Optimizing Term Vectors for Efficient and Robust Filtering

David A. Evans
Clairvoyance Corporation
5001 Baum Blvd., Suite 700
Pittsburgh, PA 15213-1854, USA
+1-412-621-0570
dae@clairvoyancecorp.com

Jeffrey Bennett
Clairvoyance Corporation
5001 Baum Blvd., Suite 700
Pittsburgh, PA 15213-1854, USA
+1-412-621-0570
bennett@clairvoyancecorp.com

David A. Hull
Clairvoyance Corporation
5001 Baum Blvd., Suite 700
Pittsburgh, PA 15213-1854, USA
+1-412-621-0570
d.hull@clairvoyancecorp.com

ABSTRACT

We describe an efficient, robust method for selecting and optimizing terms for a classification or filtering task. Terms are extracted from positive examples in training data based on several alternative term-selection algorithms, then combined additively after a simple term-score normalization step to produce a merged and ranked master term vector. The score threshold for the master vector is set via *beta-gamma* regulation over all the available training data. The process avoids parameter calibrations and protracted training. It also results in compact profiles for run-time evaluation of test (new) documents. Results on TREC-2002 filtering-task datasets demonstrate substantial improvements over TREC-median results and rival both idealized IR-based results and optimized (and expensive) SVM-based classifiers in general effectiveness.

Categories and Subject Descriptors

I.2.7 [**Artificial Intelligence**]: Natural Language Processing -- *Text analysis*. I.2.6 [**Artificial Intelligence**]: Learning -- *Parameter learning.* H.3.3 [**Information Systems]:** Information Search and Retrieval -- *Information filtering*.

General Terms

Algorithms, Measurement, Experimentation, Theory.

Keywords

Term Selection, Vector Optimization, Text Filtering

1. INTRODUCTION

Filtering and classification tasks are very challenging, in part, because it is difficult to predict, for any particular topic exemplified by a small number of examples, which of many possible filter-creation techniques will give the best results. The factors that contribute to variability in performance include selecting (and weighting) appropriate features to represent the topic space, establishing an evaluation metric (e.g., scoring with thresholds) to be used on target documents, and modeling the user's interests (sometimes approximated by a utility function). Complicating this picture further is the need in practical applications for methods that are computationally efficient and robust. Satisfying these requirements remains an elusive goal.

Some of the best recent efforts on TREC-2002 filtering tasks illustrate the problem. The IRIT Group achieved one of the highest scoring batch-filtering runs by using a term-calibration procedure based on back propagation of relevance judgments for filter-profile creation [1]. The training phase of this method is likely efficient, but the resulting filter may be extremely large—reflecting the total term space of the training set [2]. This implies a higher cost in actual application of the filter. In contrast, the North Texas University Group achieved high performance with relatively small classifiers that are arguably very efficient for testing [3]. However, establishing the classifiers requires an elaborate training procedure, involving selecting from among thirteen alternative models for each topic. Other groups have remarked on and explored the problem of topic-specific term-profile optimizations [4]. It is clearly desirable to use varieties of techniques or models for each individual topic and select the one that best reflects that topic's special features. The most important and successful alternative approaches to filter/classifier creation involve SVM learning algorithms. These have the advantage of being completely general, while still giving often excellent results. However, they do not work consistently well when there are small numbers of training data and they may require elaborate threshold calibration to achieve the best results on specific datasets.

2. APPROACH

We have developed an approach to filter creation that minimizes training, results in efficient profiles for application, and gives consistently strong results. The approach has several steps:

1. Extract terms from all positive examples using each of one or more alternative methods (cf. Figure 1 for details of the extraction techniques). This results in a weight-ranked list of terms for each method. Normalize the weights in each list by dividing the weight for each term by the weight of the first ranked term.
2. Truncate each list by discarding terms below the rank where the term weight profile begins to "level off"—determined as the point where the second-derivative (w'') satisfies the condition $0 > w'' > \varepsilon$ ($\varepsilon = 0.0001$).
3. Merge the surviving terms from each list additively.
4. Set the scoring threshold for the merged vector using *beta-gamma* regulation [5] on the full training set.

3. EXPERIMENTS

We conducted batch filtering experiments on the first fifty (so-called "Assessor") topics of the TREC-2002 Reuters96 collection. Only three topics in this dataset have more than twenty positive training examples; 33 have ten or fewer.

Using the procedure of Section 2, we prepared filters using each of our four extraction methods individually and using merged term sets, one for two extractors (Prob2 and Rocchio) and one for all four. We also prepared filters based on predicting which method would work best for each topic. In this case, we split training data into two equal sets and used all positive examples in the first set for candidate filter creation and threshold calibration. We tested the candidate filter on the second set of data and chose as the predicted-best method the one that gave the best results on this test. We then used the procedure of Section 2 with this chosen method. In one case, we made predictions based on Prob2 and Rocchio; in another, based on all four methods. (For topics with ten or fewer examples, we used all examples, and set thresholds and tested over all training data.) Thresholds were optimized for T11F; β=0.1; γ=0.4.

4. RESULTS AND DISCUSSION

Table 1 gives results and comparative performance, including TREC-2002 medians and two reported "good" runs. We also show representative SVM results (from our group), one with thresholding (requiring calibration effort) and one without. The four runs using a single extraction method each give reasonable results. The "Pseudo" runs show an idealized best performance (if we could predict and use the actually best extraction method for each topic); this sets a high ceiling on the expected results. The "AutoPick" runs show the results of our attempts to predict a best method based on tests on the training data. It is clear that the best method varies by topic and that it is difficult to predict which will work best. The merged runs give consistently solid performance, better than predicting and generally better than any individual method. Merged runs are dramatically better than TREC median and expected non-thresholded SVM performance on T11F (F-Beta) measures. They are comparable to thresholded SVMs, without requiring calibration. Filters in Merge-2 had an average of 56 terms and in Merge-4 201 terms. The merge approach is extremely efficient in both training and processing and can be used in a wide variety of applications.

5. REFERENCES

[1] Boughanem, M., Tebri, H., Tmar, M. IRIT at TREC '2002: filtering track. *Text Retrieval Conference (TREC-2002) Conference Notebook*, 2002, 413–422.

[2] Boughanem, M., Chrisment, C., Tmar, M. Mercure and MercureFiltre applied for web and filtering tasks at TREC-10. *The Tenth Text Retrieval Conference (TREC-2001)*, 2002, 303–312.

[3] Mihalcea, R. Classifiers stacking and voting for text filtering. *Text Retrieval Conference (TREC-2002) Conference Notebook*, 2002, 696–701.

[4] Evans, D.A., Shanahan, J., Tong, X., Roma, N., Stoica, E., Sheftel, V., Montgomery, J., Bennett, J., Fujita, S., Grefenstette, G. Topic specific optimization and structuring. *Tenth Text Retrieval Conference (TREC-2001)*, 2002, 132–141.

[5] Zhai, C., Jansen, P., Stoica, E., Grot, N., Evans, D.A. Threshold calibration in CLARIT adaptive filtering. *Seventh Text Retrieval Conference (TREC-7)*, 1999, 149–156.

$$Prob2(t) = log(R_t + 1)\ x \left(log(\frac{N - R + 2}{N_t - R_t + 1} - 1) - log(\frac{R + 1}{R_t} - 1) \right)$$

$$Rocchio(t) = IDF(t)\ x\ \frac{\sum_{D \in DocSet} NTF_D(t)}{R}$$

$$RocchioFQ(t) = IDF(t)\ x\ \frac{\sum_{D \in DocSet} TF_D(t)}{R}$$

$$GL2(t) = \frac{4xR_t xN_t}{(R + N_t)^2}$$

N is the number of documents in the (reference) corpus; N_t is the number of documents in the (reference) corpus that contain term t; R is the number of documents (for training or feedback) relevant to the topic; R_t is the number of documents (for training or feedback) that are relevant to the topic and contain term t; TF is the (raw) frequency of term t in a document; and NTF is the normalized frequency of term t in a document.

Figure 1. Term-extraction formulae

Table 1. Results comparisons: TREC-2002 "Assessor" topics

Run	Set Prec.	T11U	T11F	Recall
REC-2002 Median	*0.340*	*0.377*	***0.234***	*0.235*
TREC-2002 N.Texas	0.682	0.446	**0.444**	0.237
TREC-2002 IRIT	0.661	0.485	**0.455**	0.321
SVM Thresholded	0.549	0.426	**0.403**	0.303
SVM No-Threshold	0.299	0.368	**0.116**	0.057
IR-2D Prob2	0.487	0.333	**0.317**	0.320
IR-2D Rocchio	0.446	0.333	**0.342**	0.347
IR-2D RocchioFQ	0.479	0.402	**0.367**	0.323
IR-2D GL2	0.427	0.372	**0.313**	0.256
Average of 4Methods	*0.460*	*0.360*	***0.335***	*0.312*
IR-2D Pseudo Best2	*0.567*	*0.397*	***0.397***	*0.424*
IR-2D Pseudo Best4	*0.640*	*0.467*	***0.456***	*0.473*
IR-2D AutoPick2	0.412	0.321	**0.346**	0.414
IR-2D AutoPick4	0.458	0.334	**0.341**	0.340
IR-2D Merge 2	0.452	0.373	**0.378**	0.408
IR-2D Merge 4	0.515	0.386	**0.382**	0.336
Merge-4:TREC-Med	*51.5%*	*2.4%*	***63.2%***	*43.0%*
Merge-4:Average 2D	*12.0%*	*7.2%*	***14.0%***	*7.7%*
Merge-4:SVM-Thres	*-6.19%*	*-9.39%*	***-5.21%***	*10.89%*
Merge-4:SVM-NonT	*72.2%*	*4.9%*	***229.3%***	*489.5%*