THE HLTCOE APPROACH TO THE TREC 2012 KBA TRACK

Brian Kjersten
HLTCOE
Johns Hopkins University

Paul McNamee
HLTCOE
Johns Hopkins University

Abstract

Our team submitted runs for the TREC KBA Cumulative Citation Recommendation task. This task involves labeling over 300 million documents for whether they are relevant and/or central to particular entities already in a database. For this task, we used an SVM classifier that uses unigrams and named entities as binary features. In this paper, we describe our work for the 2012 evaluation and the results we obtained.

1 Introduction

The goal of the TREC KBA track[1] is to help people edit and add information about named-entities to existing databases such as Wikipedia. The defined task for the first year of KBA is to identify whether documents include facts that are relevant to certain entities that already exist in a database. We were provided with a large collection of documents consisting of blogs and news articles from between October 7, 2011 to May 2, 2012. A subset of 15,815 of the 2011 documents are labeled for whether or not they are relevant or central to any of 29 entities.

This task is interesting in a number of ways. One is the collection which contains over 300 million documents which contain fine-grained (i.e., sub-hour) timestamps for a period of 8 consecutive months. Another is that some of the entities of interest were selected to be especially difficult to disambiguate. Another interesting feature is that there is an option of labeling documents for whether they are central to an entity, rather than merely mentioning the entity. We opted to focus on labeling centrality rather than relevance.

2 Entity Search as Topic Classification

The HLTCOE team approached the task by thinking of relevance and centrality labeling as a classical document classification task such as was studied at the TREC Filltering Track (Robertson and Soboroff, 2002). Support Vector Machines (SVMs) are known to be well-performing text classifiers, and they support high-dimension representations such as bags of words. We trained a linear SVM classifier for each of the 29 entities of interest. To generate a model for each entity, we used all of the the labeled training data from the time period October 7 to December 31, 2011. If a document was labeled as "central" for an entity we included it as a positive training example for the SVM. If a document was not labeled as "central" (for example, it had a label of "relevant" or "junk"), we included it as a negative training example.

The documents that were pre-labeled for training were selected because they were the ones that were most likely to be about the entities. Therefore, we were able to assume that most of the documents that were not labeled are not central to any of the entities. In addition to the labeled examples, we added 11,007 unlabeled documents randomly selected from the same 2011 time span to use as additional, presumptive negative training examples. This served as a generic source of negative documents that are likely not topically close to the entities of interest. We also want a large source of negative documents to help the classifier avoid focusing on

[1] `http://trec-kba.org/`

that words that are particular to the training epoch (e.g., *Thanksgiving* or *December*). The same 11,007 presumed negative examples were used for each entity.

As preprocessing, we downcased words and removed punctuation within words. This meant that the word “can’t” was normalized to “cant” and “high-performing” was normalized to “highperforming”. We removed a set of 319 stopwords. To filter out non-English documents, we ignored any word that contained a character that was neither ASCII nor part of the Latin subset of Unicode. We also ignored all documents that contained fewer than 6 word types from our stop word list. This was intended as a means to avoid non-English documents, and it had the added benefit of skipping very short English documents.

All of our SVM experiments were performed with the SVM Light toolkit (Joachims, 1998). We exclusively used binary features, representing whether or not a term was present in a document, regardless of its frequency. The features we used were the unigrams in each article and the named entities labeled in each document; we used the automatically-labeled named entities that were provided as annotations with the corpus. Each of these features was assigned an integer-valued feature number f in the order it was seen.

This leads to 61,735,102 unigram features and 59,534,101 entity name features, for a total of 121,269,203 features. Since out of the box, the SVM Light tool can handle only 100 million features, we needed to map these to a smaller feature set. For features that were seen after the first 90 million, we reassigned their feature numbers using an multiplicative hashing method.

The hashing algorithm we used was to multiply the feature number by an irrational number (we chose $1/\sqrt(\pi))$, subtract the next lowest integer, and map it to the range of 10 million to 90 million to get a new feature number:

$$\lambda = f\frac{1}{\sqrt(\pi)} - \lfloor\frac{1}{\sqrt(\pi)}\rfloor$$

$$f_{new} = \lambda 90,000,000 + (1 - \lambda)10,000,000$$

This reduces the number of unique features by creating collisions between existing features. We protect the first 10 million features from collision, and some of the remaining, rarer features will have conflations, which we thought unlikely to have a significant deleterious effect.[2]

The test data was the corpus from January 1 to May 2, 2012. At test time, we ran `svm_classify` on all of the evaluation data for all of our models (i.e., one model for each entity). SVM Light produces confidence labels, which are less than 0 if a document is given a negative label, and are greater than 0 if a document is given a positive label. We inspected predictions on training data (i.e., data before Jan. 1), and we observed that the majority of these scores lie between -1.0 and 1.0. To convert these to the [0, 1000] range that is required for the TREC KBA task, we multiplied by 500 and added 500. If a document had a score greater than 1000, we forced its score to 1000. If a document had a score less than 50, we did not include it in the submission. This ensures that a neutral document is mapped to 500, and documents with scores greater than one or higher are mapped to 1000.

To produce runs we ran computing jobs for different evaluation epochs in parallel using the Sun Grid Engine (SGE) platform.

We submitted two runs: *hltcoe-wordNER* and *hltcoe-wordNER500*. The later is simply a filtering of the first run, which removed documents with scores below 500 (“neutral”).[3]

3 Discussion

We first examine the distribution of output scores (Figure 1). Remembering that a score of 500 is neutral, and higher scores are positive, we can observe that very few documents are labeled as positive. This is as expected, because most documents in the vast collection are not about the entities we are interested in. More interestingly, there is a smooth drop-off in the proportions of documents with scores as the scores increase. This will affect the way we interpret our other results.

[2] One of the authors was introduced to hashing termids in this fashion by Chris Buckley, who used a hybrid hashed/unhashed dictionary for the Terabyte track in TREC 2004.(Buckley, 2005)

[3] We did not intend to imply a ranking of documents in *hltcoe-wordNER500*, however, we neglected to transform all scores in this run to a fixed value, which would have made this explicit.

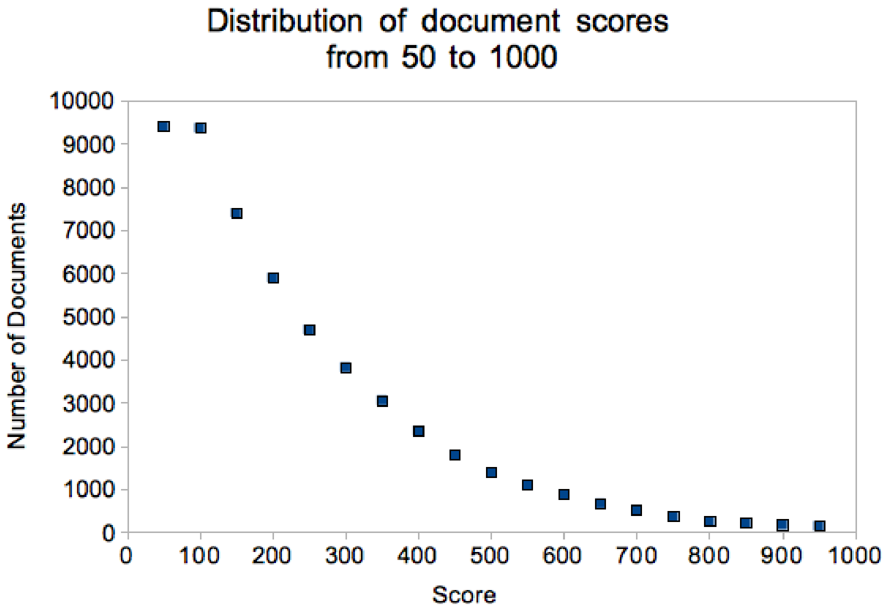

Figure 1: The distribution of document scores produced by our SVM system. Since most documents are not central, few documents earn a score of at least 500.

To see how our system performs using traditional metrics, see Figure 2. The figure shows the arithmetic mean of Precision, Recall, F-Score, and Scaled Utility across the 29 entities as cutoff value is adjusted. Surprisingly, Precision peaks near a cutoff value of 500, which corresponds to the boundary between the positive label and the negative label. This might be because it is unusual for a document to obtain very high scores, and high-scoring documents have greater variance due to sparsity. Interestingly, the F-Score continues to grow considerably after Precision drops off, suggesting that if Recall is just as important as Precision, we should use cutoff values much lower than 500.

While there are insights to be gained by using the cutoff value as the x-axis, in an IR context it is also informative to look at the ranks of the documents themselves. Figure 3 shows the same performance metrics from the perspective of rank. In this graph, the x-axis is shifted in proportion to how numerous documents are at each score level. Here, it is clear that Precision peaks early on, indicating that the true positives are some of the highest scoring.

An analysis of the scores for each individual entity is shown in Table 1. This shows the scores at the cutoff value which maximizes the F-Score. For most entities, the F-Score is maximized near a cutoff of 50 or 100. This is because of the high recall, as noted above. Some entities have a corresponding F-Score of 0, because none of the relevant documents appeared with a score above 50. It is not clear whether this method has particular difficulty when two entities share a name. None of the Boris Berezovsky (pianist) documents were correctly labeled, but Basic Element (company) was the best performing.

4 Conclusions

We have demonstrated that a Support Vector Machine classifier using bag-of-words and bag-of-entity-names is a tractable method to label very large collections of documents as central to an entity or not, and its performance is competitive with other approaches to the TREC KBA track. There are many ways that this approach can be extended in the future, including additional features based on syntactic relations or relationships between document entities. It might also be informative to compare SVMs with other standard approaches. We also envisioned dynamically adapting classifiers over time, which

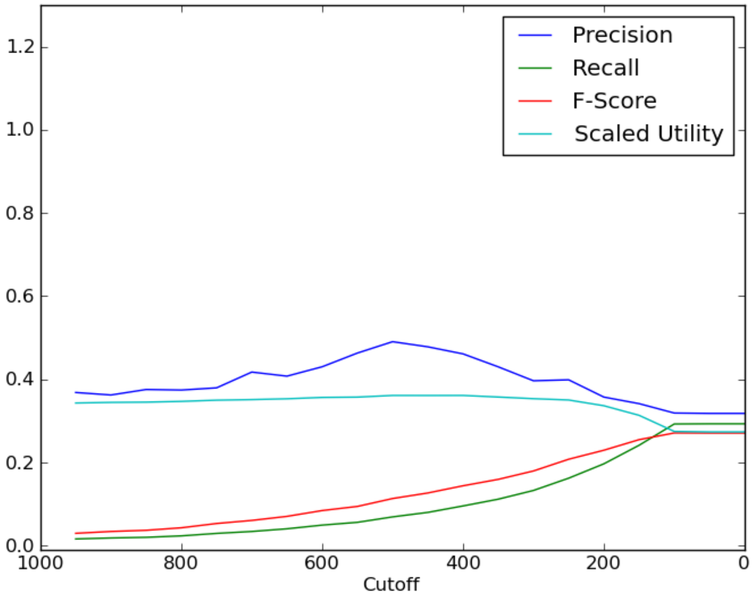

Figure 2: The performance of our system (*hltcoe-wordNER*) as a function of cutoff value. Note that Precision peaks near a cutoff=500, which corresponds to the boundary between a positive label and a negative label according to the support vector machine.

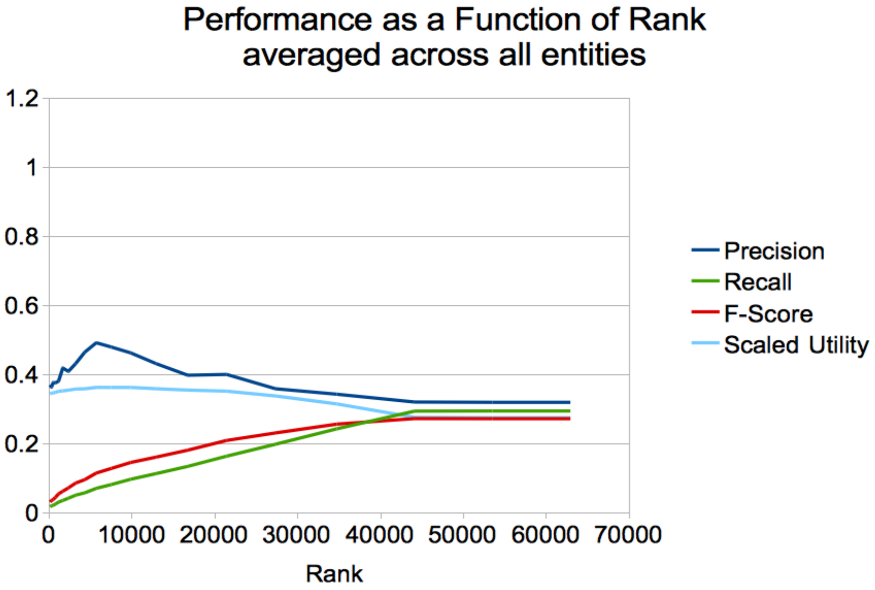

Figure 3: Since most documents were negative (and the negative class prior is quite large), the SVM produced few documents scores above 500. For this reason, it is useful to view the performance with score ranks along the x-axis. Here it is clear that the Precision is maximized for high-ranking documents, but that the F-Score continues to grow as Recall increases.

Entity	Cutoff	Precision	Recall	F-Score	Scaled Utility
Aharon Barak	100	0.27	0.13	0.17	0.31
Alex Kapranos	100	0.23	0.2	0.21	0.24
Alexander McCall Smith	100	0.16	0.45	0.24	0
Annie Laurie Gaylor	100	0.49	0.33	0.39	0.44
Basic Element (company)	200	0.88	0.5	0.64	0.64
Basic Element (music group)	100	0.95	0.22	0.36	0.48
Bill Coen	100	0.33	0.04	0.07	0.33
Boris Berezovsky (businessman)	100	0.33	0.27	0.3	0.33
Boris Berezovsky (pianist)	950	0	0	0	0.33
Charlie Savage	400	0.61	0.23	0.33	0.44
Darren Rowse	150	0.18	0.35	0.24	0.04
Douglas Carswell	100	0.13	0.32	0.19	0
Frederick M. Lawrence	950	0	0	0	0.33
Ikuhisa Minowa	50	0.45	0.66	0.53	0.5
James McCartney	50	0.14	0.3	0.19	0
Jim Steyer	100	0.72	0.39	0.51	0.55
Lisa Bloom	250	0.48	0.22	0.3	0.4
Lovebug Starski	150	0.33	0.11	0.17	0.33
Mario Garnero	100	0.96	0.36	0.53	0.57
Masaru Emoto	150	0.33	0.14	0.2	0.33
Nassim Nicholas Taleb	100	0.4	0.51	0.45	0.42
Rodrigo Pimentel	250	0.75	0.23	0.35	0.46
Roustam Tariko	100	0.33	0.41	0.36	0.33
Ruth Rendell	50	0.32	0.29	0.3	0.32
Satoshi Ishii	100	0.59	0.39	0.47	0.5
Vladimir Potanin	250	0.62	0.25	0.36	0.45
William Cohen	100	0.1	0.14	0.12	0
William D. Cohan	250	0.47	0.49	0.48	0.48
William H. Gates, Sr	100	0.1	0.04	0.05	0.25
Macro-average	0	0.32	0.29	0.27	0.27

Table 1: The cutoff values that maximize F-Score for each entity, and the corresponding scores.

might improve performance as some entities appear in the document stream in bursts due to discrete events, however, time did not permit exploration of adaptive filtering on this data.

References

Chris Buckley. 2005. Looking at Limits and Tradeoffs: Sabir Research at TREC 2005. In *Proceedings of the 14th Text Retrieval Conference (TREC 2005)*.

T. Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European Conference on Machine Learning (ECML)*, pages 137–142, Berlin. Springer.

Stephen Robertson and Ian Soboroff. 2002. The TREC 2002 Filtering Track Report. In *Proceedings of the 11th Text Retrieval Conference (TREC 2002)*.