University of Glasgow at TREC 2011: Experiments with Terrier in Crowdsourcing, Microblog, and Web Tracks

Richard McCreadie, Craig Macdonald, Rodrygo L. T. Santos and Iadh Ounis
{richardm,craigm,rodrygo,ounis}@dcs.gla.ac.uk

School of Computing Science
University of Glasgow
Glasgow, UK

ABSTRACT

In TREC 2011, we focus on tackling the new challenges proposed by the pilot Crowdsourcing and Microblog tracks, using our Terrier Information Retrieval Platform. Meanwhile, we continue to build upon our novel xQuAD framework and data-driven ranking approaches within Terrier to achieve effective and efficient ranking for the TREC Web track. In particular, the aim of our Microblog track participation is the development of a learning to rank approach for filtering within a tweet ranking environment, where tweets are ranked in reverse chronological order. In the Crowdsourcing track, we work to achieve a closer integration between the crowdsourcing marketplaces that are used for relevance assessment, and Terrier, which produces the document rankings to be assessed. Moreover, we focus on generating relevance assessments quickly and at a minimal cost. For the Web track, we enhance the data-driven learning support within Terrier by proposing a novel framework for the fast computation of document features for learning to rank.

1. INTRODUCTION

In TREC 2011, we participate in the Web adhoc and diversity tasks, the Microblog real-time search task, and the Crowdsourcing assessment and consensus tasks. Our focus is the improvement of the support for data-driven ranking models within the Terrier Information Retrieval (IR) platform [19], the effective application of these models for new tasks, e.g. microblog search, and the development of new features for these tasks.

In the assessment task of the Crowdsourcing track [13], we propose a close integration of a crowdsourcing infrastructure into our Terrier IR platform, for achieving fast generation of relevance assessments based upon Amazon's Mechanical Turk (MTurk)[1]. For the consensus task, we propose and evaluate a machine learning approach, which leverages prior worker accuracy, topic difficulty and worker impact for calculating consensus between multiple assessors.

The major goal of our participation within the Microblog track [20] is to determine whether filtering is an effective approach to improve the relevance and quality of a tweet ranking. In particular, we propose a learning to rank approach that uses multiple features extracted from each tweet to determine if that tweet should be filtered from the ranking.

In our participation in the Web track [8], our primary goal is to improve the underlying learning infrastructure within Terrier, such that it becomes easier to generate and evaluate many data-driven models in a short timeframe. In particular, in the adhoc task, we propose a framework for the fast computation of multiple query-dependent features for learning to rank. In the diversity task, we leverage learned models within our state-of-the-art xQuAD framework [24] for search result diversification, so as to learn both the relevance of a document to a query and its coverage of the multiple aspects underlying this query.

The remainder of this paper is structured as follows. In Sections 2 and 3, we describe our participation in the Crowdsourcing track assessment and consensus tasks, respectively. Section 4 details our participation in the new Microblog track real-time search task. In Sections 5 and 6, we describe our Web track adhoc task and Web track diversity task participations, respectively. Conclusions are provided in Section 7.

2. CROWDSOURCING TRACK: ASSESSMENT TASK

The aim of the assessment task of the Crowdsourcing track is to develop effective approaches to generate accurate relevance assessments for Web documents in a fast and cheap manner [13]. For our participation in the assessment task, we integrate a crowdsourcing infrastructure into our Terrier IR platform, to achieve the fast generation of relevance assessments based upon Amazon's Mechanical Turk (MTurk). The aim of integrating this crowdsourcing infrastructure is to decrease the time, effort and expertise required to crowdsource relevance assessments for novel tasks and new collections. In particular, through a closer integration to the Terrier platform that performs the searches being evaluated, aspects of the crowdsourcing, e.g. breaking down searches into crowdsourcable jobs, can be automated.

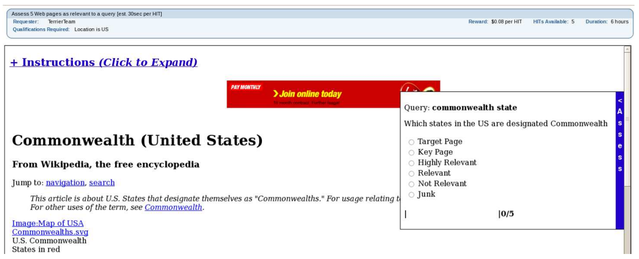
Our proposed integrated infrastructure extends Terrier with two types of functionalities facilitating fast and easy crowdsourcing of relevance assessments. Firstly, the infrastructure facilitates automatic generation of one or more relevance assessments for a Web document. In particular, it takes lists of ranked results returned by Terrier and automatically constructs a series of assessment tasks for the Web documents within the ranked results. Each assessment task is expressed as MTurk Human Intelligence Task (HIT). Each HIT is operationalised as a MTurk ExternalQuestion [3] that redirects workers to a modified instance of the Terrier Web interface[2]. This Web interface is hosted on our local ma-

[1]`https://www.mturk.com`

[2]`http://terrier.org/docs/v3.5/terrier_http.html`

		Classification Metrics				Error Metrics		
Ground Truth	Run	Accuracy	Recall	Precision	Specificity	Log-Loss	KL-Divergence	RMSE
Consensus	TREC Median	74.0%	75.4%	79.1%	**70.4%**	954.6	2058.2	63.8%
	uogTrP1rg	**78.2%**	**86.6%**	**80.4%**	64.0%	**822.5**	**612.8**	**54.4%**
TREC Assessors	TREC Median	**67.5%**	73.1%	**78.1%**	**52.5%**	**103.0**	**103.1**	**50.6%**
	uogTrP1rg	61.7%	**73.8%**	67.7%	33.0%	125.9	126.0	50.9%

Table 1: Performance of uogTrP1rg compared to the two TREC median ground truth assessment sets within the assessment task of the Crowdsourcing track. The best run is highlighted for each ground truth and measure.


Assess 5 Web pages as relevant to a query [est. 30sec per HIT]
Requester: TerrierTeam
Reward: $0.08 per HIT HITs Available: 5 Duration: 6 hours
Qualifications Required: Location is US

+ Instructions *(Click to Expand)*

PAY MONTHLY ›Join online today

Commonwealth (United States)

From Wikipedia, the free encyclopedia

Jump to: navigation, search

This article is about U.S. States that designate themselves as "Commonwealths." For usage relating t
For other uses of the term, see Commonwealth.

Image:Map of USA
Commonwealths.svg
U.S. Commonwealth
States in red

Query: **commonwealth state**
Which states in the US are designated Commonwealth
- Target Page
- Key Page
- Highly Relevant
- Relevant
- Not Relevant
- Junk

|0/5

Figure 1: Example of the Web interface used during the assessment task of the Crowdsourcing track.

chines and is used by the workers to assess each document. With regard to the Web interface design, we focus on achieving fast turn-around times, with only 1-click needed to assess each document. We also use pre-rendered Web pages in conjunction with a 'floating' assessment panel, which is always visible in the assessment interface, hence minimising the worker's scrolling effort. The interface used is shown in Figure 1.

Secondly, the proposed infrastructure provides automatic and supervised validation for quality assurance on the crowdsourced assessments produced. In particular, our validation is a two-stage process. In the first, the time spent on the task was used to automatically identify poorly performing workers [15]. During the second stage, we perform a variant of gold judgement validation [12]. From the crowdsourced assessments, a 10% subset is selected that maximises the coverage of the workers participating in the assessment task, while also accounting for the number of assessments that each worker has made, i.e. more assessments from a prolific worker will be validated than from a worker that judged only a single document. The selected assessments are manually assessed by the crowdsourcer. The resultant validation assessments are used to identify workers that are not completing the assessment task in good faith, e.g. workers that are just randomly clicking.

We submitted one run, namely uogTrP1rg, which uses our integrated crowdsourcing infrastructure to generate relevance assessments using Amazon's Mechanical Turk. Assessments were generated for the test topics over the course of a three-day period. Notably, in contrast to standard crowdsouring practices [15], only one assessment per document was collected. This assured that the crowdsourcing costs incurred were kept very low. Indeed, we payed about US$39 for our participation. As per the TREC submission guidelines [13], assessments marked for rejection from our validation process were also included in the run (this will degrade run performance). Runs were evaluated in comparison to two ground truths, namely: TREC assessors; and the majority vote of all submitted runs to the Crowdsourcing track (Consensus). Table 1 reports the performance of uogTrP1rg compared to the TREC median. Performance is reported in terms of classification measures (higher is better) and error measures (lower is better).

From Table 1, we observe that uogTrP1rg marginally underperformed the TREC median, except under Recall, in comparison to the TREC assessor ground truth. However, it markedly outperformed the TREC median under all measures but Specificity on the consensus ground truth. This shows that our approach generated assessments of similar quality to that of other participating systems. Overall, bearing in mind the limited resources allocated, i.e. only US$39, we believe that achieving such a reasonable performance validates the effectiveness of our proposed infrastructure for automatic crowdsourcing of relevance assessments. Moreover, if the assessments that were marked for rejection during validation but included in the run were removed, it is expected that performance will be markedly increased.

3. CROWDSOURCING TRACK: CONSENSUS TASK

The aim of the consensus task of the Crowdsourcing track is to investigate techniques for aggregating multiple assessments of varying quality for a single document into a single high-quality assessment for that document, i.e. calculate consensus among multiple assessors [13]. Participants calculate consensus across five crowdsourced assessments for Web documents. For our participation in the consensus task, we propose a data-driven approach that learns a model for consensus calculation. Our approach trains a model comprised of three features, namely: prior worker accuracy – how well the worker performed on previous assessments; topic difficulty – how much disagreement there is between workers for the topic; and worker impact – what proportion of the assessment task has the worker attempted. We linearly combine these features extracted for an assessment to score that assessment as shown below:

$$Score(a, w, d) =$$
$$\left|(a - 0.5) + \frac{N}{2}\left(\alpha \cdot Accuracy(w) + \beta \cdot \frac{HITsDone(w)}{TotalHITs} + \gamma \cdot Agreement(d)\right)\right|$$

where a is the assessment, w is the worker, d is the document being assessed, $Accuracy(w)$ is the prior accuracy for worker w, $HITsDone(w)$ is the number of HITs completed by worker w, $TotalHITs$ is the number of HITs that were available to the worker and $Agreement(t)$ is the level of Kappa Fleiss [11] agreement between all workers that attempted document d. N is a normalising factor to ensure that the numerator remains within the range [0-1]. α, β and γ are weights for each feature. We learn these weights using a line-search optimisation [6]. The objective function minimises Root Mean Squared Error (RMSE) on the development topic set provided for the task [13]. Each new assessment is weighted via the linear combination of feature scores. Weighted assessments for a single document are summed together to provide a final assessment value for that document as shown below:

$$Assessment(A, W, d) = \frac{1}{|A|}\sum_{i=1}^{|A|} Score(A_i, W_i, d) \qquad (1)$$

where d is the document being assessed, A is the set of assessments for document d, W is the set of workers that made the assessments in A and $|A|$ is the size of A, i.e. the number of assessors.

For the consensus task, we submitted one run, uogTrP2O4wtr that uses our learned model to calculate consensus judgements. Table 2 reports the performance of our uogTrP2O4wtr run at consensus calculation against a ground truth by TREC assessors. Performance is measured in terms of classification based measures (higher is better) and error based measures (lower is better). From Table 2, we observe that under classification metrics, uogTrP2O4wtr achieved less than the TREC median performance. However, under two of the three error based metrics, i.e. Log-Loss and KL-Divergence, uogTrP2O4wtr improved over the TREC median, i.e. a lower overall error was observed. This indicates that our learned model is good at expressing uncertainty, such as when we have no prior evidence about the workers making the assessments. However, as a consequence, assessments from good workers that we have never seen before receive little weight. Overall, our approach generates useful estimates of the performance of a worker when prior evidence exists. However, are currently further developing this approach to better account for cases where little is known about a worker.

4. MICROBLOG TRACK: REAL-TIME SEARCH TASK

The aim of the TREC Microblog track real-time search task is to investigate approaches to retrieve relevant tweets from before a point in time, when tweets are ranked in reverse-chronological order [20]. Participants rank tweets from the Tweets11 Twitter corpus for a set of query/timestamp pairs, referred to as topics. For our participation, we investigate a learning to rank approach, with the aim of learning a model that identifies tweets that can be discarded from a reverse-chronologically ordered ranking, hence improving relevance in the top ranks. In particular, this approach learns the features of a tweet that indicate when the tweet should be discarded, e.g. when it has poor relevance to the query or is written in a non-English language.

The Tweets11 corpus is unusual, in that it is not pre-provided by TREC. Instead, a set of approximately 16 million tweet identifiers are provided along with a tool for downloading the tweets for those tweet identifiers [20]. The corpus is dynamic, i.e. the number of available tweets changes over time, as users delete their own tweets, or spam accounts are removed by Twitter itself. There are two alternate methods via which the tweet downloading tool can collect tweets, namely: JSON – where full details of the tweet are downloaded; or HTML – where instead the HTML page for each tweet is scraped for content. We developed a customised version of the tool that uses the HTML method to download the corpus. Statistics of our version of Tweets11 are provided in Table 3. Using the Terrier IR platform [19] in conjunction with a new Twitter collection class to enable the parsing of the corpus[3], we indexed the Tweets11, removing stopwords and stemming each tweet using the English Porter Stemmer.

Data	Quality	Value
Tweets11 Corpus	Time Range	23/01/11 → 08/02/11
	Days	16
	# Tweets	15,663,909*
	Avg. Tweets Per Day	978,994*
	Unique URLs	2,274,350*
	Unique Twitterers	5,218,687*

Table 3: Tweets11 corpus statistics. Values marked with * are relative due to the dynamic nature of the Tweets11 corpus.

Before applying our proposed learning to rank approach, we first create a time-ordered ranking of tweets from Tweets11 for each topic. In particular, for a topic – comprised of a query/timestamp pair – we retrieve a ranking of tweets containing one or more query terms in reverse-chronological order, which were posted before the timestamp. This ranking is referred to as the *sample*. The sample has high recall but low precision, i.e. most of the relevant tweets are within the ranking, but due to the reverse chronological order these are unlikely to occur in the top ranks. We then reduce the size of the sample, by removing any tweets not containing one or more hashtags (as we will show later, this proved to be a mistake).

Next, we filter the sample using our new learning to rank approach. In particular, we define six feature sets, each describing one aspect of the tweets in the sample. Table 4 describes these feature sets and lists the number of individual features within each. Notably, the last two feature sets – *Referred Page* and *Twitterer* – were extracted from external corpora in a timely fashion, i.e. a crawled corpus of documents linked from Tweets11 and from the Twitter Gardenhose stream, respectively.

Using the Automatic Feature Selection [17] learning to rank technique on a separate topic set comprised of 55 topics crowdsourced using Amazon's Mechanical Turk, we trained models for estimating whether tweets should be filtered out. In particular, we train one model using the 66 features from the first four feature sets in Table 4, and two models using the 76 features from all feature sets. The objective function used was Normalised Discounted Accumulative Gain at rank 30 (NDCG@30).

Furthermore, during the training process, a time-decay function can be applied to the score of each tweet to create a model that promotes more recent tweets. In particular, the time-decay function discounts the score for each tweet as follows:

$$decay(score, age) = score \cdot \frac{1}{\sqrt{18 \cdot \pi}} \cdot e^{-\frac{age-15}{9}} + 0.3 \qquad (2)$$

where *score* is the score for a tweet as defined by one of our learned models and *age* is the age of that tweet in comparison to the query time (in hours). This particular function was chosen to primarily promote tweets made during the 6

[3]http://ir.dcs.gla.ac.uk/wiki/Terrier/Tweets11

		Classification Metrics				Error Metrics		
Ground Truth	Run	Accuracy	Recall	Precision	Specificity	Log-Loss	KL-Divergence	RMSE
TREC Assessors	TREC Median	**61.7%**	**73.3%**	**59.5%**	**50.2%**	2047.6	2047.8	**54.8%**
	uogTrP2O4wtr	44.1%	34.5%	42.7%	53.7%	**931.7**	**931.7**	58.8%

Table 2: Performance of uogTrP2O4wtr at consensus calculation against TREC assessors within the consensus task of the Crowdsourcing track. The best run is highlighted for each measure.

Feature Set	Internal	Summary	Number
Tweet Relevance	✔	Features encapsulating the relevance of the tweet to the query, e.g. document retrieval model scores like BM25 [23].	38
Tweet Quality	✔	Features from the tweet that may be indicators of quality, e.g. URLs and tweet length.	7
Tweet Language	✔	Features generated describing the language of the tweet, e.g. the classification probability of being written in English.	17
Spam Detection	✔	Features from the tweet that may be indicate that it is spam, e.g. a high hashtag count [10].	4
Referred Page	✘	Features extracted describing Web pages referenced from each tweet, e.g. stopwords contained [4]	5
Twitterer	✘	Features about the user that made the tweet, e.g. number of followers and statuses	5
Total			76

Table 4: A summary of all tweet filtering features used in the Microblog track real-time search task.

		All Rel		High Rel	
Run	Submitted	MAP	P@30	MAP	P@30
TREC median	-	**0.1421**	0.2229	**0.1510**	**0.2343**
uogTrUB2	✔	0.1014	0.1939	0.0714	0.0818
uogTrLqea	✔	0.0625	0.1068	0.0402	0.0485
uogTrLqeabd	✔	0.0625	0.1068	0.0402	0.0485
uogTrLqeabdd	✔	0.0625	0.1068	0.0402	0.0485
uogTrUB2_NoFilter	✘	0.0427	0.0532	0.0473	0.0559
uogTrLqeabd_NoFilter	✘	0.1136	**0.2381**	0.0973	0.2262

Table 5: Results of our submitted and unsubmitted runs for the Microblog track under mean average precision (MAP) and precision at 30 (P@30) for both the High Rel and All Rel topic sets.

hour period prior to the query time.

We apply a learned model, either with our time-decay function or not, over 66 or 76 features on the tweet sample to create each run. We submitted the following four runs to the TREC Microblog track:

- uogTrUB2: The sample (with hashtag filtering) before applying the learned model, as a baseline.

- uogTrLea: uogTrUB2 filtered using a model learned from 66 internal (extracted from the Tweets11 corpus) features: *Tweet Relevance*, *Tweet Quality*, *Tweet Language* and *Spam Detection*. The time-decay function was not used during training.

- uogTrLqeabd: uogTrUB2 filtered using the same 66 internal features as uogTrLea, but adds the 10 external features from the *Referred Page* and *Twitterer* feature sets. The time-decay function was not used during training.

- uogTrLqeabdd: uogTrUB2 filtered using the same 76 features as uogTrLqeabd, but where the time-decay function was applied to create a model that promotes more recent tweets.

Table 5 reports our run performances in terms of the mean average precision (MAP) and precision at 30 (P@30) measures for the two official topic sets, namely: High Rel (only highly relevant tweets) and All Rel (all relevant tweets) [20]. We observe that our runs are sub-median in effectiveness and that our baseline run (uogTrUB2) – that only uses the sample – outperformed runs that added the learned filtering approach. Upon analysis we observed that the hashtag filtering applied to the tweet sample proved to be overly restrictive on the test topics, in that it filtered out many (actually the majority) of relevant tweets. Further filtering of this already over-filtered ranking by the learned model could only harm performance. This shows that many relevant tweets do not use hashtags at all, hence it is a mistake to use it as a filter.

In light of this, to test our learned filtering approach, we tested two additional unsubmitted runs, uogTrUB2_NoFilter and uogTrLqeabd_NoFilter that are also shown in Table 5. uogTrUB_NoFilter uses a sample without the hashtag-based reduction or any learned model filtering. Meanwhile, uogTrLqeabd_NoFilter uses the same learned model as our third official run (uogTrLqeabd), but does not filter the sample before applying the learned model. We see that the completely unfiltered run, uogTrUB2_NoFilter, is markedly worse than its filtered counterpart, i.e. uogTrUB2. But the additional learned run, uogTrLqeabd_NoFilter, improves over our initial baseline run and provides similar performance to the TREC median. Indeed this run improves over the TREC median under P@30 on the All Rel topic set. This result is promising, as it indicates that our learning to rank approach that filters a reverse-chronologically ordered ranking of tweets may be able to improve the quality and relevance of a tweet ranking. Indeed, with training data more representative of the topics, we expect that effectiveness will improve.

5. WEB TRACK: ADHOC TASK

In the adhoc task [8], our primary aim is to enhance our data-driven learning infrastructure that has proven effective during previous participations [16, 28], such that it becomes easier to generate and evaluate many learning to rank models in a short timeframe. To this end, we extend Terrier with a framework for the fast computation of document features for learning to rank for web search. This framework ensures that the postings of the documents most likely to make it into the final ranking are readily available for an on-the-fly computation of multiple query-dependent features, without requiring multiple passes over the posting lists in the inverted file. For instance, this permits learned

Features	Total
Weighting models (DPH [2], PL2 [2], BM25 [23], LM, MQT [27])	21
Fields-based models (PL2F [14])	1
URL and link analysis features (e.g. PageRank, Absorbing Model* [22], EdgeReciprocity*)	13
Quality features (e.g., fraction of stopwords, table text [5])	8
Click feature (click count)	1
Spam feature (Cormack's fusion score [9])	1
Term-dependence models (MRF [18], pBiL [21])	2
TOTAL	45* / 47

Table 6: Document features used in the Web track. Category A runs use all features, except for the two marked with a star (*). Category B runs use all features.

Run	Category	NDCG@20	ERR@20
TREC median		0.1876	0.1061
uogTrA45Nm	A	0.3043	0.1481
uogTrA45Vm	A	**0.3052**	**0.1485**
uogTrB47Vm	B	0.2278	0.1231

Table 7: Results of the three submitted runs for the Web track adhoc task under the normalised discounted cumulative gain at rank 20 (NDCG@20) and expected reciprocal rank at rank 20 (ERR@20) measures.

models to efficiently combine multiple standard weighting models (e.g. PL2 [1]), proximity models (e.g. Markov Random Fields [18]), and fields-based models (e.g. PL2F [14]). Moreover, the framework is compatible with dynamic pruning strategies such as MaxScore [31] and WAND [7] (which permit increased efficiency without loss of effectiveness), ensuring that learned models deploying many features can be efficiently and effectively applied.

We index the category A (∼500M English documents) and category B (∼50M English documents) subsets of the ClueWeb09 corpus without stemming or stopwords. At retrieval time, a weak Porter stemmer and the DPH [2] weighting model are used to identify 5000 documents to re-rank using the learned models. In particular, our category A and B runs use a total of 45 and 47 features, respectively, as described in Table 6. For learning, we employ the AFS algorithm [17], with the 98 TREC 2009 and 2010 queries used as training data. In particular, we consider two basic learning scenarios: with and without validation data. In the former scenario, the 98 queries are randomly split into training (60%) and validation (40%), so as to prevent overfitting; in the latter scenario, all queries are used for training and no validation is performed.

We submitted three runs to the adhoc task:

- uogTrA45Nm (category A) deploys ranking models learned using 45 features, without a validation step;
- uogTrA45Vm (category A) deploys ranking models learned using 45 features, with a validation step;
- uogTrB47Vm (category B) deploys ranking models learned using 47 features, with a validation step.

Table 7 reports the performance of the three submitted adhoc runs under the normalised discounted cumulative gain at rank 20 (NDCG@20) and expected reciprocal rank at rank 20 (ERR@20) measures. We observe that we are substantially above the TREC median for the adhoc ranking task. Indeed, run uogTrA45Vm outperforms the median by 66% for NDCG@20 and 39% for ERR@20. Moreover, it was the best submitted run by NDCG@20 among all participating groups in the adhoc task of the Web track, and second ranked by ERR@20 [8]. Furthermore, our category A run that used validation marginally improved over the category A run that did not use validation, indicating that the use of validation data might be useful in an adhoc setting. Finally, we note the higher performance of the category A runs compared to the category B run, contrasting with previous experiences in the 2009 and 2010 TREC Web track [26]. This suggests that our learned models for category A runs are better able to identify high quality documents within the larger category A corpus than in previous years. Overall, we find that our learned approach for Web search has been very effective for the 2011 Web track adhoc task.

6. WEB TRACK: DIVERSITY TASK

	Category	ERR-IA @20	α-NDCG @20	NRBP	task
TREC median		0.4079	0.5160	–	diversity
uogTrA45Nm	A	0.5069	0.6118	0.4642	adhoc
uogTrA45Nmx2	A	**0.5284**	0.6298	**0.4872**	diversity
uogTrA45Vm	A	0.5025	0.6103	0.4596	adhoc
uogTrA45Vmx	A	0.5238	**0.6304**	0.4799	diversity
uogTrB47Vm	B	0.4646	0.5611	0.4304	adhoc
uogTrB47Vmx	B	0.4674	0.5649	0.4312	diversity

Table 8: Results of the submitted runs to the diversity task of the Web track.

In the diversity task [8], we continue to improve our state-of-the-art xQuAD framework for search result diversification [24, 25, 27, 30]. In particular, xQuAD models an ambiguous query as an ensemble of the multiple possible information needs underlying this query, represented as different query aspects [29]. Given an initial ranking $\mathcal{R}$ for the query q, and a set of aspects $\mathcal{A}$ identified for this query, xQuAD iteratively builds a re-ranking $\mathcal{S}$ by selecting, at each iteration, a document $d^* \in \mathcal{R} \setminus \mathcal{S}$ such that:

$$d^* = \operatorname*{arg\,max}_{d \in \mathcal{R} \setminus \mathcal{S}} \, (1 - \lambda) \, \mathrm{P}(d|q) + \lambda \, \mathrm{P}(d, \bar{\mathcal{S}}|q), \tag{3}$$

where $\mathrm{P}(d|q)$ is the probability that a document d satisfies q and $\mathrm{P}(d, \bar{\mathcal{S}}|q)$ is the probability that d, but none of the documents in $\mathcal{S}$, selected in previous iterations, satisfies the multiple aspects of q. In practice, these two probabilities can be thought of as representing the *relevance* and the *diversity* of d, respectively, with the parameter λ controlling the trade-off between the two probabilities. Additionally, the probability $\mathrm{P}(d, \bar{\mathcal{S}}|q)$ can be further expanded as follows:

$$\mathrm{P}(d, \bar{\mathcal{S}}|q) = \sum_{a \in \mathcal{A}} \mathrm{P}(a|q)\,\mathrm{P}(d|q,a) \prod_{d_j \in S} \mathrm{P}(\bar{d_j}|q,a), \quad (4)$$

where the aspect $a \in \mathcal{A}$ is one of the possible aspects underlying the query q, $\mathrm{P}(a|q)$ conveys the importance of this aspect in light of q, $\mathrm{P}(d|q,a)$ estimates the coverage of d with respect to a, and $\prod \mathrm{P}(\bar{d_j}|q,a)$ estimates the novelty of any document satisfying a, given how badly this aspect is satisfied by the previously selected documents $d_j \in \mathcal{S}$.

In TREC 2011, we deploy learning-to-rank to produce refined estimations for two key components of xQuAD: the relevance of a search result to the initial query (i.e., $\mathrm{P}(d|q)$), and the coverage of this result with respect to each of the aspects identified for this query (i.e., $\mathrm{P}(d|q,a)$). In our participation, we exploit query reformulations from a commercial search engine in order to identify multiple query aspects [24]. For learning the relevance and coverage components of xQuAD, we leverage the same document features used for our adhoc runs described in Section 5. In particular, we estimate $\mathrm{P}(d|q,a)$ as:

$$\mathrm{P}(d|q,a) = \sum_i w_i f_i(d,q,a), \quad (5)$$

where f_i is one of the features from Table 6, and w_i is its learned weight, obtained using the AFS algorithm [17], as described in Section 5. As in the adhoc task, we also consider two basic learning scenarios: with and without validation data, based upon the same set of 98 queries.

We submitted three runs to the diversity task, all of which leverage machine learned models within xQuAD:

- uogTrA45Nmx2 (category A) deploys xQuAD to diversify the top 1000 results retrieved by our adhoc uogTrA45Nm run, with document coverage learned without validation;
- uogTrA45Vmx (category A) deploys xQuAD to diversify the top 1000 results retrieved by our adhoc uogTrA45Vm run, with document coverage learned with validation;
- uogTrB47Vmx (category B) deploys xQuAD to diversify the top 1000 results retrieved by our adhoc uogTrB47Vm run, with document coverage learned with validation.

Table 8 shows the diversification performance of our submitted runs to the diversity task, as well as their corresponding baseline adhoc runs from Section 5. From the table, we first observe that all our category A runs are substantially above the TREC median (up to 29.5% and 14% above the median ERR-IA@20 and α-NDCG@20, respectively). Contrarily to previous editions [16, 28], but similar to our adhoc task observations this year, we note the higher performance of our diversity task category A runs compared to our category B run [26]. More importantly, we observe that our diversity runs consistently improve upon our strongly performing adhoc runs, once again attesting the effectiveness of our xQuAD framework for search result diversification. As for the investigated learning scenarios, we observe no marked benefit in deploying separate validation data during learning. Finally, we note that run uogTrA45Nmx2 achieved the highest ERR-IA@20 and NRBP out of all submitted runs to the Web track 2011 diversity task, while uogTrA45Vmx achieved the overall best performance in terms of α-NDCG@20 [8].

7. CONCLUSIONS

In TREC 2011, we participated in the Web adhoc and diversity tasks, the Microblog real-time search task and Crowdsourcing assessment and consensus tasks, using our Terrier IR platform. In particular, we focused on improving our infrastructure and support for efficient data-driven ranking models within Terrier for the Web and Microblog tracks, and then developed an effective new data-driven filtering approach for tweet search. For the Crowdsourcing track, we integrated a crowdsourcing infrastructure into Terrier for achieving fast generation of relevance assessments based upon Amazon's Mechanical Turk and proposed a new machine learning approach for calculating consensus over multiple assessors.

8. ACKNOWLEDGEMENTS

We would like to thank Amazon's Mechanical Turk for funding support provided for experiments detailed in this paper.

9. REFERENCES

[1] G. Amati. *Probability models for information retrieval based on Divergence From Randomness.* PhD thesis, University of Glasgow, 2003.

[2] G. Amati, E. Ambrosi, M. Bianchi, C. Gaibisso, and G. Gambosi. FUB, IASI-CNR and University of Tor Vergata at TREC 2007 Blog track. In *Proc. of TREC*, 2007.

[3] Amazon Web Services. Amazon Mechanic Turk: ExternalQuestion API Reference. `http://docs.amazonwebservices.com/AWSMechTurk/2008-02-14/AWSMechanicalTurkRequester/ApiReference_ExternalQuestionArticle.html`.

[4] M. Bendersky, W. B. Croft, and Y. Diao. Quality-biased ranking of web documents. In *Proc. of WSDM*, 2011.

[5] M. Bendersky, W. B. Croft, and Y. Diao. Quality-biased ranking of web documents. In *Proc. of WSDM*, pages 95–104, 2011.

[6] M. Box, D. Davies, and W. Swann. *Non-linear optimization techniques.* Oliver & Boyd, 1969.

[7] A. Z. Broder, D. Carmel, M. Herscovici, A. Soffer, and J. Zien. Efficient query evaluation using a two-level retrieval process. In *Proc.of CIKM*, pages 426–434, 2003.

[8] C. L. A. Clarke, N. Craswell, I. Soboroff, and E. M. Voorhees. Overview of the TREC 2011 Web Track. In *Proc. of TREC*, 2011.

[9] G. V. Cormack, M. D. Smucker, and C. L. A. Clarke. Efficient and effective spam filtering and re-ranking for large Web datasets. *Inf. Retr.*, 14(5):441–465, 2011.

[10] C. Crum. Google reveals factors for ranking tweets, 2010. `http://www.webpronews.com/google-reveals-factors-for-ranking-tweets-2010-01`, accessed on 29/09/2011.

[11] J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971.

[12] A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing user studies with mechanical turk. In *Proc. of CHI*, 2008.

[13] M. Lease and G. Kazai. Overview of the TREC 2011 Crowdsourcing Track. In *Proc. of TREC*, 2011.

[14] C. Macdonald, V. Plachouras, B. He, C. Lioma, and I. Ounis. University of Glasgow at WebCLEF 2005: Experiments in per-field normlisation and language specific stemming. In *Proc. of CLEF*, 2005.

[15] R. McCreadie, C. Macdonald, and I. Ounis. Crowdsourcing Blog Track Top News Judgments at TREC. In *Proc. of CSDM*, 2011.

[16] R. McCreadie, C. Macdonald, I. Ounis, J. Peng, and R. L. T. Santos. University of Glasgow at TREC 2009: Experiments with Terrier—Blog, Entity, Million Query, Relevance Feedback, and Web tracks. In *Proc. of TREC*, 2009.

[17] D. Metzler. Automatic feature selection in the Markov random field model for information retrieval. In *Proc. of CIKM*, pages 253–262, 2007.

[18] D. Metzler and W. B. Croft. A Markov random field model for term dependencies. In *Proc. of SIGIR*, pages 472–479, 2005.

[19] I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, and C. Lioma. Terrier: A high performance and scalable information retrieval platform. In *Proc. of OSIR at SIGIR*, 2006.

[20] I. Ounis, C. Macdonald, J. Lin, and I. Soboroff. Overview of the TREC-2011 Microblog Track. In *Proc. of TREC*, 2011.

[21] J. Peng, C. Macdonald, B. He, V. Plachouras, and I. Ounis. Incorporating term dependency in the DFR framework. In *Proc. of SIGIR*, pages 843–844, 2007.

[22] V. Plachouras, I. Ounis, and G. Amati. The static absorbing model for the Web. *J. Web Eng.*, 4(2):165–186, 2005.

[23] S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *Proc. of TREC*, 1994.

[24] R. L. T. Santos, C. Macdonald, and I. Ounis. Exploiting query reformulations for Web search result diversification. In *Proc. of WWW*, pages 881–890, 2010.

[25] R. L. T. Santos, C. Macdonald, and I. Ounis. Selectively diversifying Web search results. In *Proc. of CIKM*, pages 1179–1188, 2010.

[26] R. L. T. Santos, C. Macdonald, and I. Ounis. Effectiveness beyond the first crawl tier. In *Proc. of CIKM*, pages 1937–1940, 2011.

[27] R. L. T. Santos, C. Macdonald, and I. Ounis. Intent-aware search result diversification. In *Proc. of SIGIR*, pages 595–604, 2011.

[28] R. L. T. Santos, R. McCreadie, C. Macdonald, and I. Ounis. University of Glasgow at TREC 2010: Experiments with Terrier in Blog and Web tracks. In *Proc. of TREC*, 2010.

[29] R. L. T. Santos and I. Ounis. Diversifying for multiple information needs. In *Proc. of DDR at ECIR*, pages 37–41, 2011.

[30] R. L. T. Santos, J. Peng, C. Macdonald, and I. Ounis. Explicit search result diversification through sub-queries. In *Proc. of ECIR*, pages 87–99, 2010.

[31] H. Turtle and J. Flood. Query evaluation: strategies and optimizations. *Information Processing and Management*, 31(6):831–850, 1995.