BOOSTING WORD ERROR RATES

Christos Dimitrakakis and Samy Bengio

IDIAP
CP952
1920 Martigny
Switzerland

ABSTRACT

We apply boosting techniques to the problem of word error rate minimisation in speech recognition. This is achieved through a new definition of sample error for boosting and a training procedure for hidden Markov models. For this purpose we define a sample error for sentence examples related to the word error rate. Furthermore, for each sentence example we define a probability distribution in time that represents our belief that an error has been made at that particular frame. This is used to weigh the frames of each sentence in the boosting framework. We present preliminary results on the well-known Numbers 95 database that indicate the importance of this temporal probability distribution.

1. INTRODUCTION

Boosting and other ensemble learning methods attempt to combine multiple hypotheses from a number of *experts* into a single hypothesis. This is feasible for classification and regression problems, where the hypothesis is a fixed-length vector. Other problems, such as sequence prediction and sequential decision making, can also be cast in the classification and regression framework, thus making the application of ensemble methods to these problems feasible. However, in some cases the hypothesis is a sequence of symbols of unspecified length. One such application is speech recognition, where the hypothesis can be a sequence of words or phonemes.

In a previous paper [1] we have applied boosting to speech recognition at the phoneme level. In that framework, the aim was to reduce the *phoneme classification error* in pre-segmented examples. The resulting boosted phoneme models were combined into a single speech recognition model using *multi-stream* techniques.

Previous approaches for the reduction of word error rate include [2], which employed a "corrective training scheme" and an approach that also used boosting [3]. In the latter, the authors employed a boosting scheme where the sentences with the highest error rate were classified as 'incorrect' and the rest 'correct', irrespective of the absolute word error rate of each sentences. The weights of all frames constituting a sentence were adjusted equally and boosting was applied at the frame level.

In this paper we introduce a new training method, specific to boosting and hidden Markov models (HMMs) in order to reduce the word error rate. We employ a score that is exponentially related to the word error rate of a sentence example. The weights of the frames constituting a sentence are adjusted depending on our expectation of how much they contribute to the error. Finally, boosting is applied at the sentence and frame level simultaneously. This method has arisen from a two-fold consideration: firstly, we need to have an accurate measure of performance, which is the word error rate. Secondly, we need a way to more exactly specify which parts of an example most probably have contributed to errors in the final decision. Using boosting it is possible to focus training on parts of the data which are most likely to give rise to errors, while at the same time doing it in such a manner as to increase an accurate measure of performance. We find that both aspects of training have an important effect.

The paper is organised as follows: the following section gives an introduction to boosting. Section 3 introduces the concept of *expected error*, for the case when no labels are given for the examples. This is important for the task of word error rate minimisation. Section 5 describes word error rate-related measures for boosting. This is followed by a brief section on HMMs and multi-stream decoding, which is used to combine the boosted models. Experimental results are outlined in section 6 and the paper concludes with a discussion in section 7 which discusses the results and possible future research.

2. BOOSTING

Boosting algorithms [4, 5, 6] are a family of ensemble methods for improving the performance of classifiers by training and combining a number of *experts* through an iterative process that focuses the attention of each new expert to the training examples that were hardest to classify by previous ones. The most successful boosting algorithm for classification is AdaBoost [6], where an ensemble of experts is able to decrease the training error exponentially fast as long as each one has a classification error smaller than 50%.

More precisely, an AdaBoost ensemble is composed of a set of n_e experts, $\mathcal{E} = \{e_1, e_2, ..., e_{n_e}\}$. For each input $x \in X$, each expert e_i produces an output $y_i \in Y$. These outputs are combined according to the reliability $\beta_i \in [0, 1]$ of each expert:

$$y = \sum_{i=1}^{n_e} \beta_i y_i.$$

The expert training is an iterative process, which begins with training a single expert and subsequently trains each new expert in turn, until a termination condition is met. The experts are trained on

This work was supported in part by the IST Program of the European Community, under the PASCAL Network of Excellence, IST-2002-506778, funded in part by the Swiss Federal Office for Education and Science (OFES) and the Swiss NSF through the NCCR on IM2.

bootstrap replicates of the training dataset $\mathcal{D} = \{d_i | i \in [1, N]\}$, with $d_i = (x_i, y_i)$. The probability of adding example d_i to the bootstrap replicate $\mathcal{D}_j$ is denoted as $p_j(d_i)$, with $\sum_i p_j(d_i) = 1$. At the end of each boosting iteration j, β_j is calculated according to $\beta_j = \frac{1}{2} \ln \frac{1+\epsilon_j}{1-\epsilon_j}$ where ϵ_j is the average loss of expert e_j, given by $\epsilon_j = \sum_i p_j(d_i) l(d_i)$, where $l(d_i)$ is the *sample loss* of example d_i. If, for any predicate π, we let $[\pi]$ be 1 if π holds and 0 otherwise, it can be defined as: $l(d_i) = [h_i \neq y_i]$. After training in the current iteration is complete, the sampling probabilities are updated so that $p_j(d_i)$ is increased for misclassified examples and decreased for correctly classified examples according to:

$$p_{j+1}(d_i) = \frac{p_j(i) e^{\beta l(d_i)}}{Z_j}, \tag{1}$$

where Z_j is a normalisation factor to make D_{j+1} into a distribution. Thus, incorrectly classified examples are more likely to be included in the next bootstrap data set. Because of this, the expert created at each boosting iteration concentrates on harder parts of the input space.

3. ERROR EXPECTATION FOR BOOSTING

In traditional supervised settings we are provided with a set of examples and labels, which constitute our training set, and thus it is possible to apply margin maximisation algorithms such as Boosting. However this becomes problematic when labels are noisy. Such an example is a typical speech recognition data set. Most of the time such a data set is composed of a set of sentences, with a corresponding set of transcriptions. However, while the transcriptions may be accurate as far as the intention of the speakers or the hearing of the transcriber is concerned, subsequent translation of the transcription into phonetic labels is bound to be error prone, as it is quite possible for either the speaker to mispronounce words, or for the model that performs the automatic segmentation to make mistakes. In such a situation, adapting a model so that it minimises the errors made on the segmented transcriptions might not automatically lead into a model that minimises the word error rate, which is the real goal of a speech recognition system.

For this purpose, we would like to introduce the concept of error expectation in the context of boosting. Thus, rather than declaring with absolute certainty that an example is incorrect or not, we simply define $l(d_i) = P(y_i \neq h_i)$, so that the sample loss is now the probability that a mistake was made on example i and we consider y_i to be a random variable. We consider some cases for the distribution of y in the following section that are relevant to the problem of speech recognition.

3.1. Error Distributions in Sequential Decision Making

In sequential decision making problems the knowledge about the correctness of decisions is delayed. Furthermore, it frequently lacks detailed information concerning the temporal location of errors. A common such case is knowing that we have made one or more errors in the time interval $[1, T]$. This form occurs in individual sentence recognition, episodic reinforcement learning and various other settings. Let us denote the probability of having made an error at time $t \in [1, T]$, given as $P(y_t \neq h_t | y_1^T \neq h_1^T)$. A natural distribution for this case is to assume a flat prior and thus have $P(y_t \neq h_t | y_1^T \neq h_1^T) = 1/T$. Another common case is the exponential prior such that $P(y_t \neq h_t | y_1^T \neq h_1^T) \propto \lambda^{t-T}$, with $\lambda \in [0, 1)$.

In this paper we focus on the application of speech recognition. For the case of labelled sentence examples it is possible to have a procedure that can infer the location of an error in time. This is because correctly recognised words offer an indication of where possible errors lie. Assume some such procedure that creates an indicator function I_t such that $I_t = 1$ for instances in time where an error could have been made. We can then estimate the probability of having an error at time t as follows:

$$P(y_t \neq h_t | y_1^T \neq h_1^T) = \frac{\gamma^{I_t}}{\sum_{k=1}^T \gamma^{I_k}}, \tag{2}$$

where the parameter $\gamma \in [1, \infty)$ expresses our confidence in the accuracy of I_t. A value of 1 will cause the probability of an error to be the same for all moments in time, irrespective of the value of I_t, while when γ approaches infinity we have absolute confidence in the inferred locations. Similar relations can be defined for the exponential prior and they can be obtained through its convolution with equation (2).

In order to apply boosting to temporal data, where classification decisions are made at the end of each sequence, we use a set of weights $\{\psi_t\}$ corresponding to the set of frames in an example sentence. At each boosting iteration j the weights are adjusted through the use of (2), resulting in the following recursive relation:

$$\psi_{t,j+1} = \frac{\psi_{t,j} \gamma^{I_t}}{\sum_{k=1}^T \psi_{k,j} \gamma^{I_k}} \tag{3}$$

4. SPEECH RECOGNITION WITH HIDDEN MARKOV MODELS

This section quickly reviews how Hidden Markov Models (HMMs) are used for the task of speech recognition. Let $X = \{x_1, x_2, \ldots, x_t\}$ be a sequence of acoustic frames representing a sentence $S = \{s_1, s_2, \ldots, s_n\}$.

HMMs introduce a (hidden, unknown) state variable Q and factor the joint distribution $p(X, Q)$ using two simpler distributions, namely emission distributions $p(x_t | q_t)$ and transition distributions $p(q_t | q_{t-1})$. Such factorisation yields efficient training algorithms such as the Expectation-Maximisation algorithm (EM) [7] which can be used to maximise the likelihood of the acoustic data X.

The success of HMMs applied to speech recognition is based on a careful design of sub-models (distributions) m_j corresponding to lexical units (phonemes, or words). Given a training set of acoustic sequences representing sentences, for which we know the corresponding labelling in terms of phonemes (but not necessarily the precise alignment), we create a new HMM for each sequence as the concatenation of sub-model HMMs m_j corresponding to the sequence of phonemes. This new HMM can then be trained using EM and will have the effect of adapting each sub-model HMM accordingly.

When a new sequence of acoustic features corresponding to a sentence becomes available, the objective is to obtain the optimal sequence of sub-model HMMs $\hat{S} = \{\hat{s}_1, \ldots, \hat{s}_{\hat{n}}\}$ (representing phonemes) that could have generated the given observation sequence. An approximation of this can be done efficiently using the well-known Viterbi algorithm [8] applied on an HMM which allows for all possible sequences of words (or phonemes).

This training process maximizes the likelihood of the acoustic sequence X given the correct corresponding sentence S, while it may be more interesting to directly minimize the error between the obtained sentence $\hat{S}$ and the desired sentence S given the acoustic sequence X. This error is in general measured using the well-known *word error rate* which computes the edit distance between S and $\hat{S}$ in terms of words.

When the information is coming from multiple streams, or when there are multiple hypothesis models available, it can be advantageous to employ multi-stream decoding techniques [9]. In multi-stream decoding each sub-unit model m_j is comprised of n_e sub-models $m_j = \{m_j^i | i \in [1, n_e]\}$ associated with the sub-unit level at which the recombination of the input streams should be performed.

We consider the case of state-locked multi-stream decoding, where all sub-models are forced to be at the same state. In that case, the simplest multi-stream strategy is to use the Viterbi algorithm in an HMM where the emission distribution are estimated as follows:

$$p(x_t|q_t) = \sum_{i=1}^{n_e} w_i \cdot p_i(x_t|q_t) \tag{4}$$

where $p_i(x_t|q_t)$ represents the emission distribution of sub-model m_i at time t and w_i represents the reliability of expert e_i.

5. BOOSTING HMMS

Boosting had originally been defined for classification tasks, and recently it was generalised to the regression case (see [4] for an overview). However, the amount of research in the application of boosting to sequence learning has been comparatively small. In [10] boosting was used in a speech recognition task. That particular system was HMM-based with ANNs for computing the posterior phoneme probabilities at each state. The boosting itself was performed at the ANN level, using AdaBoost with confidence-rated predictions and in which the sample loss function was the frame error rate. The resulting decoder system differed from a normal HMM/ANN hybrid in that each ANN was replaced by a mixture of ANNs. Boosting was also applied in a similar context in [3]. The authors of the latter further describe a word error rate boosting scheme, which, however does not manage to produce as good results as the other described schemes. In our view, which is supported by the experimental results, this could have been partially due to the lack of a temporal credit assignment mechanism such as the one we present in section 3.

The work presented here explores the use of boosting for HMMs that employ Gaussian Mixture models at the state level. Similarly to [3], we use sentence-level labels as the basic error measure. However, we employ a probability distribution over the frames during training, as discussed in Section 3.

With respect to the sentence-level error measure, we needed something related to the word error rate. However the word error rate for any particular sentence takes values in $[-1, \infty)$. For this reason we employ a bijective map $f : [0, \infty) \to (-1, 1]$

$$f(x) = 2e^{-\eta x} - 1, \tag{5}$$

where x is the word error rate. When $f(x) = 1$, an example is considered as classified correctly and when $f(x) = -1$, the example is considered to be classified incorrectly. Increasing the parameter η increases the sharpness of the transition. This function is used for $l(\cdot)$ in equation (1).

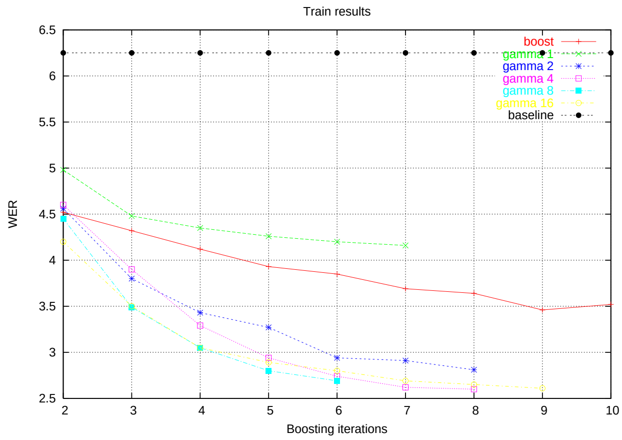

Fig. 1. Training word error rates for various values of gamma, compared with a baseline system and the previous boosting approach.

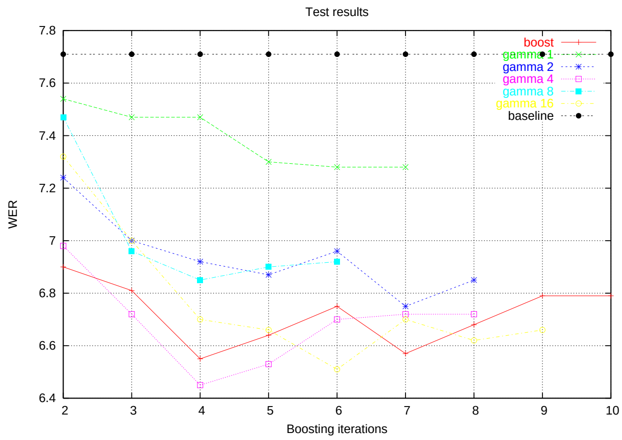

Fig. 2. Test word error rates for various values of gamma, compared with a baseline system and the previous boosting approach.

6. EXPERIMENTAL RESULTS

We experimented on the OGI Numbers 95 (N95) data set [11]. This data set was converted from the original raw audio data into a set of features based on Mel-Frequence Cepstrum Coefficients (MFCC) [12] (with 39 components, consisting of three groups of 13 coefficients, namely the static coefficients and their first and second derivatives) that were extracted from each frame. The data contains 27 distinct phonemes, consisting of 3233 training utterances and 1206 test utterances. The feature extraction and phonetic labelling is described in more detail in [13].

The HMMs were composed of three hidden states in a left-to-right topology and the distributions corresponding to each state were modelled with a Gaussian Mixture Model employing 10 Gaussian distributions. [1]

[1] These values are within the range commonly employed in speech recognition tasks where the data is composed of MFCC features. There was no attempt to perform cross-validation in order to choose those hyper-parameters optimally.

The experiment was performed as follows: firstly, a set of HMMs e_0, composed of one model per phoneme, was trained using the automatically generated phonetic labels. This has the role of a starting point for the subsequent expert models. At each boosting iteration t we take the following steps: firstly, we sample with replacement from the distribution of training sentences. We create a new expert e_t, initialised with the parameters of e_0. The expert is trained on the sentence data using the Viterbi approximation. The frames of each sequence carry an importance weight ψ , which is factored into the training algorithm. After training, all sequences are decoded with the new expert. The weights of each sentence is increased according to (5), with $\eta = 10$. For each erroneously decoded sentence we calculate the edit distance using a shortest path algorithm. All frames in which the inferred state belonged to one of the words that corresponded to a substitution, insertion, or deletion are then marked. The weights of marked frames are adjusted according to (2). The parameter γ corresponds to how smooth we want the temporal credit assignment to be.

In order to evaluate the combined models we use the multi-stream method described in equation (4), where the weight of each stream is given by

$$w_i = \frac{\beta_i}{\sum_j^{n_e} \beta_j}. \tag{6}$$

In this setting, unlike usual multi-stream settings, the same data is used in each and every stream. However, a different model corresponds to each stream. [2]

Experimental results comparing the performance of the above techniques to that of an HMM using segmentation information for training are shown in Figure 1, for the training data and figure 2 for the test data. The figures include results for the baseline system and our previous results with boosting at the phoneme level. We have included results for values of $\gamma \in \{1, 2, 4, 8, 16\}$. Although we do not improve significantly upon our previous work with respect to the generalisation error, we found that the convergence of boosting in this setting is significantly faster. While boosting with pre-segmented phoneme examples had previously resulted in a reduction of the error to 3% after approximately 30 iterations, the sentence example training, combined with the error probability distribution over frames, converged to the same error after approximately 6 iterations.

7. DISCUSSION

In this paper we presented a method for the application of boosting to complete HMMs, rather than at the frame level. State-locked multi-stream decoding techniques were investigated for model recombination in a continuous speech recognition task. We observed a significant reduction in training error and a reduction in generalisation error with a statistical significance level of 85%, the same as our previous approach.

While the two boosting approaches are equivalent in the latter respect, in our view the sentence training approach represents a more interesting alternative, for a number of reasons. Firstly, we are minimising the word error rate directly, which is a more principled approach since we optimise the real objective. Secondly, we don't in principle need to rely on segmentation information. Lastly, the temporal probability distribution, derived from the temporal structure of word errors and the state inference, provides us with a method to assign weights to parts of the decoded sequence. Its importance becomes obvious when we compare the performance of the method for various values of γ. When the distribution is flat (i.e. when $\gamma = 1$), the performance of the model drops significantly.

In the near future we will attempt to merge the two parameters η and γ, so that we have a unified framework for sequential decision making and boosting. We hope to achieve this by expressing the sample loss of a sentence as an expected value and relating it to the probability of errors at the frame level.

[2] In an unconstrained multi-stream setting we would have a model whose state-space would correspond to the Cartesian product of the state-spaces of individual sub-models. However, because of computational constraints, we consider a sub-space where all models are constrained to be at the same state.

8. REFERENCES

[1] Christos Dimitrakakis and Samy Bengio, "Boosting hmms with an application to speech recognition," in *IEEE International Conference on Acoustic, Speech, and Signal Processing, ICASSP*, 2004, IDIAP-RR 03-41.

[2] L.R. Bahl, P.F. Brown, P.V. de Souza, and R.L. Mercer, "A new algorithm for the estimation of hidden markov model parameters," in *IEEE Inernational Conference on Acoustics, Speech and Signal Processig, ICASSP*, 1988, pp. 493–496.

[3] G. Cook and A. Robinson, "Boosting the performance of connectionist large vocabulary speech recognition," in *Proc. ICSLP '96*, Philadelphia, PA, 1996, vol. 3, pp. 1305–1308.

[4] Ron Meir and Gunnar Rätch, "An introduction to boosting and leveraging," in *Advanced Lectures on Machine Learning*, LNCS, pp. 119–184. Springer, 2003.

[5] Robert E. Schapire and Yoram Singer, "Improved boosting algorithms using confidence-rated predictions," *Machine Learning*, vol. 37, no. 3, pp. 297, 1999.

[6] Yoav Freund and Robert E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum-likelihood from incomplete data via the EM algorithm," *Journal of Royal Statistical Society B*, vol. 39, pp. 1–38, 1977.

[8] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, pp. 260–269, 1967.

[9] A. Morris, A. Hagen, H. Glotin, and H. Bourlard, "Multi-stream adaptive evidence combination for noise robust ASR," *Speech Communication*, 2001.

[10] H. Schwenk, "Using boosting to improve a hybrid HMM/neural network speech recogniser," in *Proc. ICASSP '99*, 1999, pp. 1009–1012.

[11] R. A. Cole, K. Roginski, and M. Fanty, "The OGI numbers database," Tech. Rep., Oregon Graduate Institute, 1995.

[12] Lawrence R. Rabiner and Biing-Hwang Juang, *Fundamentals of Speech Recognition*, PTR Prentice-Hall, Inc., 1993.

[13] Johnny Mariéthoz and Samy Bengio, "A new speech recognition baseline system for numbers 95 version 1.3 based on torch," IDIAP-RR 16, IDIAP, 2004.