Searching and Filtering Tweets: CSIRO at the TREC 2012 Microblog Track

Sarvnaz Karimi Jie Yin Paul Thomas
Information Engineering Laboratory
CSIRO ICT Centre, Australia
firstname.lastname@csiro.au

Abstract

We report on the participation of the CSIRO[1] team in the TREC 2012 Microblog Track. We participated with four automatic runs for the adhoc search task and four automatic runs for the filtering task. In the adhoc search task, we experiment with different pre-processing and query expansion techniques. Our most important finding is highlighting the value of systematic pre-processing of tweets and its impact on improving effectiveness of search. In the filtering task, we apply different feature extraction and classification techniques. We demonstrate the potential of using SVM classifiers for filtering tweets for a given topic.

1. MICROBLOG TRACK

The Microblog Track was introduced in 2011 to TREC to encourage the information retrieval community to explore techniques for searching and filtering information in microblogs, specifically Twitter. The search task is designed as a typical search scenario where users are interested to see *recent relevant* tweets for their *time-stamped* queries. The filtering task on the other hand has *double-timestamped* queries to specify a time span that started from the last time that the user queried for this topic, till when the latest tweet was tweeted. It assumes that the user has already seen the old tweets prior to the previous tweet-time. An example query for both tasks is shown in Figure 1. In this figure, `<querytime>` and `<querytweettime>` can be used interchangebly.

Topic MB010

Query with one time-stamp:

```
<title> Egyptian protesters attack museum </title>
<querytime> Sat Jan 29 20:06:35 +0000 2011 </querytime>
<querytweettime> 31443107291598848 </querytweettime>
```

Query with double time-stamp:

```
<title> Egyptian protesters attack museum </title>
<querytime> Sat Jan 29 20:06:35 +0000 2011 </querytime>
<querytweettime> 30354903104749568 </querytweettime>
<querynewesttweet> 31443107291598848 </querynewesttweet>
```

Figure 1: A query that shows the difference between adhoc (above) and filtering (below) tasks. Tweetid is used to imply tweeting time, with bigger tweetid indicating a more recent tweet.

[1]Commonwealth Scientific and Industrial Research Organisation

Last year, a tweet corpus called *Tweets2011* was developed for this track which included two weeks' worth of sampled tweets —from 24th of January till 8th of February— comprising approximately 16 million tweets [6]. The same corpus was used for both of the adhoc and filtering tasks in 2012. In 2011 fifty queries, and in 2012 a new set of 60 queries accompanied the tweet corpus for the adhoc task. The filtering task was based on the 2011 queries. These queries however were *double-timestamped* (as shown in Figure 1), from which a subset of 39 queries were used for testing and the rest for training. Since we did not participate last year, we report our experiments using both sets of queries, where applicable, for completeness.

2. REAL-TIME ADHOC TASK

As mentioned earlier, the goal of the adhoc search task is given a *query* accompanied by a *timestamp* to find all the *recent relevant* tweets. It is therefore different from traditional adhoc search for its emphasis on *recency* of the results.

Tweet Corpus Processing

Previous reports in TREC 2011 reported mixed results with different tweet pre-processing techniques, mostly claiming negative effects. However, most of these works only reported the outcome of applying more than one technique and did not explore what the effect of each individual step was. Hence, to fully explore the effect of pre-processing on retrieval effectiveness, we performed a number of pre-processing steps that led to five different variations of the original set of downloaded tweets, as listed in Table 1 and Table 2. Given the requirement of the tasks for not including retweets and non-English tweets, we first removed tweets with null content, and retweets (tweets starting with RT) for all these datasets. We then removed non-English tweets using an off-the-shelf tool called *langid.py* [3]. A dataset that only used this basic level of pre-processing is called `DO`. These basic text processing steps reduced the size of the collection dramatically, with `DO` only containing $5,371,776$ tweets from the original crawl of $15,414,586$ (July 2011).

Other text processing strategies we investigated were a shallow lexical normalisation, mention removal, link removal, emotion removal, and short-tweet elimination (Table 1). We hypothesised that such elements in tweets may not be useful for a adhoc search task, for example because sentiment of a tweet may not be important in identifying topical relevance of the tweet. For lexical normalisation, we only considered words that were emphasised by repeating one or more letters. If a letter was repeated more than three times —we do

not know any legitimate English word that contains more than three repetitions of a letter— it was normalised to one instance of that letter. This process can introduce some errors, for example the word *sweet* is often emphasised in Twitter by repeating the letter *e*, which will be reduced to *swet* in our process. We leave further improvement on this for future work.

Emotional expressions using smilies, emoticons, or some of the Internet slangs which convey emotions such as `lol` or `xoxo` could also increase noise in a tweet. We investigated the effect of removing these words from the tweet as well. Mentions represent username in the form of `@username`, for example `@CSIROnews` or `@DaFemaleBiebzy11`. Some of these mentions reveal the name or nature of the Twitter user, and some are random names people came up with. In our experiments we show the effect of keeping or removing these elements from the tweets.[2]

Statistics on the size of each of the datasets created based on the above steps can be found in Table 3. These pre-processing steps could potentially eliminate some of the relevant tweets from the dataset and consequently the index. We calculated the number of relevant tweets remained in the datasets, and the number (and percentage) of relevant tweets missing from each dataset. Our original crawl already misses 6% of the relevant tweets in 2011 track and 4% in 2012, which means we could never retrieve those tweets even with a perfect system. Language detection almost doubles the percentage of the missing relevant tweets (10% for 2011 and 9% for 2012). This suggests that the tool we used makes mistakes in tagging English tweets as non-English. Other steps cause more elimination of relevants but their deteriorating effect is in the order of one or two percent. Our evaluation results are therefore directly affected by losses in pre-processing.

Indexing and Retrieval

We used Indri search engine [10] with its default settings for indexing and retrieval in all our experiments. We compiled a set of stopwords suitable for Twitter content ourselves.[3] It includes formal English stopwords such as *is*, *am*, informal English stopwordes such as *aint*, *gonna*, and Twitter specific stopwords such as *RT* that indicates a retweet. Note we had already removed re-tweets prior to indexing and RT was only added to the list because it can also occur inside a tweet. For example in this tweet

```
Certainly Is!  :( RT @NSWRFS: Today's #bush-
fire at Wyee - a reminder that fire sea-
son is well and truly here.  #wyeefire
http://instagr.am/p/QW3oJ5kH_W/
```

RT comes in the middle of the text and therefore this tweet is not removed from the original corpus.

We also included a list of common swear words, assuming they do not convey any information useful for the TREC tasks. We used the Krovetz stemmer [2] in all our experiments. We included hashtags as a searchable field via Indri's parameter settings.

[2]Our pre-processing script is available at `https://github.com/skarimi/SKTwitter-Tools`.

[3]Our list of stop-words can be found in: `https://github.com/skarimi/SKTwitter-Tools`

Query Processing

Hashtag Expansion (run csiroQEll112). Hashtags in a tweet can act as explicit markers of topics, as noted for example by Diaz-Aviles et al. [1]. Motivated by this observation, we used hashtags for a simple form of pseudo-relevance feedback. A first round of retrieval used the query as-is to return a set of results $\mathcal{R}_1$; a second round of retrieval used the same query but added in hashtags from $\mathcal{R}_1$ to produce $\mathcal{R}_2$. This second set formed our submission.

Tweets were indexed using Indri, with hashtags in a separate field. Set $\mathcal{R}_1$ was retrieved from indri using this index, with stemming and stopping as above.

Each hashtag h in each of the top k tweets in $\mathcal{R}_1$ was weighted in the conventional manner [9]:

$$w_h = \sum_{\text{top } k \text{ tweets}} \frac{\text{tf}}{k_1((1-b)+b\frac{\text{dl}}{\text{avgdl}})+\text{tf}} \log\left(\frac{N-n+0.5}{n+0.5}\right)$$

where tf is the term frequency of h in each tweet, dl is the length of each tweet, avgdl is the mean length of the top tweets, N is the number of tweets, and n is the number of tweets with tag h. The top-weighted H hashtags were selected and added to the original query with weight β. Re-running this query gave our final $\mathcal{R}_2$.

Experiments with the 2011 queries and relevance judgements suggested best performance with $k = 10$ (ten tweets used for feedback), $H = 3$ (up to three hashtags added), and $\beta = 0.25$ (hashtags have one quarter the weight of the original query). The usual values $k_1 = 1.2$ and $b = 0.75$ were used in weighting terms.

In total, we added hashtags to 49 of the 60 queries (in 11 cases there were no hashtags in the top-10 tweets). Table 4 gives some examples. In many cases, hashtags add terms not in the original query and which are relevant to the topic (e.g. MB057, MB066, MB069, MB078); however in some cases the connection is tenuous, or relates to a different topic (e.g. MB059, MB092) and in several cases the hashtags simply recapitulate the query text (e.g. MB099).

Text Expansion (run csiroQEt112). Run csiroQEt112 used pseudo-relevance feedback as above, but candidates were drawn from all terms in the tweet (not just hashtags). All queries were expanded with three terms drawn from the top-10 tweets, and expanded terms were added to the original query with multiplier $\beta = 0.75$.

Entity Recognition (run csiroNE112). We used a Twitter-specific named-entity recogniser [7] to find interesting entities such as names of movie, locations, or people in the queries. Run csiroNE112 used pseudo-relevance feedback over named entities (only), in the same manner as our other runs.

Search Evaluation

We evaluate our approaches to the adhoc search task in two stages. We first evaluate the effect of tweet corpus pre-processing in vanilla runs (no query expansion), and then assess the effect of query expansion. Table 5 shows the effect of different pre-processing approaches on tweets before indexing. For simplicity, these runs are named after their corresponding datasets explained in Table 2. We observed

What	How
Lang. Detection	Removed all non-English tweets using *langid.py*.
Emotion Removal	Removed all the common words that convey feelings such as `lol`, `haha`, or `xoxo`, and emoticons such as `:-)`, ☼, or ☺, and Japanese emoticons such as `(>_<)`.
Lexical Normalisation	If a character or was repeated four or more times in a tweet, it was reduced to one. For example, `yesssssssss` was turned to `yes`.
Mention Removal	All the Twitter username mentions (i.e. `@username`) were removed.
Link Removal	All the web-links (e.g. `http://t.co/sOMqSVXq`) in the tweets were removed.
Short-tweet Elimination	Any tweet that that contained less than four words, after removing punctuations, was eliminated.
Retweet Elimination	Any tweet started with `RT` followed by a username was eliminated.

Table 1: Tweet pre-processing heuristics.

Dataset	Lang. Detect.	Emotion Removal	Lexical Norm.	Mention Removal	Link Removal
D0	✓				
D1	✓	✓	✓	✓	✓
D2	✓	✓	✓	✓	
D3	✓	✓		✓	
D4	✓	✓	✓		✓
D5	✓	✓	✓		

Table 2: Specification of the pre-processed datasets indexed for the baseline runs. A ✓in a column means the corresponding dataset was pre-processed for the specified text processing heuristic.

		TREC 2011			TREC 2012		
Dataset	# tweets	#rs	#mrels	(%)	#rels	#mrels	(%)
Original	15 414 586	2784	181	(6.1)	6018	268	(4.3)
D0	5 371 776	2666	299	(10.1)	5695	591	(9.4)
D1	4 304 454	2621	344	(11.6)	5624	662	(10.5)
D2	4 342 660	2633	332	(11.2)	5661	625	(9.9)
D3	4 340 852	2633	332	(11.2)	5662	624	(9.9)
D4	4 439 881	2622	343	(11.6)	5631	655	(10.4)
D5	4 473 471	2634	331	(11.2)	5666	620	(9.9)

Table 3: Size of the *original* dataset downloaded from Twitter, and processed datasets, with the total number of relevant tweets (#rels) they include, the number of relevant tweets missing (#mrels) and the percentage of the missing relevants over total number of existing relevants in brackets. Total number of existing relevants was 2965 for TREC 2011, and 6286 for TREC 2012.

Topic	Original query	Hashtags added
MB057	chicago blizzard	#blizzard, #chicago, #smomg
MB059	glen beck	#mostannoyingtvpeople
MB066	journalists treatment in eygpt	#eygpt, #media, #jan25
MB069	high taxes	#taxsupport, #sotu
MB078	mcdonalds food	#ronald, #mcdonald, #healthy
MB092	stock market tutorial	#photoshop, #tutorial
MB099	superbowl commercials	#superbowl

Table 4: Examples of hashtag-based query expansion. Hashtags are stemmed.

Dataset	TREC 2011			TREC 2012		
	MAP	P@30	R-Prec	MAP	P@30	R-Prec
D0	0.2888	0.3408	0.3459	0.1925	0.2949	0.2423
D1	0.3024	0.3619	0.3570	0.2056	0.3085	0.2645
D2	0.2949	0.3565	0.3555	0.2006	0.3023	0.2540
D3	0.2950	0.3565	0.3555	0.2007	0.3023	0.2540
D4	0.3032	0.3592	0.3570	0.2062	0.3090	0.2613
D5	0.2955	0.3558	0.3537	0.2009	0.3073	0.2557

Table 5: Effect of dataset pre-processing for two sets of queries (2011 and 2012). Tweets and queries were stopped and stemmed.

Metric	Run				
	D4	isiFDL	ri	Best	Median
P@30	0.3592	0.4551	0.4265	0.6116	0.2575
MAP	0.3032	0.1923	0.2227	0.5127	0.1426

Table 6: Comparison of one of our baseline runs with top-performing real-time and non-real-time runs, median and best in TREC 2011.

that apart from D0, the rest of these datasets led to similar results in terms of three measures: mean average precision (MAP), precision at cut-off 30 (P@30), and recall of preferences (R-Pref). Note these runs are based on the entire corpus, therefore they use future evidence and cannot be categorised as real-time runs. TREC 2011 relied on P@30 to rank the submissions. We ran a statistical significance test (paired t-test) over 2011 results to compare these baseline runs. D0 was significantly (p-value<0.05) inferior to all other runs for all the metrics. D4 was significantly (p-value<0.05) better than all other runs in terms of MAP score.

In Table 6, we compare a baseline run based on dataset D4 with official results of best-performing real-time run from last year (isiFDL) from University of Southern California [4], highest performing non-real-time run (ri) from Kobe University [5], median and best results to show where a solely pre-processing based baseline would stand if we had participated in TREC 2011. These numbers are reported for relevance judgements based on all-relevant tweets. Note that *best* is a synthetic run composed of the top score for each query from all submissions

Our simple run (D4) is well above median of the submitted runs for last year. Comparing it to the official ranking of all the participated teams [6], surprisingly, its MAP score places this run at third rank, and P@30 would rank it 14th. This simple comparison gives us confidence that our preprocessing steps are useful.

In all the following experiments including our submitted runs we used an index based on D4 dataset, which means all the non-English tweets were removed, and within tweets words indicating emotions, emoticons, and also links to web pages were omitted. We submitted a *real-time* run(csiroR112) in which we used the same method of creating D4 but we created a separate index for each topic removing all the tweets with a timestamp after the tweettime. This way, we did not include any future evidence in our run.

We compared our query expansion strategies with a baseline run in using topics from the 2011 and 2012 tracks (Table 7). These results are based on all-relevant judgements. A query expansion strategy (csiroQEt112) that used all the query terms (hashtags and non-hashtags) performs better than all our other runs for TREC 2011 for all the metrics, and performs best based on P@30 and R-Prec in TREC 2012. For TREC 2012 queries, csiroQEll112 marginally wins over csiroNE112.

Run	TREC 2011			TREC 2012		
	MAP	P@30	R-Prec	MAP	P@30	R-Prec
D4	0.3032	0.3592	0.3570	0.2062	0.3090	0.2613
csiroR112	0.2860	0.3469	0.3319	0.1542	0.1324	0.1675
csiroQEll112	0.2943	0.3463	0.3521	0.1616	0.1393	0.1767
csiroQEt112	0.3108	0.3639	0.3603	0.1537	0.1445	0.1896
csiroNE112	0.2945	0.3408	0.3476	0.1605	0.1363	0.1770

Table 7: Effect of a vanilla run on D4 index (both realtime and non-real-time), and query expansion methods based on the submitted runs for two sets of queries (2011 and 2012).

Table 8 shows a comparison of our submitted runs with TREC official results released for the best, median, and worst synthetic aggregated runs submitted for this task. These results are based on highly relevant judgements. Our best run in terms of MAP score was csiroQEt112 which was based on pseudo-relevance feedback query expansion using all the terms in the tweets. This run was near median for MAP and above median using R-Prec measure. Using hash-tags for query expansion (csiroQEll112) led to our highest score using P@30, but it was poor in MAP.

3. REAL-TIME FILTERING TASK

An important aspect of Twitter is its near real-time streaming of tweets, which makes it a candidate application for information filtering. Traditionally, filtering techniques develop user profiles that distinguish between relevant and irrelevant texts in an incoming stream [8]. The 2012 Microblog Track introduced an adaptive filtering task where at the beginning only a small number of positive or relevant tweets are available —only one for this task— and the system should gradually learn to create a more accurate profile for a given topic.

We submitted four runs for this task, of which one was based on traditional pseudo-relevance feedback and three were based on two different types of classifiers: support vector machines (SVM) and logistic regression. We used the same pre-processed corpus (D4) that we used as baseline for the adhoc runs.

Relevance Feedback for Filtering

We submitted one run based on traditional relevance feedback as a baseline. That is, we wanted to know how well a plain retrieval approach would work compared to a more sophisticated machine learning approach. We first retrieved an initial ranked list for each query, and then to simulate judgements by a user, we expanded the queries using the provided judgements of the top 5 retrieved tweets. Obviously, if there was no relevant tweet retrieved in the original retrieved tweets, the query remained unexpanded. If the query was expanded, we then retrieved a new set of tweets that satisfied the time limits given in each query. Only the top 10 newly retrieved tweets were considered as final relevant tweets to construct a run. This run is called csiroQERF111. Although in adaptive filtering this process of getting feedback from the user continues, we did not iterate

Metric	Submitted Runs				Best	Median	Worst
	csiroR112	csiroQEll112	csiroQEt112	csiroNE112			
P30	0.1542	0.1616	0.1537	0.1605	—	—	—
MAP	0.1324	0.1393	0.1445	0.1363	0.4144	0.1486	0.0099
R-Prec	0.1675	0.1767	0.1896	0.1770	0.4473	0.1869	0.0131

Table 8: Comparison of CSIRO TREC 2012 adhoc search submitted runs with official results of best, median and worst submissions.

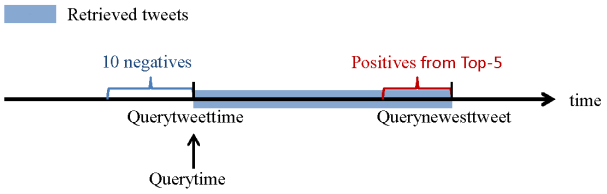

Figure 2: Illustration of constructing training data for classification.

this process any further.

Classification-based Filtering

Our other three runs cast the filtering task as a binary classification problem. Given a topic, we can use the entire corpus prior to the querytweettime as training data, and build a classifier using positive and negative examples. The classifier is thereafter applied on each tweet from querytweettime to querynewesttweet to determine whether or not the tweet is relevant to the topic. However, a main difficulty lies in that, we only have one positive example – querytweet – available for training. Thus, we focused on gradually enriching positive examples for training a classifier. For each topic, we first retrieved an initial ranked list of tweets using Indri, from which we examined the top 5 tweets and those tweets judged as relevant were used as positive examples. Besides that, another 10 tweets posted before querytweettime were taken as negative examples. Together positive and negative examples comprised the labelled training data for building a classifier for the query. This process is illustrated in Figure 2. Using the training data, we trained a classifier based on SVM or logistic regression. When applied on new tweets, the classifier not only generated a retrieval decision about whether each tweet is relevant to the given topic, but also returned the probability of being classified to be positive as the relevance score. Details of feature extraction and the corresponding submitted runs are given below.

Feature Extraction. In order to train the classifier, we extracted three types of features for each tweet: n-grams generated on the text of the tweet, the hashtags that the tweet uses, and whether or not the tweet contains terms from the query text. Using these features, tweets were converted into feature vectors upon which the classifier was built.

Tweet Classification

We submitted three runs using the classifiers. They are detailed as follows:

- csiroSVMqe111: An initial ranked list was retrieved using Indri with expanded queries similar to the baseline run (csiroQERF111). The top 5 judgements were used as positive examples to train a SVM classifier. The probability of being classified as positive was returned as the score.
- csiroshuq111: An initial ranked list was retrieved using Indri, from which the top 5 judgements were used as positive examples to train a SVM classifier. The probability of being classified to be positive was returned as the score.
- csirolrhuq111: It used the same approach to construct the training data as above, except that it used logistic regression as the classifier. The score returned for each tweet was the probability of being relevant to a specific topic.

Filtering Evaluation

The filtering task was evaluated using two main metrics: $F_{0.5}$ and $T11SU$ [8]. $F_{0.5}$ is defined based on F_β measure

$$F_\beta = (1+\beta^2).\frac{\text{precision.recall}}{(\beta^2.\text{precision}) + \text{recall}},$$

where $\beta = 0.5$. T11SU or scaled linear utility measure is defined as

$$T11SU = \frac{max(T11NU, MinNU) - MinNU}{1 - MinNU},$$

where

$$T11NU = \frac{2 \times TP - FP}{2 \times N}.$$

TP or true positive rate is the number of relevant tweets retrieved, FP or false positive is the total number of irrelevant tweets retrieved, N is total number of relevant documents, and $MinNU$ is the minimum negative utility that a user would tolerate (e.g. -0.5).

Two other measures, precision and recall, are also reported to help understand F-measure.

Table 9 reports the effectiveness of our submitted runs compared to the best, median, and worst results provided by TREC based on all the submissions. Except for csiroLRHUQ111 that was based on a logistic regression classifier, other runs were above median for both main measures ($F_{0.5}$ and $T11SS$). Among our four submitted runs, csiroQERF111, relevance-feedback run, achieved the best F-measure mainly because its high precision. On the other hand, the effectiveness of the other three runs, based on tweet classification, varies for different topics, with some queries benefiting greatly from a classifier-based approach and the others not. Our csiroshuq111 run that is based on an SVM classifier achieves precision at 0.2688, recall at 0.2294, and F-measure at 0.1594, which outperforms the other two runs. csiroSVMqe111 uses query expansion to perform relevance feedback based on the results returned by Indri, but it does

	Submitted Runs						
Metric	csiroQERF111	csiroSVMQE111	csiroLRHUQ111	csiroSHUQ111	Best	Median	Worst
$F_{0.5}$	0.2631	0.1580	0.0809	0.1594	0.6074	0.1491	0.0000
T11SU	0.3448	0.3227	0.0980	0.2971	0.5967	0.2076	0.0000
Precision	0.4781	0.1953	0.0821	0.2688	0.9224	0.1767	0.0000
Recall	0.1427	0.2217	0.3431	0.2294	0.9463	0.3343	0.0002

Table 9: Comparison of CSIRO TREC 2012 filtering submitted runs with mean of the official best, median, and worst results reported for all the test queries by all participating teams.

not help with the accuracy. csirolrhuq111 using logistic regression performs worse than two SVM-based runs and only leads to high recall value compared to the median.

To gain a better understanding on how our classification-based runs rely on the initial retrieval from Indri, we compared precision at cut-off 5 (P@5) of the an initial ranked list (no feedback) and precision gained by csiroSHUQ111. We observed that when the original run retrieves more than two relevant tweets in the top 5 results (two positive examples), the classifier tends to achieve a high precision. We also calculated Pearson's correlation of per-query precision for these two runs. For *no feedback* run, correlation between P@5 and precision was 0.4249 ($p = 0.0078$), and for csiroSHUQ111, correlation between P@5 and $F_{0.5}$ was 0.5935 ($p < 0.0001$). This suggests a weak positive correlation between the number of positive examples and effectiveness of the classifier for this task.

4. CONCLUSIONS

We reported our experiments in the TREC Microblog Track for both search and filtering tasks. In the adhoc task we experimented methods of processing tweets before indexing to reduce non-content words or words such as expressions of emotions or links to web pages. Our initial experiments show that these simple heuristics indeed lead to a promising baseline for this task. However, given the very short length of tweets, care should be taken to apply these heuristics because of their potential of removing relevant tweet from the index. It also worth noting that an application can determine what pre-processing would be required.

We also reported our experiments on the filtering task, where given a query and one relevant tweet we are to retrieve recent tweets relevant to that query. We investigated using a simple relevance feedback baseline, and two different classifiers (SVM and logistic regression). The effectiveness of our runs were all above median for the main evaluation metrics except for logistic regression which performed poorly. In the future we intend to explore how adaptive filtering can be more effectively implemented using tweet classifiers.

5. REFERENCES

[1] E. Diaz-Aviles, P. Siehndel, and K. D. Naini. Exploiting social #-tagging behavior in Twitter for information filtering and recommendation. In *The Twentieth Text REtrieval Conference Proceedings*, 2011.

[2] R. Krovetz. Viewing morphology as an inference process. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 191–202, Pittsburgh, PA, 1993.

[3] M. Lui and T. Baldwin. langid.py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*, pages 25–30, Jeju Island, Korea, 2012.

[4] D. Metzler and C. Cai. USC/ISI at TREC 2011: Microblog track. In *The Twentieth Text REtrieval Conference Proceedings*, 2011.

[5] T. Miyanishi, N. Okamura, X. Liu, K. Seki, and K. Uehara. TREC 2011 microblog track experiments at Kobe university. In *The Twentieth Text REtrieval Conference Proceedings*, 2011.

[6] I. Ounis, J. Lin, and I. Soboroff. Overview of the TREC 2011 microblog track. In *The Twentieth Text REtrieval Conference Proceedings*, 2011.

[7] A. Ritter, S. Clark, Mausam, and O. Etzioni. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, Edinburgh, UK, 2011.

[8] S. Robertson and I. Soboroff. The TREC 2002 filtering track report. In *The Eleventh Text REtrieval Conference Proceedings*, 2002.

[9] S. Robertson and H. Zaragoza. *The Probabilistic Relevance Framework: BM25 and Beyond*, volume 3 of *Foundations and Trends in Information Retrieval*. now Publishers, Delft, 2009.

[10] T. Strohman, D. Metzler, H. Turtle, and W. B. Croft. Indri: A language model-based search engine for complex queries. In *Proceedings of the International Conference on Intelligence Analysis*, 2004.