HLTCOE at TREC 2014: Microblog and Clinical Decision Support

Tan Xu
University of Maryland
College Park

Paul McNamee
Johns Hopkins University
HLTCOE

Douglas W. Oard
University of Maryland
College Park

Abstract

Our team submitted runs for both the Microblog and Clinical Decision Support tracks. For the Microblog track, we participated in both the temporally anchored ad-hoc search and the tweet timeline generation subtasks. On the Clinical Decision support task, our efforts were time limited, and our main contribution was to investigate controlling for morphological variation in this technical domain.

1 Introduction

We participated in two tracks for TREC 2014, Microblog and Clinical Decision Support. In this paper we describe our work for each in turn.

2 Microblog

The Microblog track has been conducted for four consecutive years. In this year's evaluation, in addition to the traditional temporally anchored ad-hoc search task, a new Tweet Timeline Generation (TTG) task was introduced with the purpose of providing users a more succinct list of search results. The TTG task targets a form of automatic summarization, and single tweets are grouped into topical clusters. Therefore, the expected outcome is a list of tweets that cover as many clusters as possible (high aspectual recall) with as low a number of off-topic or topically redundant clusters as possible (high cluster precision) and as few topically unrelated tweets in a cluster (high item precision). Definitional question answering has been evaluated at TREC using similar means.

Ad-hoc search is an obvious pre-processing step for timeline generation because: (1) it helps to prefilter topically irrelevant tweets; and (2) the retrieval score provides a meaningful ranking of tweets as input. Thus, we begin our Microblog track work with the temporally anchored ad-hoc search task, as described in section 2.1. We then describe our process for the TTG task in section 2.2.

2.1 Temporally-Anchored Ad-hoc Search

This task requires us to return at most 1000 relevant tweets published no later than time T, given query Q, which is expressed as a few topic-related keywords. Standard ranked retrieval measures are used to evaluate effectiveness, including: mean average precision (MAP), precision at rank 30 (P@30) and R-precision. This suggests two interrelated research questions: (1) what features can be used to determine a tweet's position in a query-focused relevance-ranked list; and (2) given these relevance-oriented features, which ranking functions generate better ranked lists.

2.1.1 Expansion

When working with a bag-of-words text representation, as we do, one challenge confronting microblog retrieval is the vocabulary sparsity issue. This sparsity arises not only because of the brevity of tweets, and the use of subjective language, but also because the queries are short (typically 2-3 keywords). We therefore tried both query expansion and tweet expansion. The last three years of Microblog track papers have shown substantial, consistent, and significant improvements in retrieval effectiveness from the use of expansion.

Specifically, the following three expansion tech-

niques are applied in our retrieval system, with the parameters tuned by using topics 1-110 from the Tweets 2011 collection:

- Query expansion with the Google custom search engine (GSE). As reported by (El Din and Magdy, 2012; El-Ganainy et al., 2013), Google search results offer a useful basis for query expansion. Thus, we created our own GSE[1] from the 123 most common linked domain names found in URL in the Tweets 2011 collection. Examples of these domains include $*.nytimes.com$ and $*.cnn.com$. From the top K_{GSE} (tuned to 20) returned GSE results,[2] we extract the descriptive snippet (1-2 sentences) for each result, and we pick the top W_{GSE} (tuned to 90) most frequently used words across all of these snippets to expand the original query term vector $\mathbf{Q_0}$ as follows: (α_1 is tuned to 0.25):

$$\mathbf{Q_{exp}} = (1 - \alpha_1) \cdot \mathbf{Q_0} + \alpha_1 \cdot \mathbf{Q_{GSE}} \quad (1)$$

- Query expansion with Pseudo-Relevance Feedback (PRF). PRF is a widely used query expansion technique that has been shown to improve (average) retrieval effectiveness in various tasks. In our work, we apply PRF after GSE, which in preliminary experiments was the ordering that generated the better results. We run PRF for R (tuned to 5) rounds so that the expanded queries could better converge. For each round, the same expansion function is used:

$$\mathbf{Q_{exp}} = (1 - \alpha_2) \cdot \mathbf{Q_{exp}} + \alpha_2 \cdot \mathbf{Q_{PRF}} \quad (2)$$

, where Q_{PRF} is the most frequent W_{PRF} (tuned to 50) words used in the top K_{PRF} (tuned to 20) retrieved tweets (α_2 is tuned to 0.1).

- Tweet expansion with embedded URL. For each tweet retrieved from the Microblog Track corpus service, we follow any embedded Web links, parse the content text from the raw HTML, and if all of the initial query terms are present in the extracted text of the page use all of that text on the page to expand the original tweet.[3] We include the requirement that the page used for expansion must contain ALL of the initial query terms because in initial experiments we found that in many cases links had been made to Web pages that contained non-relevant (or principally non-text) content.

[1]https://www.google.com/cse

[2]GSE results were customized to return only results that had been indexed by Google before the query time, specified in the query.

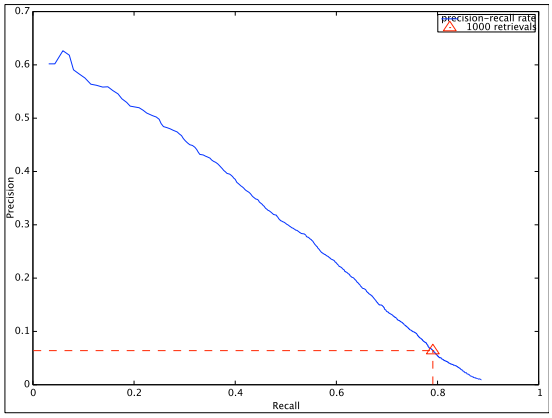

Figure 1: Precision-Recall tradeoff for the corpus service using expanded queries, Topics 1-110.

2.1.2 Learning-based Re-ranking

Starting in 2013, the Microblog track adopted a corpus-as-service evaluation structure, which can be considered as a rough way of pre-selecting putatively relevant tweets. Figure 1 shows a precision-recall curve, averaged over topics 1-110, after query expansion using both GSE and PRF. To further improve retrieval effectiveness, we then re-rank these returned tweets, with the goal of placing those most likely to truly be relevant as near the top of the ranked list as possible. We do this using Learning-To-Rank (L2R), with topics 1-110 used for training and topics 111-170 used for development testing.

We employ the *RankLib* toolkit,[4] which implements 8 popular L2R algorithms Given a query-tweet pair, we generate 9 query-dependent features, and 8 query-independent features. The 8 query-dependent features include:

[3]We use *Goose* to extract the text; see https://github.com/GravityLabs/goose

[4]http://sourceforge.net/p/lemur/wiki/RankLib/

- The corpus service retrieval score, which is the Indri language model implementation with Bayesian smoothing and a Dirichlet prior (Zobel and Moffat, 2006);
- The score from our own unigram language model with Bayesian smoothing and a Dirichlet prior, with the background tweet unigram language model trained on around 1B English tweets formed as a 15% sample of all Tweets sent between 5/25/2009 and 10/17/2010 (O'Connor et al., 2010);
- The Okapi BM25 score (with k1 tuned to 0.1, b tuned to 0.2, and the average tweet length set to be 28 words), with the IDF term approximated from the same 1 billion tweet collection;
- Cosine similarity scores based on TF alone or on TF·IDF.

With the exception of the corpus service retrieval score (which is computed only on unexpanded tweets), each of these query-dependent feature types are calculated separately for unexpanded and expanded tweets; this yields a total of 9 query-dependent features. Tweet expansion can time out (we set up a maximum 3 seconds waiting time for raw HTML fetching and parsing), and it sometimes adds more noise than useful signal; including features based on unexpanded tweets helps to mitigate both risks.

Our goal in including query-independent features is to characterize some aspects of the "quality" of a tweet. We compute the following 8 features:

- A binary feature to indicate whether a tweet contains at least one external URL;
- A binary feature indicate whether a tweet contains at least one hashtag;
- A scaled number (at 10) indicating the tweet's retweet count (relative to other tweets);
- The length of the tweet, in words;
- The length of a tweet, in "informative" words (i.e., excluding common English stopwords and tweet-specific stopwords such as "rt" or "http");
- The number of stopwords in a tweet;
- The percentage of the words in a tweet that are informative words;
- The percentage of the words in a tweet that are stopwords.

Table 1 shows Mean Average Precision (MAP) averaged over the development test queries for 8 L2R algorithms. Coordinate Ascent (CA) (Metzler and Croft, 2007) gives the best MAP with tweet expansion features (a 7.75% statistically significant improvement, compared to not reranking), and the second-best MAP without the tweet expansion features. We therefore elected to use CA for our submitted runs.

2.2 Tweet Timeline Generation

Working with the results from the ad-hoc search task, the TTG task tries to further reduce the amount of effort a user would need to expend to read at least one instance of the content of all topically relevant tweets. The track guidelines[5] propose a clustering as one possible baseline, but prior experience with novelty detection suggests that clustering meaningfully can be difficult, and that simply returning some set of K top-ranked tweets can be a hard baseline to beat. We therefore used a simple top-K baseline during our system development process.

2.2.1 Statistics of Training Data

In this first year of the TTG task there are only 10 training queries, each of which was manually created from past years' search tasks. Some statistics from the ground truth for the training topics are shown in Table 2. Note that more than 75% of the clusters contain only 1 tweet. From these training topics, we choose to hold out topic 3 as a development test topic because the statistics for topic 3 seemed to us to be reasonably representative.

2.2.2 Tweet Novelty Detection

Starting from our Top-K baseline, a straightforward improvement is to remove redundant tweets. By doing this, we convert the TTG task into a sequential binary decision task in which we ask whether each

[5] https://github.com/lintool/twitter-tools/wiki/TREC-2014-Track-Guidelines

L2R without tweet expansion (13 features)							L2R with tweet expansion (17 features)						
Algorithms	MAP	Difference	Improvement	Win	Loss	p-value	Algorithms	MAP	Difference	Improvement	Win	Loss	p-value
Baseline (without L2R)	0.4248	0.0000	(0.00%)	0	0	0.0000	Baseline (without L2R)	0.4248	0.0000	(0.00%)	0	0	0.0000
MART	0.4238	-0.0010	(-0.24%)	30	30	0.9257	MART	0.4343	0.0095	(+2.24%)	39	21	0.3938
RankNet	0.4266	0.0019	(+0.44%)	52	5	0.0000	RankNet	0.4266	0.0019	(+0.44%)	52	5	0.0000
RankBoost	0.4275	0.0027	(+0.63%)	38	21	0.2478	RankBoost	0.4365	0.0117	(+2.76%)	43	17	0.0026
AdaRank	0.1576	-0.2672	(-62.91%)	2	58	0.0000	AdaRank	0.4446	0.0198	(+4.67%)	39	21	0.0502
Coordinate Ascent	0.4431	0.0183	(+4.31%)	44	16	0.0114	**Coordinate Ascent***	**0.4577**	**0.0329**	**(+7.75%)**	45	15	0.0001
LambdaMART	0.4233	-0.0015	(-0.34%)	36	24	0.8772	LambdaMART	0.4352	0.0105	(+2.47%)	35	25	0.3032
ListNet	0.4448	0.0200	(+4.71%)	45	15	0.0000	ListNet	0.4430	0.0182	(+4.3%)	42	18	0.0010
RandomForest	0.4229	-0.0019	(-0.45%)	32	28	0.8625	RandomForest	0.4337	0.0090	(+2.11%)	37	23	0.4215

Table 1: MAP with various L2R Algorithms over Topics 111-170

Topic ID	#Relevance	#Cluster	#Ave. tweets per cluster	%Redundancy in relevance	%Unary cluster
3	38	20	1.90	47.40%	65.00%
21	155	46	3.37	70.30%	69.50%
22	148	45	3.29	69.60%	84.40%
26	144	102	1.41	29.20%	85.30%
42	34	11	3.09	67.60%	54.50%
51	61	52	1.17	14.80%	92.30%
57	104	66	1.58	36.50%	74.20%
66	190	133	1.43	30.00%	80.50%
68	165	86	1.92	47.90%	73.30%
88	269	87	3.09	67.70%	74.60%
Ave.	130.8	64.8	2.225	48.10%	75.36%

Table 2: Clustering Statistics for TTG Training Topics, ground truth.

Algorithm 1: Shingling/hashing De-duplication

$\mathcal{C} \leftarrow \{\text{SEARCH RESULTS}\}$
$\mathcal{R} \leftarrow \{\}$
$\mathcal{H} \leftarrow \{\}$
for $d \in \mathcal{C}$ **do**
  $S_d \leftarrow \text{SHINGLING}(d, k)$
  $H_d \leftarrow \text{HASHING}(S_d)$
  $\Pi_d \leftarrow \text{SHUFFLE}(H_d)\{1 \dots m\}$
  if $(\Pi_d \cap \mathcal{H})/m < \tau$ **then**
    $\mathcal{H}.\text{ADD}(H_d)$
    $\mathcal{R}.\text{APPEND}(d)$
return $\mathcal{R}$

tweet contains enough novel content to be retained, given the tweets we have previously decided to retain. We call this approach novelty detection (or de-duplication). Set-based methods have been shown to be effective in similar novelty detection tasks (Allan et al., 2001; Soboroff and Harman, 2005). The key question, therefore, is how best to measure a tweet's novelty given a previous set of (putatively) novel tweets. In our experiments, we tried both set-based similarity and binary classification.

For set-based similarity, we compared 3 similarity measures: cosine similarity, Jaccard's coefficient, and shingling/hashing. Algorithm 1 shows our shingling/hashing algorithm, in which we employ Horner's method to hash shingles and set up a hash value space of cardinality 40,000,000. There are 3 parameters to tune: the k-shingles extracted from each tweet, the number m of k-shingles selected for comparison, and the threshold τ.

Table 3 shows effectiveness measures for the Top-K (tuned to 90), Jaccard's coefficient (tuned threshold = 0.8, K = 80), cosine similarity (tuned threshold = 0.6, K = 110), and shingling/hashing (tuned K = 90, k = 10, m = 20, τ = 0.5) on Topic 3 (the one held-out development topic). As can be seen, some improvement on this one topic is observed, suggesting that set-based approaches to novelty detection seem promising.

We can then add more novelty features in combination with the 8 tweet quality features used in the search phase. Totally, there are 15 new features used, which include:

- 1 relevance score feature ($\in \mathbb{R}_{>0}$) from the CA L2R result;
- 8 set-based similarity features, where all previously selected tweets form the set, which is then used to calculate some similarity score (cosine, Okapi-BM25, unigram language modeling with Bayesian smoothing) with the (un-expanded or expanded) new tweet;
- 2 set-extreme similarity features, where only the maximum (Jaccard or cosine) similarity, maximized over all tweets in the set, is considered;
- 3 set-based distance features, which are either the number of new terms introduced by the new tweet (either un-normalized, or normalized by the length of the new tweet), or the elapsed

minutes between the latest tweet in the set and the new tweet;[6]

- 1 summary length control feature, which is currently effectively unused (it is simply set to the number of tweets in the current selected set).

Because we are interested in using a supervised binary classifier to learn how to make the novelty predication from the training topics, we manipulate the manual ground truth clustering of the training topics using the following steps to create novelty detection training data for each training topic:

Step 1: create an empty sample pool;

Step 2: collect up to 2000 tweets from the top search results (together with their CA L2R score), and put them into the sample pool;

Step 3: if any tweet in the manually created clusters does not exist in the sample pool, add it and assign a relevance score of 0;

Step 4: for each manually created ground truth cluster, mark the tweet with the highest relevance score as positive;

Step 5: mark all the rest of the tweets in the sample pool as negative.

Step 6: for each tweet in the sample pool, create feature vectors as listed above ($\in \mathbb{R}^{23}$) performing the calculation in order of decreasing CA L2R score, and incrementally adding only positive exemplars to the set.

There are plenty of binary classifiers we could choose from, but we have limited training data on which to base our choice. We visualized the feature space of the novelty detection training data, as shown in Figure 2,[7] which is a $\mathbb{R}^2$ projection from principal component analysis (PCA) with 55% of the variance preserved. This distribution suggests a non-linear classifier. We therefore chose to use a Support Vector Machine (SVM) with a radial basis kernel function.

The classifier's effectiveness on Topic 3, the one held-out development test topic, is shown in Table 3.

[6]The elapsed minutes could be negative, which would indicate seeing a new tweet that predates the last tweet in the set, since the tweets are processed in relevance rather than temporal order.

[7]In this figure, we show randomly selected negatives along with a balanced number of positives.

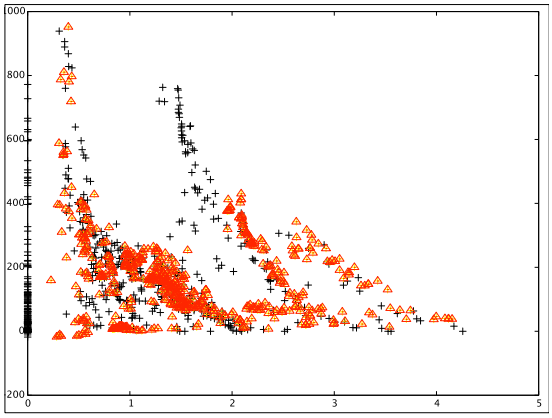

Figure 2: PCA $\mathbb{R}^2$ Projection of the Novelty Detection Training Data.

	Precision	Recall	F1
Top-K relevance	0.18889	0.85000	0.30909
Jaccard's coefficient	0.19231	0.75000	0.30613
Cosine similarity	0.20690	0.90000	0.33645
Shingling/hashing	0.21111	0.95000	0.34545
SVM (RBF kernel)	**0.22353**	**0.95000**	**0.36191**

Table 3: Performance of various Novelty Detection Methods on TTG.

2.3 Evaluation

For this year's Microblog track, there are 55 evaluation topics (topic 171-225). This section summarizes our submitted runs and analyzes the results.

2.3.1 Submitted Runs

We submitted a total of 4 ad-hoc search runs and 4 TTG runs. Table 4 summarizes our results.

2.3.2 Result Analysis

For our ad-hoc search runs, we achieved a MAP improvement over the baseline (hltcoe0) of 0.1027 absolute ($\sigma = 0.1734$) using GSE alone, a further improvement of 0.0154 absolute ($\sigma = 0.0426$) using GSE followed by PRF, and yet a further improvement of 0.0377 absolute ($\sigma = 0.0703$) using CA L2R (with tweet expansion). Paired t-tests show that each of these incremental improvements is statistically significant at $p < 0.05$. As a comparison, among all 73 automatic runs submitted, our best run (hltcoe3) achieved the maximum reported Average Precision (AP) for 19% of the topics and exceeded the median AP by for 87% of the topics. Corresponding figures for P@30 35% and 69% of the top-

Ad-hoc Search Runs				
Run	**Description**	MAP	P@30	R-Prec
hltcoe0	baseline from corpus API	0.4149	0.6236	0.4434
hltcoe1	query expansion with GSE	0.5176	0.6733	0.5072
hltcoe2	query expansion with GSE+PRF	0.5330	0.6812	0.5211
hltcoe3	L2R from hltcoe2 with tweet expansion	0.5707	0.7121	0.5660
Tweet Timeline Generation Runs				
Run	**Description**	Recall(Unweighted)	Recall(Weighted)	Precision
hltcoeTTG0	top 90 tweets from hltcoe3	0.4422	0.6356	0.2295
hltcoeTTG1	novelty detection with cosine similarity	0.4029	0.5915	0.3407
hltcoeTTG2	novelty detection with SVM	0.4622	0.6534	0.2909
hltcoeTTG3	novelty detection with shingling/hashing	0.4869	0.6658	0.2976

Table 4: 2014 Microblog Track Results, Sumbitted Runs.

ics, respectively.

However, we do note that GSE introduces large variance across topics, and for several topics the effectiveness measures with GSE alone are actually lower than the baseline. Figure 3 illustrates the nature of this problem for topic 175 "commentary on naming storm nemo", for which GSE the greatest adverse effect. For this figure, we have manually clustered the informative query terms (after expansion) according to their semantic meaning (as we interpret that meaning), and then shown the fraction of the terms that appear in each cluster. For example, the original query contains four (non-stopword) terms, so 25% of the query is present in each cluster. The query drift resulting from expansion is clearly evident in this case, with the focus (GSE and PRF) changing from naming the storm to the storm itself (According to the manual evaluation description, relevant tweets should include "discussion/commentary/jokes about naming the northeaster storm Nemo".)

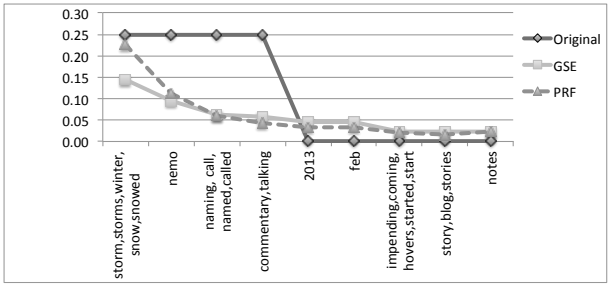

Figure 3: Query Drift from Query Expansion for Topic 175.

With regard to the TTG runs, we see substantial and statistically significant improvements over the baseline (hltcoeTTG0) from all three set-based novelty detection methods that we tried both for weighted (where the importance of cluster is considered according to the number of tweets and informative tweets contained) and unweighted (where clusters are treated equally) evaluation measures. However, we note that our supervised binary classifier (hltcoeTTG2) did not do as well as either of the other two set-based methods, perhaps reflecting the limited amount of training data that we had available. Perhaps with cross-validation we might have done better than we did with a single held-out development test topic. Figure 4 shows the learning curve for our SVM classifier on Topic 3. The fact that the error on the training set stops decreasing as we add more training instances suggests underfitting,[8] which in turn suggests that additional work on model and feature engineering are called for.

From these results, we are now able to formulate some additional research questions, including:

- Currently, our novelty detection methods assume a relevance-ordered processing, but we might also try other orders (e.g., temporal).

- Because of the cascade design of the timeline generation, we should explore interaction effects with the design of our ad-hoc search. Ceiling analysis (using ground truth ad-hoc relevance judgments) shows that if we had a perfect ad-hoc search, our current best TTG system (hltcoeTTG3) could further improve weighted F_1 by 0.3777 absolute and unweighted F_1 by 0.3977 absolute. These fairly

[8] SVM parameter C and kernel function parameter $Sigma^2$ have already been tuned to optimal for this plot

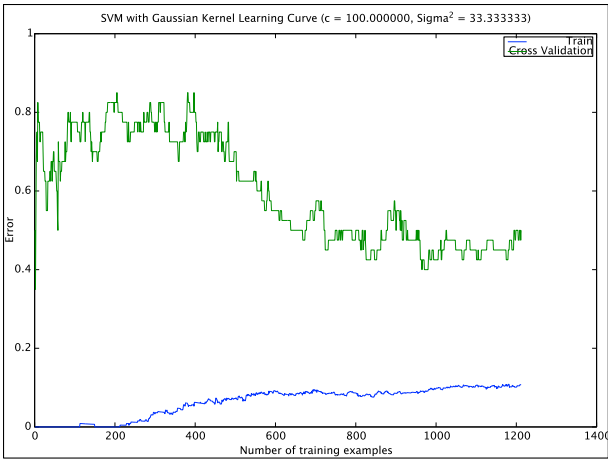

Figure 4: SVM Learning Curve on Training/Validation.

large potential improvements suggest that ad-hoc search is still a bottleneck, and thus worthy of further attention.

3 Clinical Decision Support

For this track we used the JHU HAIRCUT retrieval engine described by McNamee and Mayfield (2004). One of HAIRCUT's distinctive traits is the use of character n-grams as indexing terms, which have proven to be effective for controlling the effects of morphological variation (McNamee et al., 2009). While morphological variation is not nearly as substantial a problem in English as it is in some other languages, we thought it likely that the highly technical terminology prevalent in medical literature could benefit from n-gram indexing.

We had hoped to explore other techniques, including relevance feedback using "collection enrichment" from appropriate medical domain side corpora, however we did not have adequate time to conduct those experiments over the summer. Our submissions were produced in about 1.5 person-days of effort, with a significant portion of that time spent in preprocessing the PubMed Open Access Subset corpus.

We did not make any use of domain-specific resources such as ontologies, thesauri, or biomedical IE tools. We sought through our participation to determine the performance that a domain-agnostic, state-of-the-art retrieval engine might obtain.

Run	Topics	Terms	RF	Index
hltcoe5s	Summ	5-grams	None	Partial
hltcoe5srf	Summ	5-grams	Yes	Partial
hltcoe5drf	Desc	5-grams	Yes	Partial
hltcoewsrf	Summ	words	Yes	Complete

Table 5: Parameters for submitted Clinical Decision Support runs.

3.1 Submissions

We submitted four runs to the track, that varied in several ways:

- the type of tokenization used – either plain words or overlapping character 5-grams;
- whether pseudo-relevance feedback was used, or not;
- if "description" or "summary" topics fields were used; and,
- whether the full collection was indexed.[9]

The last variation, indexing of the entire collection, is due to an error that was only discovered just prior to submission. Due to the relatively small percentage of documents that were not indexed, we do not believe that the n-gram runs are too badly affected.

The parameters for each run are summarized in Table 5.

When relevance feedback was used, queries were modified after examining the top 20 and bottom 75 (of 1000) documents, and selecting either 60 (words) or 200 (5-grams) expansion terms.

3.2 Results

NIST provided results for each of our official runs, and we report average results over the 30 topics in Table 6. Metrics include inferred average precision (infAP), inferred normalized discounted cumulative gain (infNDCG), precision at the number of known relevant documents (R-prec), and precision at a fixed cutoff of 10 documents (P@10).

[9]For all runs docids 3016743 & 3261719 were not indexed due to parsing difficulties. Additionally, and more significantly, the 5-gram runs ran on a slightly "broken" index which accidentally left out 3.5% of the documents.

Run	infAP	infNDCG	R-prec	P@10
hltcoe5s	0.0562	0.2008	0.1739	0.3200
hltcoe5srf	**0.0812**	**0.2587**	**0.2193**	**0.3700**
hltcoe5drf	0.0677	0.2199	0.1834	0.3000
hltcoewsrf	0.0725	0.2412	0.2117	0.3533
median	0.0316	0.1514	0.1257	0.2333
oracle	0.1805	0.5197	0.3496	0.7100

Table 6: Averaged results for submitted Clinical Decision Support runs, compared to median performance and oracle best results by automatic runs.

3.3 Discussion

We make several observations from these preliminary results.

Summaries beat descriptions. Runs hltcoe5srf and hltcoe5drf differ only in whether description of summary topics were used. Performance in all metrics was notably higher using the Summary topics.

Relevance feedback boosts average performance. Relevance feedback led to substantial improvements, on both recall-sensitive and precision-focused metrics. Runs hltcoe5s and hltcoe5srf differ only in the use of relevance feedback, which led to average relative gains of 44% in infAP and 29% in infNDCG.

Character 5-grams outperformed words. Run hltcoe5srf beat hltcoewsf on all four metrics, however, while the gains were palpable, they were not dramatically different (*i.e.,* relative gains varied from 3.5% to 12%, depending on the metric).

In comparison with all automatic runs submitted to the track, our submitted runs were substantially above the median (58% to 157%, depending on the metric). However, results were far below (about half of) the hypothetical results that would be produced by a per-topic oracle combination derived from post hoc examination of all automated submissions.

4 Conclusions

In general we are pleased with this year's Microblog track, and in particular with the fact that we now will have a considerable amount of training data for the TTG task, which we see as a useful addition to the track. Our results in that track suggest several productive directions for future work, including: (1) term-weighting in the query expansion, (2) novelty detection features, and (3) novelty detection binary classification model.

In the first running of the Clinical Decision Support task, we attained high performing results that were considerably above the median of automatic runs, and we saw confirmation that established techniques, namely relevance feedback and use of character n-grams to address morphology are effective in this domain.

References

James Allan, Rahul Gupta, and Vikas Khandelwal. 2001. Temporal summaries of new topics. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 10–18, New York, NY, USA. ACM.

Ahmed Saad El Din and Walid Magdy. 2012. Web-based pseudo relevance feedback for microblog retrieval. In *TREC*.

Tarek El-Ganainy, Zhongyu Wei, Walid Magdy, and Wei Gao. 2013. Qcri at trec 2013 microblog track. In *TREC*.

Paul McNamee and James Mayfield. 2004. Character n-gram tokenization for european langauge text retrieval. *Information Retrieval*, 7(1-2).

Paul McNamee, Charles Nicholas, and James Mayfield. 2009. Addressing morphological variation in alphabetic languages. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 75–82. ACM.

Donald Metzler and Bruce W. Croft. 2007. Linear feature-based models for information retrieval. *Inf. Retr.*, 10(3):257–274, June.

Brendan O'Connor, Ramnath Balasubramanyan, Bryan R Routledge, and Noah A Smith. 2010. From tweets to polls: Linking text sentiment to public opinion time series. *ICWSM*, 11:122–129.

Ian Soboroff and Donna Harman. 2005. Novelty detection: The trec experience. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 105–112, Stroudsburg, PA, USA. Association for Computational Linguistics.

Justin Zobel and Alistair Moffat. 2006. Inverted files for text search engines. *ACM Comput. Surv.*, 38(2), July.