Learning Threshold Parameters for Event Classification in Broadcast News

Ron Papka
Center for Intelligent Information Retrieval
Computer Science Department
University of Massachusetts
Amherst, MA 01003
papka@cs.umass.edu

Abstract

In this paper we present two methods for automatic threshold parameter estimation for an event tracking algorithm. We view the threshold as a statistic of the incoming data stream, which is assumed to contain broadcast news stories from radio, television, and newswire sources. Query bias defined in terms of threshold estimators can be identified when a word co-occurrence representation for text is used. Our results suggest that both approaches learn bias from training corpora, leading to improved classification accuracy for event tracking applications.

1 Introduction

The following work describes two automatic threshold selection algorithms for the event tracking problem. This problem was defined by the Topic Detection and Tracking (TDT) research initiative, a DARPA-sponsored effort comprising research groups from several commercial and academic sites. Event tracking is a form of supervised learning in which a system formulates a classifier for broadcast news using a few relevant stories discussing an event.

The tracking task is similar in nature to the filtering problem of TREC [3], but there are four fundamental differences: 1) the information domain is a temporally ordered stream of broadcast news; 2) information requests specifically concern *events* and not *topics*; 3) classification decisions must be made on-line with no knowledge of future news; and 4) classifiers are trained using only 1, 2, 4, 8, or 16 relevant training documents for each event.

In TDT, an *event* refers to some unique thing that happens at some point in time. The property of *time* is the salient feature that distinguishes an event from the more general *topic*. For example, "the computer virus detected at British Telecom, March 3, 1993," is an occurrence of an event, whereas "computer virus outbreaks" would be considered a more general *topic* using the TREC convention. This definition can

be extended to include the spatial component of an event, namely location. For example, "the 1995 earthquake in Kobe, Japan" is a description of an event that uses this property.

What has emerged as the text classifier of choice for TDT and TREC research is a query represented as a vector of features comprising stemmed words and associated weights. Query words and weights are determined through statistical and learning techniques using inter- and intra-document word occurrences in the training data. The most common document representation is a vector of *tf·idf* weights co-occurring with the words in the query. Queries are compared to documents using a similarity function, which in many instances is an extension of the inner-product of two vectors. This representation has the nice property of being extensible to linear classification using both supervised and unsupervised learning approaches. In what follows, we define and illustrate threshold estimator bias for document classification based on this representation. We discuss two methods that produce bias-reduced estimators resulting in improved classification accuracy.

2 Evaluation Methodology for Tracking

The tracking methodology used for TDT is based on a temporally ordered view of the data, in which the same corpus is used for both training and testing. If a system is evaluated using four relevant training documents, that is, $N_t = 4$, then all documents in the stream up to and including the fourth training document are considered the training corpus, and the testing corpus comprises the documents in the remainder of the stream. This methodology implies that different events effectively have different training and test corpora. In contrast, the TREC methodology uses a holdout method with independent training and testing corpora.

Classification is evaluated using a cost function which is a linear combination of system miss and false alarm rates. In TDT2 and in the experiments that follow, we used $Cost = cost_{fa} * P(fa) + cost_m * P(m)$, where $P(fa)$ is the probability that a system produces a false alarm, i.e., its false alarm rate, and $P(m)$ is the probability that a system produces a miss, i.e., its miss rate. In TDT2, cost was defined with $cost_{fa} = 0.98$ and $cost_m = 0.02$, and was based on the prior probability of a document being relevant to an event as being 0.02.

3 Implementation

Our tracking system uses the Inquery retrieval engine [2], and we have extended the engine's document ranking functionality with the ability to make hard classification decisions. We use a query formulation process that works on-line to create an event classifier from single or multiple documents using a word co-occurrence model and a *tf·idf* representation for text. Query formulation involved a three-step process of term selection, weight assignment, and threshold estimation, the details of which are described in [7, 8]. The query and thresholds form a document classifier, which is applied to the testing portion of the stream. Queries are formulated for each event using the n most frequent nonstopwords from the relevant documents in the training data. The words are given weights using an assignment based on the number of times the word appears in the relevant and non-relevant training documents.

3.1 TDT Corpora

The Linguistic Data Consortium [1] (LDC), a participant in the TDT project, has maintained the four sets of news used in this work. The TDT1 corpus includes 15862 documents from CNN broadcast news and Reuters newswire from 07/01/1994 through 06/30/1995. The sources from the TDT2 corpora are divided into three sets which are referred to below as TDT2-Train, TDT2-Dev, and TDT2-Eval. The TDT2 sets each contain two months worth of news comprising 63,309 documents from the New York Times News Service, Associated Press Worldstream News Service, CNN Headline News, ABC World News Tonight, PRI The World, and VOA English News Programs from 01/04/1998 through 06/30/1998. The audio sources for TDT1 were converted to text using a manual transcriptions of the program. The audio signals from the TDT2 corpora were converted to text using Dragon Systems' automatic speech recognition (ASR) system [9]. The document boundaries for the audio sources were determined as part of the manual annotation effort for the corpora.

Each corpus contains relevance judgments for between 25 and 35 specific events for a total of 119 events. In the experiments that follow, we used the 85 events and text available from the TDT1, TDT2-Train, TDT2-Dev corpora for training, and the 34 events from the TDT2-Eval corpus to test classification effectiveness. Documents were judged on a ternary scale to be non-relevant, to have content relevant to the event, or to contain only a brief mention of the event in a generally non-relevant document. The corpora were exhaustively and independently judged for each event; however, all the events in the data were not identified. There were totals of 7782 relevant documents and 1903 brief mentions available for the TDT corpora. Brief mentions were excluded from processing.

3.2 TDT2 Event Tracking Experiments

The second phase of TDT (TDT2) ended in March 1999 with an evaluation presented at the DARPA Broadcast News Workshop. Participants had access to training corpora (TDT1, TDT2-Train, and TDT2-Dev) in addition to the manual annotations and judgements for each event. These data were used to train and develop the systems, and the TDT2-Eval corpus was used in a blind evaluation, in which participants did not have access to the judgements prior to submitting their results. A comparison of tracking results from research groups at the University of Pennsylvania, University of Massachusetts, BBN, Dragon Systems, Carnegie Mellon, GE, University of Maryland, and University of IOWA are listed in Table 1. In the table, lower TDT2 cost implies higher classification accuracy. Due to space limitations we can not describe these systems here, and we refer the reader to the site writeups in the *Proceedings of the DARPA Broadcast News Workshop (1999).*

We tested static approaches where query terms and thresholds are held constant over time, and adaptive approaches where unlabeled documents assumed to be relevant to an event are used to reformulate the query and threshold over time. We also tested several weight-learning approaches as extensions to the static query formulation process. Our results suggested that weight-learning approaches such as Widrow-Hoff [10], Exponentiated Gradient Descent [5], and Dynamic Feedback Optimization [1] are of limited use for event tracking. It was observed that the

[1] http://www.ldc.upenn.edu

Table 1: NIST evaluation of TDT2 Tracking Systems (Story-Weighted Cost, Nt=4).

	TDT2 Cost	$P(m)$	$P(fa)$
UPENN	0.0058	0.0934	0.0040
UMASS-A	0.0059	0.0855	0.0043
BBN	0.0063	0.1415	0.0035
UMASS-S	0.0070	0.0747	0.0056
DRAGON	0.0070	0.1408	0.0043
CMU	0.0077	0.2105	0.0035
GE	0.0216	0.1451	0.0191
UMD	0.0225	0.8197	0.0062
UIOWA	0.0499	0.0819	0.0492

classifiers and thresholds that were formulated already separated the training data when few relevant training instances were used, so not much improvement resulted from weight-learning. We also found that the additional threshold parameter in our adaptive tracking approach (UMASS-A, in Table 1) made it more difficult to determine good threshold parameters empirically than did estimating the single parameter needed for our static tracking approach (UMASS-S), which is described in this work. Since it appeared that our representation for text was comparable to those of other systems, we determined to explore the issues related to threshold parameter estimation, which is the algorithm that makes the hard classification decision.

4 Threshold Estimation Experiments

The threshold methodology that we used involved an optimization process combined with a ranked-retrieval process. The optimization step is to find the query/document similarity score s, that when used as a threshold u, optimizes average user utility over the labeled documents in the sorted list of training instances returned by the ranked-retrieval engine. In the experiments below, we estimate a threshold u for each query with estimator $\hat{u} = 0.4 + \theta * (s_{optimized} - 0.4)$, where 0.4 is an Inquery constant, θ is a global system parameter, and $s_{optimized}$ is the similarity value resulting from the query that, when applied to the event's labeled training documents, optimizes the target TDT2 cost function defined in Section 2.

From preliminary experiments using the 85 events that were available from the TDT1, TDT2-Train and TDT2-Dev corpora, it was determined that when fewer relevant training documents were used, $s_{optimized}$ (our estimator $\hat{u}$ when $\theta = 1.0$) was consistently above the parameter u it was trying to estimate. In other words, the optimal threshold for the unprocessed stream of data for a particular event, or simply $s_{optimal}$, was overestimated using $s_{optimized}$. In what follows, we assume that the quantity of estimator bias $b(\hat{u}) = E[s_{optimal}] - s_{optimized}$, which is similar to the definition of bias used to analyze classification effectiveness [6]. We attempt to learn threshold estimator bias over varying numbers of relevant training documents (Nt) and query/document features.

4.1 The Histogram Method

The histogram method, which utilizes threshold estimator $\hat{u}$, is illustrated with 50-feature queries in Figure 1. Using the TDT1, TDT2-Train, and TDT2-Dev corpora training data, histograms of optimal values for θ were collected for each value of Nt

and for queries formulated with 10, 20, 50, 100, 200, 600, and 10000 features. Using this method on the training data, for example, determined that for one relevant document ($Nt = 1$) and 50-feature queries, 48 out of 85 queries had optimal cost when $\theta = 0.2$. For each pair of Nt and number of features, we calculate $E[\theta]$ from the corresponding histogram, and use the resulting value for θ when estimating thresholds on the evaluation corpus (TDT2-Eval).

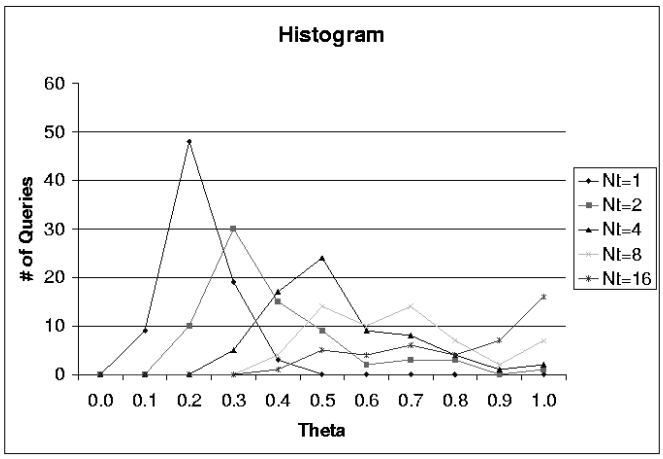

Figure 1: Histograms of optimal threshold parameter θ for varying Nt (50 features).

The data in Figure 1 illustrate estimator bias when $N_t \leq 16$ relevant training documents are used. The data suggest that as Nt increases $E[\theta]$ increases. Also, as $E[\theta]$ increases, on average, $s_{optimized}$ is closer to $s_{optimal}$; thus less total estimator bias results when more relevant training examples are used. We also observed similar but less significant increases in bias when more features were used.

4.2 Linear Regression Method

The observation that increasing training instances reduces the bias of an estimator, in general, is not surprising. James, for example, shows that estimates move toward the true population values when training instances are increased for data assumed to have multivariate-normal distributions [4]. An explanation of the phenomenon follows from the law of large numbers. However, James also proposes that once it is found, it may be possible to reduce the bias using a linear transformation.

We test James's theory in the following experiments. We define a threshold estimator $\hat{o}$, such that $\hat{o} = s_{optimized}$. We then define a new bias-reduced threshold estimator $\hat{e}$, such that $\hat{e} = m\hat{o} + b$. We then compare the effectiveness of the query thresholds produced by our original estimator $\hat{u}$, which is bias-reduced using near optimal values for θ, to those produced by $\hat{e}$, with parameters m and b learned through linear regression. In what follows, we use linear regression to learn query bias for estimator $\hat{o}$.

The linear regression method for estimator $\hat{e}$ is illustrated in Figure 2. Instead of collecting histograms, points represented by $s_{optimized}$ and $s_{optimal}$ are fitted using a line for each value of Nt and number of features. The slopes and intercepts of

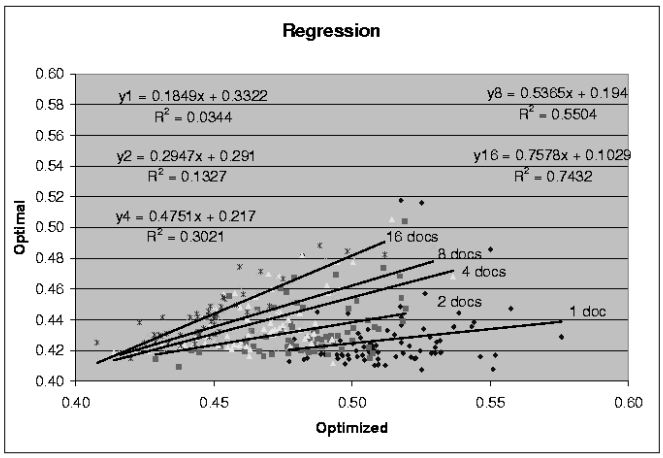

Figure 2: Regression of optimal threshold parameter θ for varying Nt (50 features).

lines produced by the various regressions are subsequently used as parameters m and b for estimator $\hat{e}$ during evaluation. This method learns the same tendencies in threshold estimator bias as the histogram approach for estimator $\hat{u}$. As the number of relevant training documents increases, the slope of the resulting regression line approaches 1.0. As the slope approaches 1.0, $s_{optimized}$ approaches $s_{optimal}$, which implies that higher values of Nt give rise to less estimator bias in the training data.

4.3 Comparison of Methods

From the TDT2 tracking experiments discussed in Section 3.2, we knew the histogram approach for estimator $\hat{u}$ generalized and provided relatively low cost on the TDT2-Eval corpus. In the following experiment, we compare the effectiveness of threshold estimator $\hat{e}$ to that of $\hat{u}$ using the same static query formulation and tracking processes on the evaluation corpus. The results, which are listed in Table 2, illustrate the percent increase in cost resulting from tracking with estimator $\hat{e}$ instead of estimator $\hat{u}$.

Table 2: Percent of cost increase from replacing estimator $\hat{u}$ with estimator $\hat{e}$.

# of Features	Number of Relevant Training Documents (Nt) 1	2	4	8	16
10	8.2%	-20.4%	-13.2%	-21.9%	-2.9%
20	27.5%	-17.0%	-32.1%	-23.4%	-1.6%
50	5.0%	53.2%	0.0%	-4.0%	-8.6%
100	1.1%	12.7%	4.5%	-6.0%	-20.3%
200	3.3%	-7.5%	-8.2%	2.0%	-16.4%
600	27.8%	-6.0%	-7.3%	-8.0%	-1.8%
10000	27.8%	-6.0%	-7.3%	-8.0%	0.0%
Average	14.4%	1.3%	-9.1%	-9.9%	-7.4%

In Table 2, an increase in cost implies a decrease in classification effectiveness, hence the data suggest that, on average, the histogram method works better for $Nt = 1$ and marginally so for $Nt = 2$. However, for $Nt > 2$ the regression method appears

to reduce cost consistently for most of the query sizes tested. The correlation coefficients (R^2) that are calculated for the regressions depicted in Figure 2 indicate that using fewer documents led to lower correlation than using more documents. This trend was evident across the varying number of query sizes that were tested. This suggests that the resulting regression was not a good fit for $Nt \leq 2$. However, the improvements realized by the regression method suggest a mixture of approaches where estimator $\hat{u}$ is used for $Nt \leq 2$ and estimator $\hat{e}$ otherwise.

5 Conclusion

We described two approaches for automatic threshold parameter estimation for a static query formulation process applied to the problem of event tracking. We view the threshold as a statistic of the incoming news stream that is estimated using an optimization process for a pre-specified utility measure applied to the training data. We defined the notion of query bias in terms of threshold estimators, and illustrated that the amount of bias increases when fewer relevant training samples are used on the TDT data. Our results suggest that our automatic thresholding approaches can learn query bias and result in effective estimates of threshold parameters. We suggest using the histogram approach when estimating thresholds for $Nt \leq 2$ relevant instances, and the linear regression approach otherwise.

Acknowledgments

This material is based on work supported in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623 and the Air Force Office of Scientific Research under grant number F49620-99-1-0138. Any opinions, findings and conclusions or recommendations expressed in this material are the author's and do not necessarily reflect those of the sponsor.

References

[1] C. Buckley and G. Salton, "Optimization of Relevance Feedback Weights," *Proceedings of ACM SIGIR*, 351-357, 1995.

[2] J. Callan, B. Croft, and J. Broglio, "TREC and TIPSTER Experiments with INQUERY," *Information Processing & Management*, 31(3):327-343, 1994.

[3] D.K. Harman, *Proceedings of Text REtrieval Conferences (TREC)* 1993-9.

[4] M. James, *Classification Algorithms*, John Wiley & Sons, New York, 1985.

[5] J. Kivinen and M. Wartmuth, "Exponentiated Gradient Versus Gradient Descent for Linear Predictors," UCSC Technical report: UCSC-CRL-94-16, 1994.

[6] T. M. Mitchell, *Machine Learning*, McGraw-Hill, Boston, MA, 1997.

[7] R. Papka, J.P. Callan, and A.G. Barto, "Text-Based Information Retrieval Using Exponentiated Gradient Descent," *Proceedings of NIPS*9*, 3-9, 1996.

[8] R. Papka, J. Allan, and V. Lavrenko, "UMASS Approaches to Detection and Tracking at TDT2," *Proceedings of the DARPA Broadcast News Workshop*, 1999.

[9] S. Wegmann, P. Zhan, I. Carp, M. Newman, J. Yamron, and L. Gillick, "Dragon Systems 1998 Broadcast News Transcription System," *Proceedings of the DARPA Broadcast News Workshop*, 1999.

[10] B. Widrow and M. Hoff, "Adaptive Switching Circuits," *IRE WESCON Convention Record*, 96-104, 1960.