Small Statistical Models by Random Feature Mixing

Kuzman Ganchev and **Mark Dredze**
Department of Computer and Information Science
University of Pennsylvania, Philadelphia, PA
{kuzman,mdredze}@cis.upenn.edu

Abstract

The application of statistical NLP systems to resource constrained devices is limited by the need to maintain parameters for a large number of features and an alphabet mapping features to parameters. We introduce random feature mixing to eliminate alphabet storage and reduce the number of parameters without severely impacting model performance.

1 Introduction

Statistical NLP learning systems are used for many applications but have large memory requirements, a serious problem for mobile platforms. Since NLP applications use high dimensional models, a large alphabet is required to map between features and model parameters. Practically, this means storing every observed feature string in memory, a prohibitive cost for systems with constrained resources. Offline feature selection is a possible solution, but still requires an alphabet and eliminates the potential for learning new features after deployment, an important property for adaptive e-mail or SMS prediction and personalization tasks.

We propose a simple and effective approach to *eliminate* the alphabet and reduce the problem of dimensionality through random feature mixing. We explore this method on a variety of popular datasets and classification algorithms. In addition to alphabet elimination, this reduces model size by a factor of 5–10 without a significant loss in performance.

2 Method

Linear models learn a weight vector over features constructed from the input. Features are constructed as strings (e.g. "`w=apple`" interpreted as "contains the word apple") and converted to feature indices maintained by an alphabet, a map from strings to integers. Instances are efficiently represented as a sparse vector and the model as a dense weight vector. Since the alphabet stores a string for each feature, potentially each unigram or bigram it encounters, it is much larger than the weight vector.

Our idea is to replace the alphabet with a random function from strings to integers between 0 and an intended size. This size controls the number of parameters in our model. While features are now easily mapped to model parameters, multiple features can collide and confuse learning. The collision rate is controlled by the intended size. Excessive collisions can make the learning problem more difficult, but we show significant reductions are still possible without harming learning. We emphasize that even when using an extremely large feature space to avoid collisions, alphabet storage is eliminated. For the experiments in this paper we use Java's `hashCode` function modulo the intended size rather than a random function.

3 Experiments

We evaluated the effect of random feature mixing on four popular learning methods: Perceptron, MIRA (Crammer et al., 2006), SVM and Maximum entropy; with 4 NLP datasets: 20 Newsgroups[1], Reuters (Lewis et al., 2004), Sentiment (Blitzer et al., 2007) and Spam (Bickel, 2006). For each dataset we extracted binary unigram features and sentiment was prepared according to Blitzer et al. (2007). From 20 Newsgroups we created 3 binary decision tasks to differentiate between two similar

[1] `http://people.csail.mit.edu/jrennie/20Newsgroups/`

Proceedings of the ACL-08: HLT Workshop on Mobile Language Processing, pages 19–20,
Columbus, Ohio, USA, June 2008.

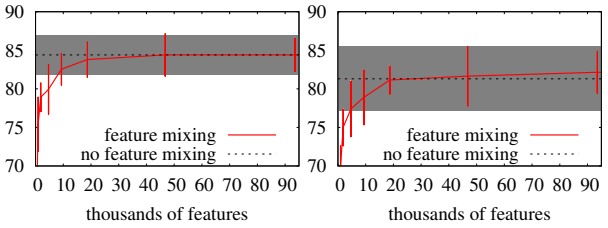

Figure 1: Kitchen appliance reviews. Left: Maximum entropy. Right: Perceptron. Shaded area and vertical lines extend one standard deviation from the mean.

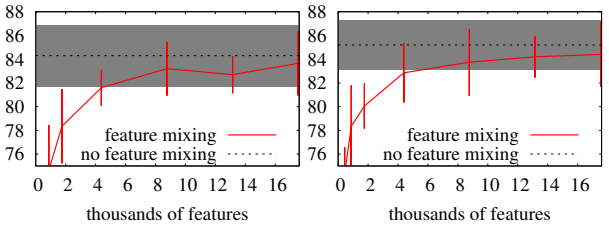

Figure 3: The anomalous Reuters dataset from figure 2 for Perceptron (left) and MIRA (right).

labels from computers, science and talk. We created 3 similar problems from Reuters from insurance, business services and retail distribution. Sentiment used 4 Amazon domains (book, dvd, electronics, kitchen). Spam used the three users from task A data. Each problem had 2000 instances except for 20 Newsgroups, which used between 1850 and 1971 instances. This created 13 binary classification problems across four tasks. Each model was evaluated on all problems using 10-fold cross validation and parameter optimization. Experiments varied model size to observe the effect of feature collisions on performance.

Results for sentiment classification of kitchen appliance reviews (figure 1) are typical. The original model has roughly 93.6k features and its alphabet requires 1.3MB of storage. Assuming 4-byte floating point numbers the weight vector needs under 0.37MB. Consequently our method reduces storage by over 78% when we keep the number of parameters constant. A further reduction by a factor of 2 decreases accuracy by only 2%.

Figure 2 shows the results of all experiments for SVM and MIRA. Each curve shows normalized dataset performance relative to the full model as the percentage of original features decrease. The shaded rectangle extends one standard deviation above and below full model performance. Almost all datasets perform within one standard deviation of the full model when using feature mixing set to the total number of features for the problem, indicating that alphabet elimination is possible without hurting performance. One dataset (Reuters retail distribution) is a notable exception and is illustrated in detail in figure 3. We believe the small total number of features used for this problem is the source of this behavior. On the vast majority of datasets, our method can reduce the size of the weight vector and eliminate the alphabet without any feature selection or changes to the learning algorithm. When reducing weight vector size by a factor of 10, we still obtain between 96.7% and 97.4% of the performance of the original model, depending on the learning algorithm. If we eliminate the alphabet but keep the same size weight vector, model the performance is between 99.3% of the original for MIRA and a slight improvement for Perceptron. The batch learning methods are between those two extremes at 99.4 and 99.5 for maximum entropy and SVM respectively. Feature mixing yields substantial reductions in memory requirements with a minimal performance loss, a promising result for resource constrained devices.

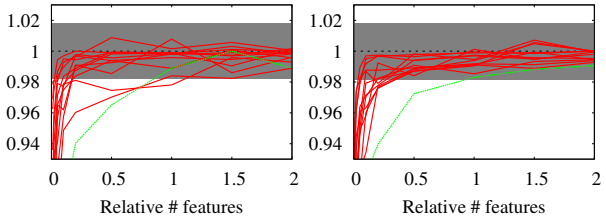

Figure 2: Relative performance on all datasets for SVM (left) and MIRA (right).

References

S. Bickel. 2006. Ecml-pkdd discovery challenge overview. In *The Discovery Challenge Workshop*.

J. Blitzer, M. Dredze, and F. Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*.

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Ressearch*, 7.

D. D. Lewis, Y. Yand, T. Rose, and F. Li. 2004. Rcv1: A new benchmark collection for text categorization research. *JMLR*, 5:361–397.