Frame-level AnyBoost for LVCSR with the MMI Criterion

Ryuki Tachibana[1], Takashi Fukuda[1], Upendra Chaudhari[2], Bhuvana Ramabhadran[2], and Puming Zhan[3]

[1] *IBM Research – Tokyo, IBM Japan Ltd, Yamato, Japan*
{ryuki,fukuda1}@jp.ibm.com

[2] *IBM T. J. Watson Research Center, Yorktown Heights, NY, USA*
{uvc, bhuvana}@us.ibm.com

[3] *Nuance Communications Inc., Boston, MA, USA*
Puming.Zhan@nuance.com

***Abstract*—This paper propose a variant of AnyBoost for a large vocabulary continuous speech recognition (LVCSR) task. AnyBoost is an efficient algorithm to train an ensemble of weak learners by gradient descent for an objective function. We present a novel training procedure that trains acoustic models via the MMI criterion using data that is weighted proportional to the summation of the posterior functions of previous round of weak learners. Optimized for system combination by n-best ROVER at runtime, data weights for a new weak learner are computed as a weighted summation of posteriors of previous weak learners. We compare a frame-based version and a sentence-based version of our proposed algorithm with a frame-based AdaBoost algorithm. We will present results on a voice search task trained with different amounts of data with gains of 5.1% to 7.5% relative in WER can be obtained by three rounds of boosting.**

I. Introduction

A review of learning algorithms that construct an ensemble of classifiers and subsequently use a weighted vote of their decisions for classifying test data is given in [1], [2]. For the ensemble to be better than any of the individual classifiers, they need to be accurate and diverse. Random Forests, introduced by Breiman [3] belong to this family of ensemble methods. The main principle here is to combine many binary decision trees, built by randomly choosing at each node a subset of predictors. The randomized decision tree based approach to build an ensemble of classifiers was introduced in [4] and successfully applied to large vocabulary continuous speech recognition (LVCSR) in [5]. In Bootstrap Aggregating, known as Bagging [6], multiple versions of a predictor (classifier) are used to get an aggregated predictor. Each of the classifier's are trained using the same amount of data obtained by sampling with replacement. Bagging improves performance of unstable classifiers. Boosting is yet another technique [7] that uses all of the training data for each classifier but weights the data to reflect its importance such that each classifier focuses on a different aspect of the data yielding diverse classifiers. All these approaches are aimed at combining classifiers to further improve the performance of each individual classifier.

Boosting algorithms have been popular in the field of machine learning, and recently they have gained popularity in large vocabulary continuous speech recognition (LVCSR) [8]. Boosting is a technique to generate an accurate statistical model based on the combination of many simple and less accurate models, which are usually called base or weak learners [9]. One of the most popular boosting algorithms, AdaBoost [7] uses the classification accuracy of the models learnt at each training iteration of the boosting algorithm to reweight the training samples for the successive round of training. The Gradient Boosting algorithm [10] isolates the base learners from the loss function, by fitting the base learners to the negative functional gradient of the loss function in the least square sense. This allows for a wide variety of loss functions. Recently, gradient boosting has been applied in the context of robust training of Hidden Markov Models for ASR [11], [12]. Stochastic gradient boosting [13] introduces another degree of freedom by randomly subsampling the training data at each iteration of the training process.

Zhang et al. presented AnyBoost-based training scheme [14] that uses the MCE discriminative criterion for constructing ensembles [11]. The proposed method derived the weights of the data that decreases the MCE-based loss function using gradient descent. Successive models in the ensemble were trained based on the weighted data distribution. Saon and Soltau [12] proposed a method based on AdaBoost, where the weights of each data sample in the training data were determined based on whether or not the previous ensemble chose the correct hypothesis for the frame. The acoustic models thus obtained were then discriminatively trained using the boosted MMI criterion [15] with the newly weighted data. In this paper, we present a gradient boosting algorithm based on the MMI discriminative criterion for the training of an ensemble of acoustic models.

The rest of this paper is organized as follows. We first review n-best ROVER, AdaBoost, and AnyBoost which are the bases of our method in Section II. Based on these algorithms, we describe our version of gradient boosting in Section III.

Section IV describes the experiments and demonstrates the effectiveness of the proposed algorithm using corpora of two different sizes. We draw conclusions in Section V.

II. Background

Our objective is to reduce the WER of an LVCSR system by using an ensemble of acoustic models. While increasing the number of acoustic models in the ensemble generally leads to WER reduction to some extent, it also increases the amount of necessary computational resources both for training and test conditions. This is problematic especially when the training corpus is huge or when the run-time system requirements impose time and memory constraints. To achieve a maximal WER reduction with a limited number of the models in the ensemble, the models should be designed so that the mutual complementarity of the models is optimal for system combination techniques to be used at run-time. We design our boosting algorithm so that the models optimally work with the n-best ROVER algorithm [16]. In this section, after reviewing the ROVER algorithm, we review two important boosting algorithms which are the bases of our algorithm, AdaBoost and AnyBoost.

A. n-best ROVER

ROVER [17] is a system combination technique that aims to reduce word error rates by exploiting differences in the errors made by multiple ASR systems. While the standard ROVER algorithm uses only the 1-best hypotheses of the multiple systems, the "n-best ROVER" algorithm [16] combines the n-best lists of the systems. It first estimates word posterior probabilities by normalizing the likelihood of the hypothesis over those of all hypotheses.

$$P_t(y|x) \quad = \quad \frac{P_t(x|y)P(y)}{\sum_{\hat{y}} P_t(x|\hat{y})P(\hat{y})}, \tag{1}$$

where x is a sequence of acoustic feature vectors, y is a hypothesis, and t is the index of the systems to be combined. Then, it computes a combined posterior as a linear combination of the posteriors using

$$P(y|x) \quad = \quad \sum_t \alpha_t P_t(y|x), \tag{2}$$

where α_t are system weights. As in the case of 1-best ROVER, complementarity (diversity) of the component systems is the key factor for successful combination by the n-best ROVER algorithm.

B. AdaBoost

Saon and Soltau applied the AdaBoost algorithm to LVCSR [12]. Each round of boosting trained a new weak learner based on a training data distribution modified by multiplying data weights. A weak learner is a full acoustic model including a phonetic decision tree. The algorithm worked at the frame level; it assigned a data weight to each frame of the training data. After each round of boosting, all the utterances in the training corpus were decoded with the latest weak learner. Then, the data weight, $D_t(i)$, of each frame was increased or decreased depending on whether the decoding result for the frame was incorrect or correct.

$$D_{t+1}(i) \quad = \quad \frac{D_t(i)}{Z_t} \cdot \begin{cases} \beta, & (\text{if } h_t(x_i) = y_i) \\ 1, & (\text{otherwise}) \end{cases}, \tag{3}$$

where β is a constant weight decaying factor, which was tuned on a held-out set. $h_t(x_i)$ is the decoding result for the i-th frame by the t-th weak learner. Z_t normalizes D_{t+1}. The next round of boosting trains the next weak learner with this updated data distribution.

The decoding performances of the weak learners on the training corpus were also used to determine the system weights, α_t, of the weak learners, which were calculated by

$$\alpha_t \quad = \quad \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right), \tag{4}$$

where ϵ_t is the frame error rate of the t-th weak learner. During decoding of the test data (run-time), the algorithm combines the weak learners by calculating the weighted summation of the acoustic scores from the weak learners using these system weights. While 4-system combinations by the algorithm reduced the WER by 4.0% relative for models trained via the maximum-likelihood (ML) criterion and by 4.8% relative for the discriminatively-trained models, the WER reductions for a 2,000 hour training corpus were limited to 3.1% and 1.9% relative for ML and discriminatively-trained models respectively.

C. AnyBoost

The AnyBoost algorithm described in [14], [11] trains an ensemble, $F_T(x, y)$ that minimizes a given loss function $L(F_T)$. The ensemble is defined as a weighted summation of weak learners, $f_t(x, y)$.

$$F_T(x, y) = \sum_{t=1}^{T} \alpha_t f_t(x, y). \tag{5}$$

Each round of AnyBoost trains a new weak learner, f_t, in the direction in which the loss function, $L(F_{t-1} + \alpha_t f_t)$, decreases most rapidly. Here, where F_{t-1} is the ensemble of the weak learners trained in the previous rounds. The direction is sought by training the new weak learner f_{t+1} that maximizes, $\langle -\nabla L(F_{t-1}), f_t \rangle$, the inner product of the new learner with the gradient of the loss function. Since the loss function is defined as the summation of the loss functions of the training data,

$$L(F_t) \quad = \quad \frac{1}{N}\sum_{i=1}^{N} L_i(Ft(x_i, y_i)), \tag{6}$$

the inner product determines the data weights $w_t(i)$ for the next round.

$$\begin{aligned} &\langle -\nabla L(F_{t-1}), f_t \rangle \\ &\quad = \quad -\sum_{i=1}^{N}\left[\frac{\partial L_i(F_{t-1})}{\partial F_{t-1}(x_i, y_i)} f_t(x_i, y_i)\right] \qquad (7) \\ &\quad = \quad \sum_{i=1}^{N}\left[w_{t-1}(i) f_t(x_i, y_i)\right]. \qquad (8) \end{aligned}$$

After training the new weak learner, its system weight, α_{t+1}, is determined via a line search.

In [11], Zhang et al. use the MCE discriminative objective function as the loss function, and the posterior probability $P(y|x)$ of an acoustic model as the weak learner, $f(x, y)$. This definition of the weak learner as the posterior probability is a good match with the subsequent ROVER combination strategy. The definition of the ensemble $F(x, y)$ is given below:

$$F_T(x,y) = \sum_{t=1}^{T} \alpha_t P_t(y|x) \quad (9)$$

$$= \sum_{t=1}^{T} \alpha_t \frac{P_t(x|y)P(y)}{\sum_{\hat{y}\in Y} P_t(x|\hat{y})P(\hat{y})} \quad (10)$$

The experimental results in [11] based on a relatively small training corpus with 30k utterances showed 3.7 % relative reduction in WER after six rounds of the AnyBoost algorithm. This was not significantly better than the gains from the AdaBoost algorithm despite the use of functional gradient descent for the MCE criterion. It may be possible that the data weighting at the sentence level impairs the advantages of the AnyBoost algorithm.

III. Boosting algorithms

We apply the AnyBoost algorithm to LVCSR at the frame level. We assume that acoustic models to be combined are discriminatively trained using the MMI criterion [15] and that n-best ROVER is used to combine the acoustic models at runtime. The definition of the ensemble in [11] as the weighted summation of the posterior functions (Eq. (9)) matches very well with our posterior-based ROVER (Eq. (1)) and the MMI-trained models. Although, the MCE loss function in [11] should have better discriminative power as it discriminates sequences, we use a simpler loss function in this paper,

$$L(F_t) = \sum_{i=1}^{N} -\log F_t(x_i, y_i). \quad (11)$$

This function is motivated by the following equations and experimental results, which show that the discriminative power is derived from the use of posterior functions alone, without a discriminative loss function. In addition, the derived weights for our proposed algorithm match well with MMI training [15]. Since the functional derivative of the loss function is written as

$$\frac{\partial L_i(F)}{\partial F_t(x_i, y_i)} = -\frac{1}{F_t(x_i, y_i)}, \quad (12)$$

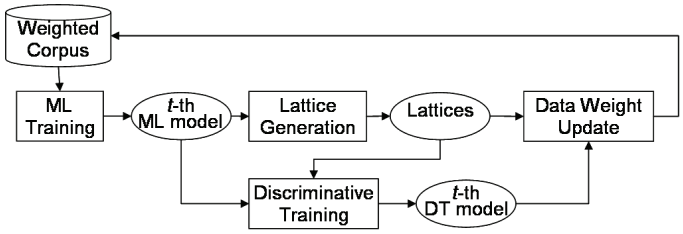

Fig. 1. Process flow.

the inner product (Eq. (7)) of the nabla of the loss function and a new weak leaner becomes

$$\langle -\nabla L(F_{t-1}), f_t \rangle = \sum_{i=1}^{N} \frac{1}{F_{t-1}(x_i, y_i)} f_t(x_i, y_i) \quad (13)$$

$$= \sum_{i=1}^{N} \frac{1}{F_{t-1}(x_i, y_i)} \frac{P_t(x_i|y_i)P(y_i)}{P_t(x_i)} \quad (14)$$

$$= \sum_{i=1}^{N} w_{t-1}(i) \frac{P_t(x_i|y_i)P(y_i)}{\sum_{\hat{y}_i} P_t(x_i|\hat{y}_i)P(\hat{y}_i)}. \quad (15)$$

This implies that training an acoustic model via the MMI criterion and the data weights, $w_{t-1}(i) \equiv 1/F_{t-1}(x_i, y_i)$, which is calculated based on the weighted summation of the posterior functions by the previous weak learners (Eq. (9)).

When applying this formula to LVCSR, the following points should be considered.

A. Competing hypothesis generation

Our AnyBoost algorithm requires a set of competing hypotheses for the calculation of the data weights $w_t(n)$. Meanwhile, the original procedure of MMI training also uses competing hypotheses, $\hat{y}_i$, for calculating $\sum_{\hat{y}_i} P_t(x_i|\hat{y}_i)P(\hat{y}_i)$ in the denominator of the MMI criterion [15]. One recipe that works very well uses lattices produced by decoding the training corpus with an ML acoustic model and a unigram language model [15]. We can use the lattices generated for the MMI training also for the data weight calculation of boosting. Of course, it is possible to separately generate another set of lattices by using the discriminatively trained acoustic models, and that is expected to better reflect the true performances of the previous weak learners. However, since lattice generation for a large training corpus takes very long time, there is a trade-off between the training time and the quality of the generated lattices. The entire process flow of our AnyBoost algorithm sharing the lattices for the two purpose is illustrated in Fig. 1 and in Algorithm 1.

B. Sentence-level and frame-level boosting

We can apply Eq. (15) to LVCSR either at the sentence level or the frame level. If we apply it at the sentence level, y_i denotes a sentence hypothesis, $P(x_i|y_i)$ the cumulative acoustic likelihood throughout the sentence, $P(y_i)$ the cumulative linguistic likelihood. If we use weighting at the frame level

Algorithm 1 AnyBoost for LVCSR

Input: Training data $\{(x_i, y_i)\}_{i=1}^{N}$
Initialize: $w_0(i) = 1/N$.

1: **for** $t = 1$ to T **do**
2: Train ML model using data weights $w_{t-1}(i)$.
3: Generate lattices using the ML model.
4: Train DT model f_t using the lattices and the data weights.
5: Choose system weight α_t for f_t.
6: Let $F_t = F_{t-1} + \alpha_t f_t$.
7: Calculate new data weights $w_t(i)$ using the lattices and the trained model f_t.
8: **end for**

instead, x_i denotes the acoustic feature vector of a frame. We define y_i as a leaf of the phonetic decision tree. By approximating every y_i having the same linguistic probability $P(y_i)$, the weak learner in Eq. 9 is rewritten as

$$f_t(x_i, y_i) \approx \frac{P_t(x_i|y_i)}{\sum_{\hat{y}\in Y_{i,\text{den}}} P_t(x_i|\hat{y})} \quad (16)$$

$$\approx \frac{\sum_{\hat{y}\in Y_{i,\text{num}}} P_t(x_i|\hat{y})}{\sum_{\hat{y}\in Y_{i,\text{den}}} P_t(x_i|\hat{y})}, \quad (17)$$

in which a summation of likelihoods is also calculated for the numerator lattices since the exactly correct phonetic alignments is unknown. We generate the numerator lattices by using the correct transcript of the sentence. $P_t(x|y)$ is a likelihood of the acoustic feature vector x with respect to the GMM that corresponds to the context-dependent state y of the phonetic decision tree of the t-th model. Y_i is a set of such states assigned to the i-th frame. To ensure that the denominator state set $Y_{i,\text{den}}$ always be larger than the numerator state set $Y_{i,\text{num}}$, the context-dependent states in the numerator should be added to the denominator state set.

C. Dynamic range of data weights

An example of the distribution of the data weights calculated by Eq. (14) for the frame-level AnyBoost is shown in Fig. 2. First, we can see from the figure that the majority of the frames are given a weight of 1. This is because even for the hypotheses that are wrong at the sentence level, the number of the frames contained in wrong word hypotheses is limited. We can also see in the figure that the distribution has a very long tail. This may cause a handful of frames to have very large weights and that majority of the frames to have relatively very small weights. This is expected to significantly reduce robustness against wrong transcriptions in the training corpus. Therefore, to avoid this, we use a scaling factor, γ, and a threshold, T_w.

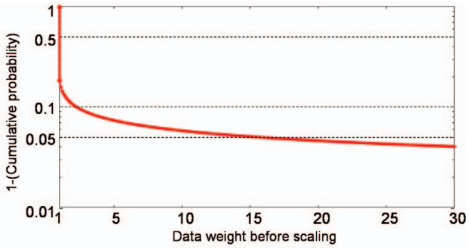

Fig. 2. Distribution of data weights before scaling, shown by (1 − (cumulative probability)).

With these parameters, the data weights are adjusted by

$$w'_t(i) = \begin{cases} w_t(i)^\gamma & (w_t(i) \le T_w) \\ T_w^\gamma & \text{otherwise} \end{cases}. \quad (18)$$

IV. Experiments

A. Baseline Models

The experiments presented in this paper are all based on speaker independent models that are discriminatively trained on a large vocabulary continuous speech recognition task, namely, voice search. We present results on an in-house test set for voice search in English. To date, no standardized test exists in the community to benchmark systems for the voice search task. However, similar tasks have been studied in the literature [18], [19] where the baseline systems range in WER's from 16% to 25%. The acoustic models are built on data from several hundreds of speakers with the data ranging from a few seconds to few hours per speaker.

The raw acoustic features for transcription are 13-dimensional PLP features, including c0. Utterance level mean normalization, where the statistics are calculated only on the speech regions of the data, is used throughout the Maximum Likelihood (ML) and discriminative training process. This is because a significant portion of the data was non-speech, with about 40% silence. In ML training, LDA+MLLT transforms are generated by splicing 9 frames of PLP features and reducing the feature vector to the 40-dimensional feature space. We build two sets of acoustic models trained from different quantities of data. Training set A comprises of about 150 hours and training set B comprises of an order of magnitude more data i.e. over 5000 hours (Table I). The ML models for set A contain roughly 150K Gaussians with 5000 quinphone context-dependent states while the ML models for set B contain roughly 600K Gaussians with 20K states. After ML training, the models are discriminatively trained using the boosted MMI criterion [15]. Results are presented on 2 test sets: **Dev**, and **Eval** with approximately 4K and 70K words. We limit our experiments in this paper to three rounds of gradient boosting given that the computational time required for training increases almost proportionally to the number of the weak learners. N-best ROVER [16] was used to combine the acoustic models.

Table I tabulates the baseline performance. Consistent with what has been reported in the literature, an order of magnitude

TABLE I
BASELINE WER ON DISCRIMINATIVELY TRAINED MODELS.

	WER%	
Training Sets	Dev	Eval
Set A	24.0	25.3
Set B	20.8	21.4

TABLE II
SYSTEM WEIGHTS AND THEIR RESULTS FOR VARIOUS CONDITIONS.

	AdaW		**AnyW**	
	Weights α_t	*WER%*	Weights α_t	*WER%*
Cond. 1	$\{0.70, 0.30\}$	24.4	$\{0.79, 0.21\}$	24.4
Cond. 2	$\{0.73, 0.27\}$	24.0	$\{0.82, 0.18\}$	24.1
Cond. 3	$\{0.53, 0.23, 0.24\}$	24.0	$\{0.70, 0.16, 0.14\}$	24.4

increase in the amount of training data leads to approximately 14 to 15.5% relative improvement in the Word Error Rate (WER). Several training methodologies to derive benefits from additional training data have been presented in the literature with a consistent message that one has to carefully select the most beneficial parts and find a model that best represents the selected data.

B. Preliminary Experiments

We conducted a few preliminary experiments to help set parameters in the proposed algorithm and held-out data. Based on these experiments, the scaling parameter γ and the threshold T_w were set to be 1.

1) System weights : While our algorithm determines data weights using AnyBoost, we can choose a system weight computation strategy independently. In this first set of preliminary experiments presented in Table II, we compare the performances of two strategies of computing system weights. The first strategy (**AdaW**) computes the system weights by using Eq. (4) of AdaBoost. It is based on the frame error rates of the weak learners on the training corpus. The second strategy (**AnyW**) calculates the optimal α_t that maximizes Eq. (5) based on the numerator scores and the denominator scores of the weak learners in the training corpus. We compared the two strategies for various conditions. Examples of the experimental results are shown in Table II.

As we can see in the table, **AnyW** tends to give extreme weights for the weak learners in these experiments. The results of **AdaW** were equal or better than those of **AnyW** in most cases. Therefore, we decided to use **AdaW** for the following experiments. Since additional computational time to calculate the frame error rate for the entire training corpus is expensive, we used a random subset of the training corpus for these computations.

2) Complementarity : In the next set of preliminary experiments, we investigated the complementarity of the trained models. We decoded a subset of the training corpus by each of the three weak learners and analyzed the overlap in the decoded errors. Fig. 3 illustrates the distribution of the frame errors by the three weak learners (f_1,f_2,and f_3) for (a) frame-level AdaBoost and (b) frame-level AnyBoost. The number in a circle is the percentages of the frames which the weak learner failed to decode. For example, we can see in Fig. 3b that, the percentage of the frames which f_1 could not correctly decoded was $2.2 + 1.5 + 5.7 + 0.8 = 10.2\%$. While neither f_2 nor f_3 could not decode 5.7% frames among these frames, remaining $2.2+1.5+0.8 = 4.5\%$ was successfully covered by either or both of f_2 and f_3. Although it is difficult to compare which of the two algorithms, AdaBoost and AnyBoost is better in generating complementary models based solely on these numbers, we can see that AnyBoost also succeeded in capturing many frame errors made by the previous weak learners, even though AnyBoost does not explicitly use frame errors in data weight calculation.

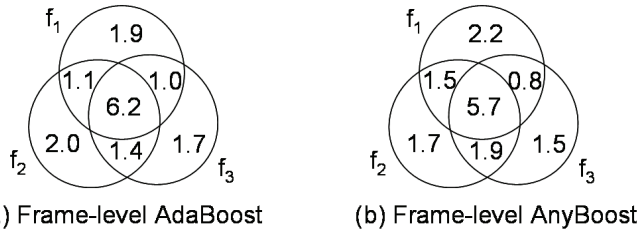

Fig. 3. Frame error rates [%] of the weak learners, f_t.

C. Experiments with Training Set A

We compared the performances of models trained using Set A via the three boosting methods: (1) frame-level AdaBoost, (2) sentence-level AnyBoost, and our proposed algorithm, (3) frame-level AnyBoost. These methods were only different in the equations presented in Section III to calculate the weights of the data. We used only three boosting rounds for each of the algorithms. This was because the amount of the computational time required for both of the training and the runtime increases almost proportionally to the number of the weak learners. The n-best ROVER [20] was used to combine the acoustic models. Since the first rounds of these algorithms effectively train a boosted MMI model based on the training corpus with all the data weighted equally (or unweighted data), we used the baseline model as the first model in these algorithms and train subsequent rounds of boosting. The results are shown in Table III. The numbers in the *Sngl* columns are the WERs of the weak learners trained in each round of boosting. The numbers in the *nRV* columns, which are more important, are the WERs obtained after system combination of the baseline system with the weak learners up to the mentioned rounds of boosting. These systems were combined using n-best ROVER. For example, we can see that frame-level AnyBoost has a WER of 22.2% after nbest ROVER using three rounds of gradient boosting on the development test set, which is 7.5% relative improvement over the baseline model (24.0%).

We also compared two strategies for generating competing hypothesis for the frame-level AnyBoost algorithm: (1) lattices generated by using ML models and shared with boosted MMI training, and (2) lattices generated using discriminatively trained models exclusively for calculating AnyBoost data weights. The WERs for the two strategies were almost the

same. Hence, we can share the lattices to save computational time without a significant increase in WER.

TABLE III
COMPARISON OF THREE BOOSTING ALGORITHMS FOR THE DEVELOPMENT SET. *Sngl* ARE WERS OF SINGLE SYSTEMS, WHILE *nRV* ARE OF N-BEST ROVER.

	Frame-level AdaBoost		Sentence-level AnyBoost		Frame-level AnyBoost	
	Sngl	*nRV*	*Sngl*	*nRV*	*Sngl*	*nRV*
Round 1	24.0	-	24.0	-	24.0	-
Round 2	25.1	23.2	25.7	23.4	24.8	22.7
Round 3	24.6	23.2	26.0	23.6	24.6	22.2

D. Experiments with Training Set B

To validate the effectiveness of the frame-level AnyBoost for larger training data, we conducted experiments using the larger training corpus with an order of magnitude more data. Recall that the baseline model is the first round model in the boosting step. The number of iterations used for discriminative training (BMMI) was tuned on the development set. While we used four iterations for boosted MMI training in the experiments with Training Set A, we used more iterations with Training set B until no significant improvement in performance was obtained. The WERs of the single systems are also shown in the *Sngl* columns. The proposed algorithm achieved 6.3% and 5.1% relative reductions in WER compared to the baseline boosted MMI model for **Dev** and **Eval** sets, respectively. These experimental results show that the algorithm is effective for larger training data sets and scales well.

TABLE IV
RESULTS OF FRAME-LEVEL ANYBOOST TRAINED ON SET B. *Sngl* ARE WERS OF SINGLE SYSTEMS, WHILE *nRV* ARE OF N-BEST ROVER.

	Dev		**Eval**	
	Sngl	*nRV*	*Sngl*	*nRV*
Round 1	20.8	-	21.4	-
Round 2	21.2	20.0	21.6	20.7
Round 3	20.3	19.5	21.0	20.3

V. CONCLUSION

The objective of this research work is to reduce the WER of an LVCSR system through an ensemble of a small number of weak learners. In this paper, we reported that a frame-level AnyBoost algorithm achieved 7.5% WER reduction after three rounds of boosting and outperformed frame-level AdaBoost and sentence-level AnyBoost for a smaller training corpus. The experiments with a larger corpus validated the effectiveness of the algorithm also for large corpora. The improvement reached 6.3% by increasing the number of MMI iterations. The experimental results and the derivation of the data weights for boosting proved to be an excellent combination for the n-best ROVER algorithm, the AnyBoost algorithm, and acoustic models trained with the MMI criterion. Our future work will include comparisons with other loss functions such as the MCE criterion. We would also like to explore a less heuristic method for determining the scaling parameter and the threshold for converting acoustic scores to data weights.

REFERENCES

[1] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," in *Bagging, boosting, and randomization. Machine Learning*, 1998, pp. 139–157.

[2] ——, "Ensemble methods in machine learning," in *Proc. First International Workshop on Multiple Classifier Systems*, 2000, pp. 1–15.

[3] L. Breiman and E. Schapire, "Random forests," in *Machine Learning*, vol. 45, no. 1, 2001, pp. 5–32.

[4] O. Siohan, B. Ramabhadran, and B. Kingsbury, "Constructing ensembles of asr systems using randomized decision trees," in *Proc. ICASSP*, 2005, pp. 197–200.

[5] B. Ramabhadran, O. Siohan, L. Mangu, G. Zweig, M. Westphal, H. Schulz, and A. Soneiro, "The IBM 2006 speech transcription system for European parliamentary speeches," in *TC-STAR Workshop on Speech-to-Speech Translation*, Barcelona, Spain, Jun. 2006, pp. 111–116.

[6] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[7] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," pp. 148–156, 1996.

[8] G. Saon, H. Soltau, U. Chaudhari, S. Chu, and B. Kingsbury, "The IBM 2008 GALE arabic speech transcription system," in *Proc. ICASSP*, 2010, pp. 4378–4381.

[9] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, pp. 119–139, August 1997. [Online]. Available: http://portal.acm.org/citation.cfm?id=261540.261549

[10] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[11] R. Zhang and A. I. Rudnicky, "Investigations of issues for using multiple acoustic models to improve continuous speech recognition," in *Proc. Interspeech*, 2006, pp. 529–533.

[12] G. Saon and H. Soltau, "Boosting Systems for LVCSR," in *Proc. Interspeech*, 2010, pp. 1341–1344.

[13] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367 – 378, 2002.

[14] L. Mason, J. Baxter, P. Bartlett, and M. Frean, "Boosting algorithms as gradient descent," in *Advances in Neural Information Processing Systems 12*. MIT Press, 2000, pp. 512–518.

[15] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, "Boosted MMI for model and feature-space discriminative training," in *ICASSP*, 2008, pp. 4057–4060.

[16] A. Stolcke, H. Bratt, J. Butzberger, H. Franco, V. R. R. Gadde, M. Plauche', C. Richey, E. Shriberg, K. So"nmez, F. Weng, and J. Zheng, "The SRI March 2000 Hub-5 conversational speech transcription system," in *Proc. NIST Speech Transcription Workshop*, 2000.

[17] J. G. Fiscus, "A Post-Processing System To Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER)," in *Proc. IEEE ASRU Workshop*, 1997, pp. 347–352.

[18] C. Chelba, J. Schalkwyk, T. Brants, V. Ha, B. Harb, W. Neveitt, C. Parada, and P. Xu, "Query language modeling for voice search," in *Proc. IEEE Workshop on Spoken Language Technology*, 2010.

[19] G. Dahl, D. Yu, L. Deng, and A. Acero, "Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition," *IEEE TSAP*, 2011.

[20] A. Stolcke, H. Bratt, J. Butzberger, H. Franco, V. R. R. Gadde, M. Plauche', C. Richey, E. Shriberg, K. So"nmez, F. Weng, and J. Zheng, "The SRI March 2000 Hub-5 conversational speech transcription system," in *Proc. NIST Speech Transcription Workshop*, 2000.