR Workshop on Crowdsourcing for Search Evaluation*, pp. 9-16, 2010.

E.M. Voorhees, "Variations in Relevance Judgments and the Measurement of Retrieval Effectiveness", *Information Processing and Management*, vol. 36, no. 5, pp. 697-716, 2000.

D. Zhu and B. Carterette, "An Analysis of Assessor Behavior in Crowdsourced Preference Judgments", *ACM SIGIR Workshop on Crowdsourcing for Search Evaluation*, pp. 21-26, 2010.