Filtering Documents over Time for Evolving Topics – The University of Amsterdam at TREC 2013 KBA CCR

Tom Kenter
ISLA, University of Amsterdam
Science Park 904, 1098 XH Amsterdam, The Netherlands
tom.kenter@uva.nl

Abstract

In this paper we describe the University of Amsterdam's approach to the TREC 2013 Knowledge Base Acceleration (KBA) Cumulative Citation Recommendation (CCR) track. The task is to filter a stream of documents for documents relevant to a given set of entities. We model the task as a multi-class classification task. Entities may evolve over time and the classifier should be able to adapt to these changes at runtime. To achieve this, the classifier performs online self-learning, i.e., learning only from the examples it is most confident about, based on a confidence score it produces for every prediction it makes.

1 Introduction

This year's TREC KBA Cumulative Citation Recommendation (CCR) task is described as follows: "Given a fixed list of target entities from Wikipedia and Twitter, filter documents worth citing in a profile of the entity, e.g., their Wikipedia or Freebase article. CCR has no requirement for novelty or salience."[1] A 6.45TB corpus of documents is provided by the TREC organizers, which is the next generation of the KBA stream corpus 2012 (see (Frank et al., 2013)). The documents can be thought of as arriving in a stream over time. When the time span during which the stream of documents is being monitored is considerable the topics under consideration can be expected to evolve over time. A document filtering system should be sensitive to these changes and should be able to adapt over time.

We model the CCR task as a multi-class classification task where the entities being monitored constitute the classes. We only use the training data provided by TREC in the streaming corpus. No Wikipedia profile pages or any other external data sources are considered.

Dealing with evolving topics calls for an online learning approach where the document filtering system can adapt while it is running. As no relevance feedback is available after the initial training phase (i.e., after the time cutoff set by the TREC organizers) the scenario lends itself particularly well to a self-learning approach where the classifier learns from the documents it observes at run time. This approach was also tested on the TREC 2012 KBA data, see (Berendsen et al., 2012).

In what follows we describe the system we used to classify documents in §2, and the setup we used for generating runs in §3. The results are described in §4 and we conclude in §5.

2 Our approach

We model the CCR task as a multi-class classification task where the entities being monitored constitute the classes. The only training material considered are the assessed documents in the streaming corpus. No external sources (like, e.g., Wikipedia entity profile pages) are taken into account.

We use a multinomial Naive Bayes classifier with feature selection based on time-aware χ^2 as described in (Kenter et al., 2013). The time-aware χ^2 metric is calculated over batches of documents,

[1] http://trec-kba.org/trec-kba-2013.shtml

in a streaming setting with documents arriving over time. Once the maximum batch size is reached, χ^2 values are computed for every feature, for every class, over the batches. This is done either from a weighted sum over the concurrency tables of the most recent batches or from a weighted sum over the χ^2 scores of the latest batches. The batches are weighted over time by a decay function (e.g., applying higher weights to more recent documents). The top n features based on this measure are selected. This yields a restricted feature space for the classifier to base its decision on. By producing a new feature selection every time the maximum batch size is reached, the classifier is able to adapt over time.

Confidence scores are calculated for every prediction the classifier makes. When an example is classified as being relevant (relevance level 2) and its confidence score exceeds a certain threshold level, the classifier updates its internals based on the example. This is an online learning step (or *self-learning* step) that allows the classifier to adapt to any changes happening to the entity. We applied several threshold values for the different runs (see Table 1). In the runs that we submitted the scores were normalized and mapped onto the (0, 1000] integer scale.

As detailed in §3.3 we first filter the documents for entity mentions. The actual classification of a document is a two-step process. First, a document is discarded as being irrelevant (relevance level 0) if no name mention was found. Second, the documents that do have name mentions are classified by the classifier. Their relevance level is set to 2 if the class predicted by the classifier coincides with any of the name mentions. If not, i.e., the classifier predicts a class no name mention was found for in the document, the relevance level is deemed to be 0 (irrelevant).

3 Experimental setup

3.1 Data

For our experiments we use the 'English and unknown' subset[2] of the KBA stream corpus. We do not consider all entities in our experiments, taking only entities into account for which 'vital' annotations were provided. This results in a subset of 101 entities, out of the total set of 141 entities.

A part of the corpus is already pre-processed by the TREC organizers.[3] The output of the pre-processing is stored in serialized files with several data fields per document. We consider only those documents for which a non-empty 'clean_visible' field is available.

3.2 Pre-processing

For pre-processing we used a 90 node Hadoop cluster. Pre-processing consists of filtering files containing mentions of the entities of interest. As noted above we only consider documents in the corpus that are tokenized already. We filter the documents for entity mentions and apply additional tokenization.

In the next sections we will describe the pre-processing steps in more detail.

3.2.1 Entity mentions

For the Wikipedia entities we obtain alternative names and spellings from a DBPedia dump of August 31, 2011, from fields like 'nickname,' 'alias,' 'alternativenames,' 'birthname.' Some simple heuristics are applied to eliminate duplicates, and useless (for this step) phrases like 'Jr.' and 'Sr.' For the Twitter entities we manually examine the Twitter profiles for alternate names, as suggested by the TREC organizers on the TREC-KBA Google Group.[4]

As noted in §2 we only consider documents with entity mentions for classification. Documents in which no entity mention is found, i.e., the vast majority of the corpus, are deemed irrelevant.

3.2.2 Tokenization

We use the tokens provided in the serialized files in the stream corpus, but perform an additional cleaning step to delete leading and trailing non-ASCII non-token characters such as quotes, brackets and ellipses.

[2] In full: kba-streamcorpus-2013-v0_2_0-english-and-unknown-language

[3] http://trec-kba.org/kba-stream-corpus-2013.shtml

[4] "Since Twitter does not offer a name-expansion API, it is acceptable to manually examine the twitter profile page to identify alternate names for these entities. This is still considered "run_type:" "automatic," because a human entering this entity as a query could easily be asked to examine the twitter profile page (and no other texts)." https://groups.google.com/forum/#!msg/trec-kba/utOe7Lz1RZ0/t9--G1zf_SMJ

A list of 143 common stop word tokens is filtered out, as are tokens consisting of only digits and/or only non-word characters. After these steps, the documents together contain 1,825,422 unique tokens.

Lastly, we omit tokens appearing 2 times or less in the corpus. Our learning algorithm monitors word occurrences to select features that appear more frequently for a certain class over time. Words occurring only once or twice in the entire corpus have no chance of attributing anything in this respect. Hence, we omit them, which yields 588,693 features in total.

3.2.3 Train and test set

Selecting documents in the corpus as described in the previous subsections yields 1,801 documents before the time cutoff (i.e., training examples) and 66,118 examples after the cutoff (the test examples).

3.2.4 Feature selection

As we employ feature selection using the time-aware χ^2 metric, the number of features to be selected has to be decided on. We select 5 features per class (so 505 features at most) as this setting proved to yield the best results on the TREC 2012 KBA data.

3.3 Experiments

Table 1 lists the parameter settings for the runs we submitted.

The runs starting with 'bsln' or 'bl' are non-adaptive runs in the sense that no features were re-selected after the time cutoff. However, the probabilities for the features were updated during the run.

For the other runs, the most salient features were selected again for every class, at the end of each time batch. The runs differ in parameters settings.

As a reference we included a run without any classification, called 'uva_kba_run_av'. If a mention was found for an entity, the file was considered to be vital in this run.

We submitted 26 runs in total. For convenient comparison between the results, every run lists all the entries of the 66,118 document set.

4 Results

In Table 2 the results for each run are listed, based on the ground truth file of September 26, 2013, expanded with ssf inferred vitals, without documents that have an empty (or no) `clean_visible` attribute.

This ground truth file includes documents for the full set of 141 entities. It is important to note, however, as previously mentioned in §3.3, that the classifier we use in our experiments is trained only on documents for a subsection of 101 entities for which training examples are provided. Furthermore, only 'vital' documents are used as training examples.

To examine the performance of our classifier for the task it was trained for, we perform an additional evaluation on a subset of the ground truth data, in which we included only documents relevant to the set of 101 entities. The results of this evaluation are listed in Table 3.

The results in Table 3 improve over the results in Table 2. This is not suprising as in Table 3 the classifier is evaluated only on the entities it was trained on. As we can see, the performance is similar to the top TREC results for some runs in terms of micro F1, micro SU and macro SU. The reported scores would be among the top 10 results for those metrics. We note that this is only indicative. A true comparison can not be made, because, as noted earlier, the figures of top TREC results are based on different ground truth data than the ones reported in Table 3.

Interestingly, there are adaptive, self-learning runs that have the same or higher performance compared to the non-adaptive baseline runs on some metrics. This shows the potential of our approach. The absence of a consistent pattern, however, also shows the difficulty of finding the right settings.

5 Conclusion

In this paper we describe the UvA approach to the TREC 2013 KBA CCR task. We model the task as a multi-class classification task. We detail the preprocessing steps we applied to the documents in the corpus provided by the TREC organizers. A multinomial Naive Bayes classifier is used for the experiments, which is able to adapt to changes in the classes it monitors over time by selecting features based on time-aware χ^2 (Kenter et al., 2013). The classifier is self-learning; predictions with a confi-

Table 1: Parameter settings per run. Columns are: run identifier, decay function, the way batches are aggregated to compute χ^2, number of batches, maximum number of examples per batch, confidence score, threshold for the confidence score.

run id	decay	aggreg. of χ^2	# of b.	# per b.	conf. score	threshold
bsln_5_100_100	flat	weightedConcs	100	100		
bl_na_wChis_c1	linear	weightedChis	100	100	confidence1	-100
bl_na_wChis_c3	linear	weightedChis	100	100	confidence3	-100
bl_na_wConcs_c1	linear	weightedConcs	100	100	confidence1	-100
bl_na_wConcs_c3	linear	weightedConcs	100	100	confidence3	-100
uva_kba_run_1	linear	weightedConcs	10	500	confidence1	-200
uva_kba_run_2	linear	weightedConcs	40	70	confidence1	-200
uva_kba_run_3	linear	weightedConcs	10	500	confidence3	-200
uva_kba_run_4	linear	weightedConcs	40	70	confidence3	-200
uva_kba_run_5	linear	weightedChis	10	500	confidence1	-200
uva_kba_run_6	linear	weightedChis	40	70	confidence1	-200
uva_kba_run_7	linear	weightedChis	10	500	confidence3	-200
uva_kba_run_8	linear	weightedChis	40	70	confidence3	-200
uva_kba_run_9	linear	weightedChis	80	50	confidence1	-75
uva_kba_run_10	linear	weightedChis	80	70	confidence1	-75
uva_kba_run_11	linear	weightedChis	80	50	confidence3	-75
uva_kba_run_12	linear	weightedChis	80	70	confidence3	-75
uva_kba_run_13	flat	weightedChis	80	50	confidence1	-75
uva_kba_run_14	flat	weightedChis	80	70	confidence1	-75
uva_kba_run_15	flat	weightedChis	80	50	confidence3	-75
uva_kba_run_16	flat	weightedChis	80	70	confidence3	-75
uva_run_wChi_c1	linear	weightedChis	100	100	confidence1	-100
uva_run_wChi_c3	linear	weightedChis	100	100	confidence3	-100
uva_run_wCon_c1	linear	weightedConcs	100	100	confidence1	-100
uva_run_wCon_c3	linear	weightedConcs	100	100	confidence3	-100
uva_kba_run_av	*every document containing a mention is considered to be vital.*					

dence score above a pre-set threshold are used as additional training material. We perform experiments for different parameter settings of the classifier. When evaluated on the entities it was trained on, top ten performance (compared to the top TREC results) is observed in some settings on some metrics.

Acknowledgments This work was carried out on the Dutch national e-infrastructure with the support of SURF Foundation.

This research was supported by the Netherlands Organisation for Scientific Research (NWO) under project nr HOR-11-10.

References

Richard Berendsen, Edgar Meij, Daan Odijk, Wouter Weerkamp, and Maarten de Rijke. 2012. The University of Amsterdam at TREC 2012. In *TREC '12*. NIST.

John R. Frank, Max Kleiman-Weiner, Daniel A. Roberts, Feng Niu, Ce Zhang, Christopher Ré, and Ian Soboroff. 2013. Building an entity-centric stream filtering test collection for TREC 2012. In *TREC '12*.

Tom Kenter, David Graus, Edgar Meij, and Maarten de Rijke. 2013. Time-aware chi-squared for document filtering over time. In *Proceedings of the SIGIR 2013 Workshop on Time-aware Information Access*.

Table 2: Results per run based on the ground truth file of 2013-09-26, expanded with ssf inferred vitals, with a non-empty `clean_visible` attribute. The highest values per metric for our runs are displayed in boldface. The reported TREC maximum values are for uiucGSLIS bayes02 (micro F1), BIT RFClassStrict (micro SU), BIT RFClassStrict (macro F1) and uiucGSLIS bayes08 (macro SU), BIT RFBurst_1 (weighted F1) and BIT RFBurst (weighted SU).

run id	micro F1	micro SU	macro F1	macro SU	weighted F1	weighted SU
TREC maximum	0.324	0.401	0.311	0.277	0.002	0.002
bsln_5_100_100	0.294	0.333	0.161	**0.255**	**0.001**	**0.002**
bl_na_wChis_c1	0.280	0.333	**0.164**	**0.255**	**0.001**	0.001
bl_na_wChis_c3	0.280	0.333	**0.164**	**0.255**	**0.001**	0.001
bl_na_wConcs_c1	0.304	0.330	0.160	0.250	**0.001**	**0.002**
bl_na_wConcs_c3	**0.305**	0.333	0.160	**0.255**	**0.001**	**0.002**
uva_kba_run_1	0.233	0.305	0.098	0.232	**0.001**	0.001
uva_kba_run_2	0.211	0.332	0.072	0.240	**0.001**	0.001
uva_kba_run_3	0.219	0.314	0.097	0.238	**0.001**	0.001
uva_kba_run_4	0.220	0.331	0.078	0.239	**0.001**	0.001
uva_kba_run_5	0.225	0.299	0.102	0.234	**0.001**	0.001
uva_kba_run_6	0.204	0.326	0.086	0.247	**0.001**	0.001
uva_kba_run_7	0.228	0.301	0.103	0.237	**0.001**	0.001
uva_kba_run_8	0.256	0.331	0.091	0.252	**0.001**	0.001
uva_kba_run_9	0.108	0.310	0.050	0.252	0.000	0.001
uva_kba_run_10	0.186	0.340	0.071	0.253	**0.001**	0.001
uva_kba_run_11	0.106	0.333	0.052	**0.255**	0.000	0.001
uva_kba_run_12	0.162	**0.348**	0.062	**0.255**	**0.001**	0.001
uva_kba_run_13	0.147	0.316	0.060	0.251	**0.001**	0.001
uva_kba_run_14	0.162	0.347	0.063	0.253	**0.001**	0.001
uva_kba_run_15	0.121	0.333	0.058	**0.255**	0.000	0.001
uva_kba_run_16	0.161	0.347	0.070	**0.255**	**0.001**	0.001
uva_run_wChi_c1	0.136	0.338	0.054	0.253	0.000	0.001
uva_run_wChi_c3	0.136	0.338	0.054	**0.255**	0.000	0.001
uva_run_wCon_c3	0.232	0.334	0.127	**0.255**	**0.001**	0.001
uva_run_wCon_c1	0.217	0.329	0.111	0.253	**0.001**	0.001
uva_kba_run_av	0.186	0.340	0.071	0.228	**0.001**	0.001

Table 3: Results based on the same ground truth data used for for Table 2, only filtered for entities for which training material was available. The highest values per metric are displayed in boldface.

run id	micro F1	micro SU	macro F1	macro SU	weighted F1	weighted SU
bsln_5_100_100	0.308	0.333	0.225	**0.274**	**0.002**	**0.002**
bl_na_wChis_c1	0.293	0.333	**0.228**	**0.274**	**0.002**	**0.002**
bl_na_wChis_c3	0.293	0.333	**0.228**	**0.274**	**0.002**	**0.002**
bl_na_wConcs_c1	0.320	0.329	0.224	0.267	**0.002**	**0.002**
bl_na_wConcs_c3	**0.321**	0.333	0.224	**0.274**	**0.002**	**0.002**
uva_kba_run_1	0.248	0.300	0.136	0.241	**0.002**	**0.002**
uva_kba_run_2	0.227	0.332	0.100	0.253	0.001	**0.002**
uva_kba_run_3	0.231	0.310	0.135	0.250	**0.002**	**0.002**
uva_kba_run_4	0.236	0.331	0.109	0.252	0.001	**0.002**
uva_kba_run_5	0.239	0.293	0.143	0.245	0.001	**0.002**
uva_kba_run_6	0.217	0.325	0.120	0.263	0.001	**0.002**
uva_kba_run_7	0.241	0.296	0.144	0.248	**0.002**	**0.002**
uva_kba_run_8	0.278	0.331	0.127	0.269	0.001	**0.002**
uva_kba_run_9	0.121	0.306	0.069	0.269	0.001	**0.002**
uva_kba_run_10	0.208	0.341	0.099	0.271	0.001	**0.002**
uva_kba_run_11	0.119	0.333	0.073	**0.274**	0.001	**0.002**
uva_kba_run_12	0.182	**0.350**	0.086	**0.274**	0.001	**0.002**
uva_kba_run_13	0.164	0.313	0.083	0.268	0.001	**0.002**
uva_kba_run_14	0.182	0.349	0.087	0.271	0.001	**0.002**
uva_kba_run_15	0.135	0.333	0.081	**0.274**	0.001	**0.002**
uva_kba_run_16	0.181	0.349	0.098	**0.274**	0.001	**0.002**
uva_run_wChi_c1	0.153	0.339	0.075	0.271	0.001	**0.002**
uva_run_wChi_c3	0.153	0.339	0.075	**0.274**	0.001	**0.002**
uva_run_wCon_c1	0.228	0.328	0.155	0.270	**0.002**	**0.002**
uva_run_wCon_c3	0.244	0.334	0.177	**0.274**	**0.002**	**0.002**
uva_kba_run_av	0.208	0.341	0.099	0.236	0.001	**0.002**