Time-Aware Chi-squared for Document Filtering over Time

Kenter, T.M.; Graus, D.P.; Meij, E.J.; de Rijke, M.

Published in:
SIGIR 2013 Workshop on Time-aware Information Access: #TAIA2013: August 1, 2013

SIGIR 2013 Workshop on Time-aware Information Access (#TAIA2013)

August 1 , 2013

Time-Aware Chi-squared for Document Filtering over Time

Tom Kenter
ISLA, University of Amsterdam
Amsterdam
tom.kenter@uva.nl

David Graus
ISLA, University of Amsterdam
Amsterdam
d.p.graus@uva.nl

Edgar Meij
Yahoo! Research
Barcelona
emeij@yahoo-inc.com

Maarten de Rijke
ISLA, University of Amsterdam
Amsterdam
derijke@uva.nl

ABSTRACT

Document filtering over time is applied in tasks such as tracking topics in online news or social media. We consider it a classification task, where topics of interest correspond to classes, and the feature space consists of the words associated to each class. In streaming settings the set of words associated with a concept may change. In this paper we employ a multinomial Naive Bayes classifier and perform periodic feature selection to adapt to evolving topics. We propose two ways of employing Pearson's χ^2 test for feature selection and demonstrate their benefit on the TREC KBA 2012 data set. By incorporating a time-dependent function in our equations for χ^2 we provide an elegant method for applying different weighting and windowing schemes. Experiments show improvements of our approach over a non-adaptive baseline, in a realistic settings with limited amounts of training data.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.5.4 [**Applications**]: [Text processing]

1. INTRODUCTION

Traditional ad hoc information retrieval systems focus on producing a ranked list of documents relevant to a user's query. Users might, however, persist in being interested in a concept and might want to track it over time. In this paper we propose a method for filtering a stream of documents for the ones relevant to a certain topic. The task is modeled as a multi-class classification task. Topics may evolve over time, making a classical approach of training a classifier on a set of examples and keeping it fixed at testing time unsuitable. We present a topic filtering system that is capable of adapting to drift in topics by periodically selecting features for a multinomial Naive Bayes classifier. Two versions of Pearson's χ^2 test are proposed for feature selection. Both incorporate the notion of time, which is not considered in the original formula. We show that both modifications improve over a non-adaptive baseline.

TAIA'13 August 1, 2013, Dublin, Ireland.

2. RELATED WORK

An extensive body of research has been published about topic tracking, from the seminal work by Allan [1] through recent tracks in TREC KBA [2].

The effectiveness of using χ^2 for selecting features has been noted before, e.g., by Yiming and Pedersen [6]. Kim and Chang [4] use χ^2 in a multinomial Naive Bayes classifier setting for selecting features and for weighting (boosting) them. When incrementally updating feature weights they use a supervised approach (i.e., the class a training example belongs to is always given). Closest to our approach is [3] in which a dynamic feature space is employed by using χ^2 for feature selection for a Naive Bayes classifier. Weka's implementation of χ^2 is augmented to allow for incremental updates, but no implementation details are provided. The weighting of examples they describe might be similar to our approach using Equation 2 and uniform and other decay functions (see Section 3) though they weigh documents individually rather than in batches as we do. They use the 20 Newsgroups corpus as well as the SpamAssasin corpus and extract training sets of 600 and 932 examples, respectively. Updates of their models are based on ground truth classes.

In [5] SVMs are used for adaptive window management. The authors use an exponential decay function like the one presented here (Section 3) to weigh examples rather than to select features. The authors observe that adapting time windows and batch selection works better than weighting batches. In other words, it is better to either keep or discard examples, rather than to apply weights to them.

In the literature, a common approach is to allow for adaptation based on the gold standard labels [3, 4]. In a real life scenario this setting does not apply as the 'real' labels will not be available. However, this scenario does provide an upper bound on performance: how well can an adaptive approach perform in the most ideal circumstances?

Our work differs from the work described above in the following important ways. Firstly we present two ways of calculating χ^2 over time. Secondly, by incorporating a time-dependent function in our equations for χ^2 we provide compact and elegant methods for applying different weighting and windowing schemes. Thirdly, we apply our algorithms

in a realistic setting where very little training material is available. And, most critically, we use a setting where the algorithms learn from their own predicted labels, rather than the gold standard ones.

3. OUR APPROACH

Pearson's χ^2 test has proven to be a robust measure for feature selection [3, 4, 6]. It is usually explained using a contingency table:

	w	$\neg w$	
c	a	c	C
$\neg c$	b	d	$\neg C$
	W	$\neg W$	N

Here, c is a concept, represented by a class in the present case, and w is an word. Value a is the number of times c and w co-occur, i.e., how many documents in class c contain w. Similarly, b is the number of documents in which w occurs which are not in c. W is the sum of a and b and N is the summation over all values $(a + b + c + d)$. The default equation for calculating χ^2 for a certain word and class then is:

$$\chi^2 = \frac{N(ad - bc)^2}{C \cdot \neg C \cdot \neg W \cdot W}. \quad (1)$$

As is evident from Equation 1, time plays no role in this formula. In what follows we describe two ways of incorporating the notion of time in this equation. We assume that there is a stream of documents arriving over time. Sequential documents are accumulated in batches, which we will refer to as *time buckets*.

3.1 χ^2 from weighted contingencies over time

We can take a weighted sum over the contingency tables of subsequent time buckets and calculate a χ^2 measure at time T from the result for every word in every class:

$$\chi^2_T = \frac{N'(a'd' - b'c')^2}{C' \cdot \neg C' \cdot \neg W' \cdot W'}, \quad (2)$$

where $X' = \sum_{t=0}^{T} f(T-t) \cdot X_t$, where X_t is the quantity of X in time bucket t. The function $f(\cdot)$ can be used as a time windowing or decay function (see below).

3.2 χ^2 from weighted χ^2 values over time

Alternatively, we can calculate a χ^2 measure for every time bucket, for all words and classes, and take a weighted sum over those:

$$\chi^2_T = \sum_{t=0}^{T} f(T-t) \cdot \chi^2_t, \quad (3)$$

where χ^2_t is the χ^2 measure for time bucket t and function $f(\cdot)$ is again a weighting or decay function.

3.3 Weighting or decay function

Different implementations of $f(\cdot)$ yield different variants of Equation 2 and 3 and hence different properties of the resulting classifier.

For the standard χ^2-measure (1) computed across the entire corpus one can simply apply $f(t) = 1$ to Equation 2. For a sliding window of length L we use the following *flat* decay function: $f(t) = 1$ if $t \leq L$, and 0 otherwise. For a *linear* decay over a window with length L: $f(t) = \max(0, 1 - t/L)$. For *exponential* decay we can apply $f(t) = e^{-t}$.

Many other weighting functions exist, including ones that allow for recurrent weighting for topics that appear periodically. The focus of this paper, however, is not to perform an in-depth investigation of the effect of applying different types of weighting functions. In our experiments we use a flat, linear and exponential weighting function, because they are simple, intuitive and commonly applied.

3.4 Adaptation

Adaptation occurs at two levels: updating the probabilities for the Naive Bayes classifier and selecting new feature sets per class. Since updating the probabilities is a cheap operation we do so after every example.

Feature selection is performed at the end of every time bucket, as it is based on χ^2 and hence requires more examples (this applies in particular when Equation 3 is used). Moreover, it is a more expensive operation. Based either on Equation 2 or 3 we take the top n features per class. These features together comprise the vocabulary used for the multinomial Naive Bayes classifier.

Intuitively, the difference between the two approaches is that Equation 2 captures saliency of a term for a certain class (expressed by its χ^2 value) regardless of its salience at every time point, but based on all the past (weighted) data. Equation 3, on the other hand, keeps track of the strikingness of every term for each class per time bucket and compiles a total from this (where every χ^2 measure is based only on the data per time bucket).

4. EXPERIMENTS

By integrating a time-dependent function in the χ^2 equation, we can easily compare different adaptive settings. The decay functions (Section 3.3) allow for weighting of past material and defining a sliding window at the same time. In what follows we discuss the experimental setup used for assessing our approaches, as well as the outcomes.

4.1 Experimental setup and data

Data. The NIST TREC 2012 conference hosted a Knowledge Base Acceleration (KBA) task whose aim it was to filter a stream for documents relevant to a set of topics. We use the TREC 2012 KBA data as described in [2]. There are 29 topics of interest and the documents are annotated for different levels of relevance to these topics. Only documents annotated as being 'central' are taken into account in our experiments. After de-deduplication we are left with 8,208 documents in total, of which 3,140 are training documents (i.e., they were published before the temporal cutoff) and 5,068 are test documents (after the temporal cutoff). After lowercasing and stemming (with the Porter stemmer), this gives us 139,284 unique features.

Number of training examples. In order to mimic a real world scenario where a user is interested in a particular topic but does not have tens or hundreds of example documents at hand we use very little training examples in our experiments; we only select either the last 1 or the last 2 training documents per class. This emulates as best as possible an environment in which adaptation is needed at runtime.

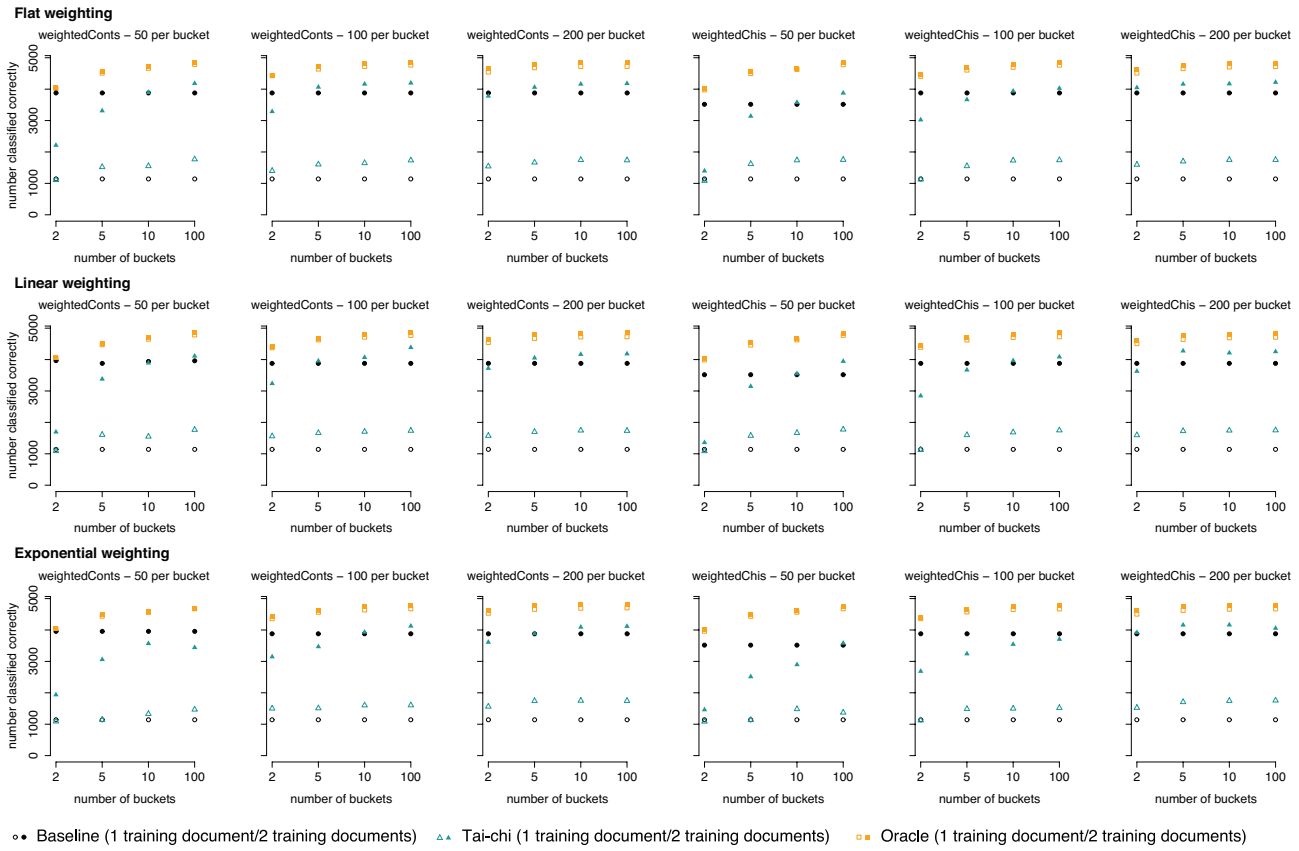

Figure 1: Test results. The rows from top to bottom show graphs for flat, linear and exponential weighting respectively. The three leftmost columns show graphs for feature selection based on weighted contingencies. The three columns at the right side show result graphs for feature selection based on weighted χ^2's. The first and fourth column have 50, second and fifth 100 and third an sixth column 200 examples per time bucket. The x-axis displays the number of buckets taken into account. The y-axis lists the number of examples classified correctly. Results are by displayed the amount of documents per bucket (2, 5, 10, 100). Cf. Section 4.1.

Time buckets. Time buckets are filled by collecting examples as they come in over time and making a new bucket when a certain limit is reached. An advantage of this approach is that it automatically allows for fast adaptation when a lot of documents appear in a short time span, while a more gradual evolution takes place when fewer documents are issued.

An alternative way would be to collect all documents published in a certain time period (measured in, e.g., seconds or days). This poses problems when using the χ^2 metric as we are, especially in Equation 3, as χ^2 is undefined when, e.g., only one document occurs in a bucket. Also, it is not immediately evident what should happen when no documents occur in one or more buckets.

In our experiments we vary the number of examples per bucket between 50, 100 and 200. The number of time buckets (i.e., the window size in terms of time buckets) is either 2, 5, 10 or 100.

Feature selection. Based either on Equation 2 or 3 we take the top n features per class ('weightedConts' and 'weightedChis,' respectively in the graphs above). These features together comprise the vocabulary used for the multinomial Naive Bayes classifier. E.g., if we select a maximum of 5 features per class, we end up with $5 \cdot 29 = 145$ features at most (features might overlap).

We empirically determined the optimal number of features per class to be 7 or 8, yielding up to 203 and 232 features, respectively. In the experiments presented below we used a setting with maximally 8 features per class. This number deviates from what is reported in, e.g., [3] who use 500 features in total, though they actually note that "past results have shown that a few hundreds of words are an appropriate size of features."

Adaptation strategies. In our experiments, three adaptation strategies are employed. The baseline approach is to adapt up until the last training example and remain static from then on. As noted above, a common approach in research is to allow for adaptation based on gold standard labels. We refer to this as the 'oracle' setting. Arguably the most interesting setting is one in which the classifier tries to learn from its own predicted labels. This resembles a real world setting where no explicit feedback is available. As this setting most properly reflects our Time aware implementation of chi-squared we refer to it as Tai-chi.

Do note that in this adaptive setting updates to the probabilities and the selection of the features are based on all

predicted labels; the ones predicted correctly as well as the wrong ones. While this poses a threat of drifting in the wrong direction, the results show that adapting in this manner can lead to improvement over the non-adaptive baseline.

4.2 Results

Baselines. As we can see from the graphs in Figure 1, training on just one example gives a low baseline (1114 classified correctly, 21.98%). The adaptive Tai-chi approach (green transparent triangles) is able to improve over this in almost all cases. So even though very few of the examples are classified correctly, the algorithm is able to learn. The baseline consistently scores higher in the setting where the classifier learns from two examples per class (the opaque data points), with between 3517 (69.40%) and 3969 (78.31%) instances classified correctly. It is evident from the graphs that the adaptive runs perform less well when limited history is taken into account (a small amount of examples per time buckets and few time buckets). The weighted contingencies algorithm appears to be most robust to this. The 5 buckets setting already shows improvements over the baseline with 100 examples per bucket (i.e., a window size of 500 examples in total) both with flat weighting and with linear weighting.

Our approach. The weighted χ^2's approaches (rightmost three columns of the graphs in Figure 1) prove to be vulnerable in settings with limited amounts of buckets and small bucket size, especially in the adaptive setting with two training examples (the green opaque triangles). This is because the χ^2 measure is calculated, for every word and class, for every bucket. If little data is available in a bucket, less reliable χ^2 scores will be obtained and any choice of features based on these will suffer from this. This will cause the classifier to make even more errors, after which the process repeats.

In the training from two examples setting, the exponential weighting for the Tai-chi runs leads to improvements over the baseline in fewer cases than when the other weighting functions are applied. This is in line with the observation in [5] that selecting batches works better than weighting them, even though applying linear weights leads to scores comparable to the ones with flat weights being applied, in our tests.

The reason the different decay functions do not appear to have a large effect might lie in the fact that they are only being applied for selecting features (rather than weighting the features themselves). Relative to the total number of possible features (139,284) we select very few (at most 232 as noted above). Hence it might be that regardless of the weighting being applied, the same features will be chosen in all settings.

Another reason, which would also explain the relatively high scores of the baselines (close to 90% in most settings where two training examples are used) and their consistency across different numbers of time buckets and their sizes, might be that there is in fact no immediate need to adapt in the material at hand. If concepts do not evolve a lot over time this would also contribute to the lack of discriminative power the weighting functions demonstrate.

Oracle runs. As noted earlier, a lot of existing research is based on classifiers learning from the correct labels. We followed this strategy as well (referred to as the 'oracle' runs, the orange squares in Figure 1). These runs always improve over the baseline and can reach high scores (up to 4863 (95.96%) for weighted contingencies, flat weighting, 100 buckets, 50 examples per bucket, and 4862 (95.94%) for the same settings with linear weighting). The only difference between the two oracle runs is one example per class in their initial training material. This explains the high similarity between their results.

It is interesting to note that in these top scoring cases only part of the material seen so far is taken into account (not all as, e.g., in the 100 buckets, 100 examples per bucket case). This shows that being adaptive over time is beneficial.

5. CONCLUSION AND FUTURE WORK

We have presented two time-aware versions of the χ^2 measure for selecting features for a multinomial Naive Bayes classifier. Results show that in a setting where little training material is available (only one or two training examples per class) the algorithms presented can improve over a non-adaptive baseline.

Further research will focus on example selection. The gap between the regular adaptive and oracle runs might be reduced by a more considerate adaptation approach. In the experiments reported here, the non-oracle adaptive runs learn from all examples, whether predicted correctly or erroneously. This may be improved upon by selecting the most promising examples or the ones we are most confident about.

Acknowledgements. This research was partially supported by the European Community's Seventh Framework Program (FP7/2007-2013) under grant agreements nr 258191 (PROMISE Network of Excellence) and 288024 (LiMoSINe project), the Netherlands Organisation for Scientific Research (NWO) under project nrs 640.004.802, 727.011.005, 612.001.116, HOR-11-10, the Center for Creation, Content and Technology (CCCT), the QuaMerdes project funded by the CLARIN-nl program, the TROVe project funded by the CLARIAH program, the Dutch national program COMMIT, the ESF Research Network Program ELIAS, the Elite Network Shifts project funded by the Royal Dutch Academy of Sciences (KNAW), the Netherlands eScience Center under project number 027.012.105 and the Yahoo! Faculty Research and Engagement Program.

6. REFERENCES

[1] J. Allan. Introduction to topic detection and tracking. In *Topic detection and tracking*, pages 1–16. Springer, 2002.
[2] J. Frank, M. Kleiman-Weiner, D. Roberts, F. Niu, C. Zhang, C. Ré, and I. Soboroff. Building an entity-centric stream filtering test collection for TREC 2012. In *Proceedings of the 21st TREC*, 2012.
[3] I. Katakis, G. Tsoumakas, and I. Vlahavas. Dynamic feature space and incremental feature selection for the classification of textual data streams. In *PKDD*, pages 102–116, 2006.
[4] H. J. Kim and J. Chang. Integrating incremental feature weighting into naive bayes text classifier. In *Machine Learning and Cybernetics, 2007 International Conference on*, volume 2, pages 1137–1143, 2007.
[5] R. Klinkenberg. Learning drifting concepts: Example selection vs. example weighting. *Intelligent Data Analysis*, 8 (3):281–300, 2004.
[6] Y. Yiming and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML '97*, pages 412–420, 1997.