COMBINATION AND JOINT TRAINING OF ACOUSTIC CLASSIFIERS FOR SPEECH RECOGNITION

Katrin Kirchhoff and Jeff A. Bilmes

SSLI Laboratory
Department of Electrical Engineering
University of Washington
EE/CS, Box 352500, Seattle, WA, USA
{katrin,bilmes}@ssli.ee.washington.edu

ABSTRACT

Classifier combination is a technique that often provides significant improvements in accuracy, and also furnishes a useful mechanism to support multi-modal information sources. In this paper we discuss the problem of acoustic classifier combination in speech recognition systems. We present new techniques that generalize previously used combination rules, such as the mean, product, *min*, and *max* functions. These new rules have continuous and differentiable forms and can thus not only be used for combination of independently trained classifiers but also as objective functions in new joint classifier training schemes. We demonstrate the application of these rules to both combination and joint training using different input features, and we analyze their effects on word recognition accuracy. We find a significant word-error improvement over the product rule when jointly training and combining multiple systems using a generalization of the product rule.

1. INTRODUCTION

A challenge for automatic speech recognition (ASR) research is to develop systems that successfully utilize information from multiple modalities (such as vision, gestures, hand-writing, as well as the audio signal) or different representations or partitions of the same modality (e.g., different features extracted from the speech signal). This is necessary to embed ASR in advanced multimodal applications where the user interacts with computers simultaneously using several input modes (e.g., speech and gestures). The use of different speech signal representations itself can significantly enhance recognition robustness in adverse acoustic environments and thus has great potential for real-world applications such as wireless communication, speech recognizers for the car, and so on. A key issue for multi-modal/multi-representation research is to determine the best way to combine the information available in the different input representations so as to maximize performance with a minimum of computational effort.

Classifier combination can fuse together such different information sources whether they are multi-modal (such as speech and vision [27, 10]) or transformations [19, 18, 21] or (e.g., spectral) partitions [7, 26, 25] of the same signal.

Combining independently trained classifiers often produces appreciable gains, even when individual classifiers exhibit widely varying accuracies. This has been demonstrated for automatic speech recognition (ASR) [11, 12, 21] and for pattern classification [15, 16, 22, 30]. Combination rules often operate directly on classifier probabilities. One method (the mean rule) computes a weighted average of classifier outputs. Another method (the product rule) multiplies and then renormalizes these probabilities. Other techniques compute the maximum, minimum, or median of the classifier outputs [22]. Other methodologies combine using statistical models [5] or jointly train separate classifiers [28, 14].

In ASR, classifier combination can occur at different levels including the feature stream [3, 19, 12, 20], the HMM state [21], or at higher levels such as at the syllable [32] or sentence [11]. In ASR, HMM state-level combination (e.g., combining the outputs of phonetic class posterior probability estimators) has mostly used the mean or product rules. Moreover, classifiers are often trained separately without regard to joint optimization during training.

In this paper we present a variety of new combination rules which generalize the mean, product, *min* and *max* rules. These new rules are all continuous and differentiable. Not only can they be used for combination of independently trained classifiers, but also as objective functions in a joint classifier training scheme. We evaluate these new classifier combination schemes and joint training algorithms and present preliminary results with respect to continuous numbers recognition.

Section 2 reviews previous work on classifier combination in machine learning and in speech recognition. Section 3 presents our combination architecture, and serves to introduce notation used in this paper. Section 4 presents our new combination schemes. Section 5 develops the various joint training algorithms associated with each of the combination rules. Section 6 provides experimental results using baseline and our generalized rules. Final conclusions are presented in Section 7.

2. BACKGROUND: CLASSIFIER COMBINATION

The underlying goal of classifier combination theory is to identify the conditions under which the combination of an ensemble of classifiers yields improved performance compared to the individual classifiers. One widely investigated combination method is the mean rule, where the outputs of the individual classifiers are averaged:

$$P(c|x_1,...,x_N) = \sum_{n=1}^{N} \alpha_n P(c|x_n) \qquad (1)$$

where $P(c|x_n)$ is the probability for class c given by the n^{th} classifier, and α_n is the weight for the $n'th$ classifiers which uses feature vector x_n.

Classification is related to regression. In that case, several theoretical studies[16, 6] have shown that mean-rule combination is successful (a lower mean-squared error) when the errors of each system are independent. Error reduction is related to ensemble bias (the degree to which the averaged output of the ensemble of classifiers diverges from the true target function) and variance (the degree to which the ensemble members disagree) [23, 30, 6]. Generally, a low error requires both a low bias and variance, but since variance is reduced by averaging, it is sufficient to combine classifiers with low bias.

Tumer & Ghosh [31] have related the degree of correlation of classifier outputs to the ensemble error and have quantified it in terms of Bayes error. The total ensemble classification error E_t can be represented as the Bayes error E_b and the added ensemble-incurred error $\hat{E}_a$, where the relationship $E_t = E_b + \bar{E}_a$ holds. When combining unbiased correlated classifiers, the added error can be shown to be

$$\bar{E}_a = E_a \frac{1 + \rho(N-1)}{N}$$

where N is the number of classifiers, ρ a measure of error correlation, and E_a is the (common) added error of each individual classifier. As can be seen, the added error grows with the degree of classifier error correlation.

Producing ensemble members with decorrelated errors can be achieved by a variety of methods, e.g., training classifiers with different structures [31], varying the initial conditions from which classifiers are trained, training on disjoint [1] or partially overlapping data sets, specialized training schemes, using different input signals [1], or different feature representations of the input [15]. Popular combination methods include linear combinations of the output distributions (e.g., by averaging over, or multiplying, the outputs) [13, 22], combining the outputs by a higher-level classifier, Dempster-Shafer theory [29], and majority voting [24]. An alternative approach is to model a dependence between classification errors [6, 31, 5].

Many studies on classifier combination focus on what we call single-level classification. A typical single-level classification system consists of two components: a feature-extraction component which maps the input signal to feature vectors, and a classification component which assigns a class label to each feature vector. The desired classes are thus directly recognized from the feature space without any intermediate representation.

Speech recognition, however, requires a multi-level classification scheme because, for practical reasons, it is necessary to re-use classifiers at a lower level (such as for subphones, phones, or words) rather than attempting to discriminate between an inordinate number of classes (e.g., the number of possible sentences). In such case, the class C is related to the features $x_1, ..., x_N$ indirectly via some intermediate representation Q, and the system often makes simplifying conditional independence assumptions as in:

$$\begin{aligned} p(c|x_1,...,x_N) &= \sum_q p(c,q|x_1,...,x_N) \\ &= \sum_q p(c|q)p(q|x_1,...,x_N) \end{aligned}$$

Classifier combination can be applied at any intermediate stage in a multi-level classification system. The combination methods, however, should ideally take into account the requirements of the last stage (i.e., the Bayes error for the final class variable C). Successful combination methods in a multi-level classification system therefore might differ greatly from those combination methods which have proved beneficial for single-level classification.

Classifier Combination in Speech Recognition

Different partial recognition hypotheses can in principle be combined at either the feature, sub-phone, phone, word, or sentence level. In this study, we concentrate on combination at the phone level. Most approaches to phone-level combination have used different acoustic preprocessing techniques to generate an ensemble of classifiers trained on different feature spaces [19, 12, 20]; in some cases, the speech signal is split into a number of narrower frequency bands and the ensemble classifiers operate on individual subbands [26, 25]. Combination methods have in general used either the mean rule (Equation (1)) or the product rule: [1]

$$P(c|x_1,...,x_N) = \frac{\prod_{n=1}^{N} P(c|x_n)}{Z} \qquad (2)$$

where Z is a normalizing constant.

The mean rule is useful for combining uni-modal distributions into a single multi-modal distribution. Since mixing increases entropy [9], such a procedure is poor for representing low-entropy distributions where probability is concentrated in narrow input-space regions. In such cases, the product rule is useful, where each classifier must supply probability to the correct class, but may also supply probability to incorrect classes as long as one or more of the other classifiers do not supply probability to those incorrect classes. These are therefore called “AND” style combination schemes [21] since only the logical AND of each classifier’s probabilistic decision will survive combination. It is also the case that such a combination scheme is useful when the underlying distributions factorize over the probabilistic space of C [14].

Virtually all the aforementioned studies reported the largest performance increases for the product rule. This appears surprising since one may arrive at this rule by making

[1] This is of course equivalent to the mean rule applied to log probabilities.

the assumption of conditional independence of the input features given the output class [5]. This assumption is certainly not true in general — neither different feature representations derived from nor different spectral sub-bands of the same signal are conditionally independent given the class [2]. On the other hand, producing low-entropy distributions over HMM states from a product of sometimes incorrect classifiers might outweigh this inaccuracy. Alternatively, as argued in [4], an assumption that is incorrect for predictive accuracy does not ensure discriminative inaccuracy.

In previous work [21] we additionally investigated other rules such as the *max* rule:

$$P(c|\mathbf{x}_1, ..., \mathbf{x}_N) = \frac{max_n P(c|\mathbf{x}_n)}{\sum_{c=1}^{K} \max_n P(c|\mathbf{x}_n)} \tag{3}$$

and the *min* rule:

$$P(c|\mathbf{x}_1, ..., \mathbf{x}_N) = \frac{min_n P(c|\mathbf{x}_n)}{\sum_{c=1}^{K} \min_n P(c|\mathbf{x}_n)} \tag{4}$$

We found that significant error reductions occurred with the AND rules (product and $\min$) whereas the sum and the *max* rule (OR style rules) yielded only slight improvements and sometimes even worsened global performance.

In the following sections we present new AND-style combination rules which generalize the standard combination rules and can also be used as joint classifier training schemes.

3. BASIC COMBINATION ARCHITECTURE

This section describes our combination architecture. We use L neural network classifiers, each a multi-layer perceptron (MLP). Each classifier uses the multiple logistic[2] nonlinearity in the final layer:

$$z_k^l = \frac{exp(a_k^l)}{\sum_{k'} exp(a_{k'}^l)}$$

where z_k^l is the k^{th} output of the l^{th} classifier. These outputs are combined using one of the soon-to-be-defined combination rules

$$V_k = \text{combination\_rule}(z_k^1, z_k^2, \ldots, z_k^L).$$

The combined outputs are re-normalized, thereby producing the final combined system outputs

$$z_k = \frac{V_k}{\sum_j V_j}.$$

This architecture is depicted in Figure 1.

Under normal circumstances, each classifier is trained separately using the standard back-propagation algorithm. During testing the outputs of each of the sub-classifiers are combined using a combination rule, and then used in subsequent stages of classification. In Section 5, we will consider methods to jointly train the different classifier systems.

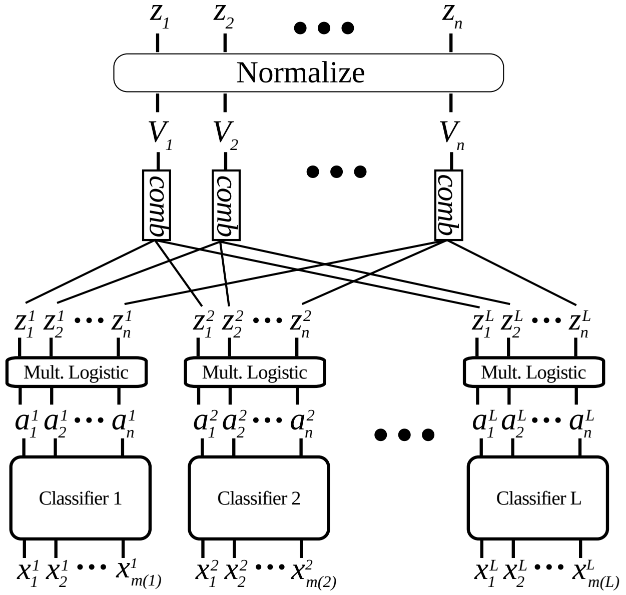
z_1 z_2 z_n
Normalize
V_1 V_2 V_n
comb
comb
comb
$z_1^1\, z_2^1 \cdots z_n^1$
$z_1^2\, z_2^2 \cdots z_n^2$
$z_1^L\, z_2^L \cdots z_n^L$
Mult. Logistic
Mult. Logistic
Mult. Logistic
$a_1^1\, a_2^1 \cdots a_n^1$
$a_1^2\, a_2^2 \cdots a_n^2$
$a_1^L\, a_2^L \cdots a_n^L$
Classifier 1
Classifier 2
Classifier L
$x_1^1\, x_2^1 \cdots x_{m(1)}^1$
$x_1^2\, x_2^2 \cdots x_{m(2)}^2$
$x_1^L\, x_2^L \cdots x_{m(L)}^L$

Figure 1: Architecture and notational definitions used in this paper. Input features x^l are presented to the l^{th} classifier. These produce linear outputs a^l which are then normalized using the multiple logistic function. For each output k of each of the L classifiers, the values are combined using one of the combination methods, and this produces value V_k. The result is once again normalized producing the final probability mass function z at the output of the combined system.

4. GENERALIZED COMBINATION RULES

This section presents new "soft" continuous rules that generalize the often used "hard" combination rules such as the mean (Eq 1), product (Eq 2), *min* (Eq 4), *max* (Eq 3). This is done by defining a variety of soft minimum functions, and then by showing how they generalize other rules. Each function is dependent on a softness parameter β. For notational convenience, we also define single letter versions of the function using a superscripted V.

We define the *smin* function as follows:

$$V_k^{(s)} \triangleq \mathit{smin}_\beta(z_k^1, z_k^2, \ldots, z_k^L) \triangleq \left(\sum_{l=1}^{L} (z_k^l)^{-\beta} \right)^{-1/\beta}$$

A second rule useful when the arguments are probabilities ($0 < z_k < 1 \ \forall k$) is defined as follows:

$$\begin{aligned} V_k^{(p)} &\triangleq \mathit{psmin}_\beta(z_k^1, z_k^2, \ldots, z_k^L) \\ &\triangleq \exp\left(- \left(\sum_{l=1}^{L} (\ln 1/z_k^l)^\beta \right)^{1/\beta} \right) \end{aligned}$$

We define a third $\min$ function as follows:

$$V_k^{(e)} \triangleq \mathit{esmin}_\beta(z_k^1, z_k^2, \ldots, z_k^L) = \sum_{l=1}^{L} \frac{z_k^l e^{-\beta z_k^l}}{\sum_{\ell=1}^{L} e^{-\beta z_k^\ell}}$$

as well as a corresponding version when the input lies within

[2] Often referred to as the "softmax" function, but we call it the *multiple logistic function* in this paper to avoid any confusion with our soft minimum and maximum functions defined in Section 4.

the range $0 < z_k < 1$:

$$V_k^{(q)} \triangleq \mathit{qsmin}_\beta(z_k^1, z_k^2, \ldots, z_k^L)$$
$$= \exp\left(\sum_{l=1}^{L} \frac{\ln(z_k^l)(1/z_k^l)^\beta}{\sum_{\ell=1}^{L}(1/z_k^\ell)^\beta}\right)$$

For each of the soft minimum functions, there exists a dual "soft maximum" function[3] obtained by negating the value of β. For example, we may define a function $\mathit{smax}_\beta(\cdot) \triangleq \mathit{smin}_{(-\beta)}(\cdot)$. Note that all of the minimum functions approach the true $\min$ function as β gets large since:

$$\min(z_1, z_2, \ldots, z_L) = \lim_{\beta\to\infty} {*}\mathit{min}_\beta(z_1, z_2, \ldots, z_L)$$

These soft functions are useful because they approximate the minimum (resp. maximum) functions as β gets large and positive (resp. as β gets large and negative). These functions are also continuous and differentiable with respect to their arguments. And surprisingly, they generalize most of the functions that are commonly used in classifier combination systems. For example, smin_{-1} is the sum rule, smin_1 is a scaled harmonic mean, psmin_1 is the product rule, and so on. For certain values of β and certain combination methods, some interesting new rules result, such as a "harmonic product" rule using *psmin* with $\beta = -1$. Figure 2 depicts all the generalizations made by the various soft combination rules.

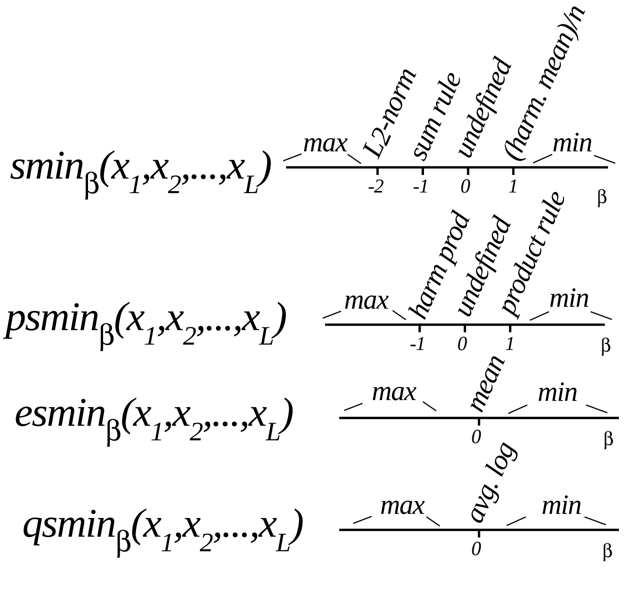

Figure 2: The soft minimum rules generalize many of the more common combination rules (and specify some new ones) depending on the value of β.

5. DERIVATION OF JOINT TRAINING ALGORITHMS

As mentioned above, unlike hard combination rules such as $\min$ and $\max$, the soft versions are continuous and differentiable. Therefore, a new learning algorithm can be defined that jointly trains the L networks. According to the analysis presented in Section 2, a joint training rule should encourage the classifiers to perform as well as possible, but should also encourage any errors, if they must be made, to be as statistically independent as possible.

It might at first seem counterintuitive to jointly train networks in an attempt to produce independent errors. With separate training, however, there is no independence guarantee, instead there is only the hope that the solution arrived at by each classifier will have this property. On the other hand, by using an appropriate joint training rule, the error dependence may be adjusted in a controlled fashion, encouraging the classifiers to arrive at different solutions when it is advantageous to do so. Boosting [28], for example, is a method where the manner in which later classifiers are trained is dependent on the performance of earlier trained classifiers. Other examples of joint training include the mixture of experts architecture [17] (where each sub-classifier essentially manages a subspace of the original feature space) and joint training of product distributions [14].

A joint training algorithm can be defined for any of the soft combination rules mentioned in Section 4. The following analysis will be needed.

First, we need the derivatives of the soft minimum functions. The first one is as follows:[4]

$$\frac{\partial V_k^{(s)}}{\partial z_j^l} = \left(\frac{V_j}{z_j^l}\right)^{1+\beta} \delta_{jk} \qquad (5)$$

Note that when $\beta > 1$, the derivative with respect to the smallest argument of *smin* has the largest value. When β gets large, this gradient approaches unity, while the derivative with respect to larger arguments approaches zero. The exact opposite is true when $\beta < 1$ and gets smaller, i.e., the derivative with respect to the largest element has the largest value. This behavior is as expected, since for the *smin* function, the outputs of the networks other than the minimum do not survive combination when β is large enough. Therefore, they need not change. Only the network with the smallest output should have a non-zero gradient.

The derivatives for the other three soft-min rules are as follows:

$$\frac{\partial V_k^{(e)}}{\partial z_j^l} = \frac{V_j}{z_j^l}\left(\ln z_j^l / \ln V_j\right)^{\beta-1}\delta_{jk},$$

$$\frac{\partial V_k^{(p)}}{\partial z_j^l} = \frac{e^{-\beta z_j^l}}{\sum_\ell e^{-\beta z_j^\ell}}(1 - \beta z_j^l + \beta V_j)\delta_{jk},$$

and

$$\frac{\partial V_k^{(q)}}{\partial z_j^l} = \frac{V_j}{(z_j^l)^{\beta+1}\sum_\ell (z_j^\ell)^{-\beta}}(1 - \beta \ln z_j^l + \beta \ln V_j)\delta_{jk}.$$

These derivatives have interpretations similar to Equation (5), i.e., when β gets large, the derivatives approach

[3] Not to be confused with the standard softmax function used for neural networks which, in this paper, we refer to as the multiple logistic function.

[4] Here, δ_{ij} is the Dirac delta.

unity only when j corresponds to the index for the smallest argument.

For a cost function, we use the relative entropy between the targets t_k and the final combined network outputs z_k (i.e., $J = \sum_k t_k \ln t_k / z_k$). As in normal back-propagation, we compute $\frac{\partial J}{\partial w}$ for each weight w in all of the networks, and perform gradient descent. When the classifiers are MLP-based, the difference between independent training and joint training is that each network has an output-layer "delta" [6] dependent on the other networks (the hidden-layer deltas and the remaining update steps are identical). This can be seen by noting that

$$\frac{\partial J}{\partial z_j^l} = \sum_k \frac{\partial J}{\partial z_k}\frac{\partial z_k}{\partial z_j^l} = \frac{1}{V_j}\frac{\partial V_j}{\partial z_j^l}(z_j - t_j)$$

which leads to the definition of the output delta for the the k^{th} output of the l^{th} network:

$$\begin{aligned}\delta_k^l &= \frac{\partial J}{\partial a_k^l} = \sum_j \frac{\partial z_j^l}{\partial a_k^l}\frac{\partial J}{\partial z_j^l} \\ &= \sum_j (\delta_{jk} - z_k^l)\frac{z_j^l}{V_j}\frac{\partial V_j}{\partial z_j^l}(z_j - t_j)\end{aligned}$$

In this form, the derivative of the appropriate soft minimum rule (or in fact, any rule possessing a derivative) may be substituted in place of $\partial V_j / \partial z_j^l$ in the above to obtain the final output layer deltas. First, define $\alpha_{jk} \triangleq (\delta_{jk} - z_k^l)(z_j - t_j)$. For each of the soft minimum functions, we get the following output layer deltas:

$$\delta_k^{l(s)} = \sum_j \alpha_{jk}\left(\frac{V_j}{z_j^l}\right)^{\beta} \qquad (6)$$

$$\delta_k^{l(p)} = \sum_j \alpha_{jk}\left(\ln z_j^l / \ln V_j\right)^{\beta - 1} \qquad (7)$$

$$\delta_k^{l(e)} = \sum_j \alpha_{jk}\frac{z_j^l e^{-\beta z_j^l}}{\sum_\ell z_j^\ell e^{-\beta z_j^\ell}}(1 - \beta z_j^l + \beta V_j) \qquad (8)$$

$$\delta_k^{l(q)} = \sum_j \alpha_{jk}\frac{1}{(z_j^l)^{\beta}\sum_\ell (z_j^\ell)^{-\beta}}(1 - \beta z_j^l + \beta V_j) \qquad (9)$$

We provide an intuitive explanation of Equation (6) which can be expanded as follows:

$$\delta_k^{l(s)} = \left(\frac{V_k}{z_k^l}\right)^{\beta}(z_k - t_k) - z_k^l \sum_j (z_j - t_j)\left(\frac{V_j}{z_j^l}\right)^{\beta} \qquad (10)$$

This equation says that $\delta_k^{l(s)}$ is the difference of two terms. First, if there is a single network ($L = 1$), one obtains the normal delta rule, $z_k - t_k$. In the general case where $L > 1$, the first term is similar to the normal delta term for single network training (weights are decreased if $z_k > t_k$) except that the difference is weighted according to how close the output z_k^l is to the determining element. The determining element is defined as the minimum (respectively maximum) if $\beta > 1$ (respectively if $\beta < 1$). The second term measures how well the combined classifier system is doing overall on this training pattern, but this score is an average over all output units, each weighted according to the proximity of that unit to the determining element.

Note that with this rule delta, if one network classifier is right and the other is in error, both networks will be encouraged to produce the right solution. However, if $\beta > 0$, then the network that was in error will be allowed to pursue a more relaxed solution (i.e., other outputs will be encouraged to increase) because during the minimum-like combination, those additions will not effect the final result. This delta rule then, in some sense, corrects both networks in response to errors, but allows (and perhaps encourages) them to come up with different solutions to the problem.

The delta rules for the other soft minimum functions can be understood in an analogous manner.

6. EXPERIMENTS AND RESULTS

In this section, we evaluate the above combination schemes as follows. We report results for 1) a baseline system using standard combination rules with embedded training, 2) the new combination rules applied to independently trained networks, 3) jointly trained networks, and 4) jointly embedded trained networks.

We use the OGI Numbers95 telephone-speech continuous numbers corpus [8] with 3233 utterances for training, 357 for cross-validation, and 1206 for testing. The classifiers are initialized using the hand-labelled phone transcriptions part of the Numbers95 distribution. We use an ensemble of $L = 2$ 3-layer MLP-based classifiers for the acoustic modeling component, each trained on different feature representations of the speech signal (RASTA-PLP and MFCC). The dimensionality of the RASTA-PLP feature space is 18, that of the MFCC feature space is 26. We use a window of 9 frames at the input level, which corresponds to approximately 115 ms of speech. The number of input units in each network is 234 (26*9) and 162 (18*9), respectively. In order to compensate for the different dimensionalities of the feature spaces the MFCC network has 400 hidden units, compared to 578 hidden units in the RASTA-PLP network. The number of output units is 32 (which equals the number of HMM states in the system) for both networks.

Baseline experiments

As a baseline system, the MLPs are independently trained using backpropagation, a KL-distance based cost function, and multiple logistic outputs. We trained two bootstrap networks in a single training pass and combined their outputs using the four combination rules (max, min, product and sum) described above. The results are shown in Table 1. Throughout the paper we report results obtained from 10 different test runs for each case, with varying language model weights (1 to 10). This was done to compare the behavior of combination rules across a range of conditions simulating different degrees of acoustic reliability.

WER	MFCC	RASTA	min	max	prod	sum
min	7.0%	8.4%	6.0%	6.7%	5.9%	7.1%
av	7.6%	8.7%	6.6%	7.5%	6.3%	7.1%
max	9.1%	10.3%	7.7%	8.5%	7.3%	8.0%

Table 1: Results for baseline combination experiments; minimum, average and maximum word error rate for 10 different test runs with varying language model factor

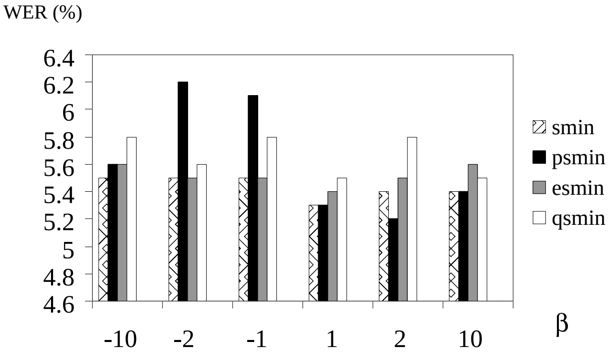

Figure 3: Word error rates for different β values.

Similar to our previous studies [20, 21], we find that the product rule achieves the lowest word error rate. For all embedded training experiments we therefore only use product combination of acoustic scores. After embedded training, the best baseline system achieves a minimum word error rate of 5.1% (5.5% average, 6.2% maximum).

New combination schemes

We evaluated the new combination rules without using embedded training with a variety of β values as shown in Figure 3. Note again that negative β values result in soft maximum rules. We found that changes to β in the range -10 to 10 did not entail appreciable variation in word error rate. We did verify that the *psmin* rule with $\beta = 1$ was similar to the standard product rule, as expected.

Joint training

In our next set of experiments, we jointly trained the acoustic classifiers using the newly developed soft combination schemes, and we combined their outputs after each iteration of embedded training. One difficulty in using the new rules is finding appropriate values for β. Because of low sensitivity to β (Figure 3), we used the same β value (2.0) for all rules in our joint training experiments. We again compared different rules (the standard rules and the respective soft *min* rule) for output score combination after the first training pass. The results are shown in Figure 4. Several points are worth noting. First, certain mismatches between the combination rule used for training and that used for testing sharply increase word error rate. This concerns the *smin* and *psmin* rules in those cases where an "AND" rule (product or one of the *min* rules) is used for training and an "OR" (sum or *max*) rule is used for testing. Combination with the appropriate rule, however, yields a reasonable performance - in almost all cases, a significant reduction in word error rate is achieved compared to the baseline. In the case of training with the *esmin* and *qsmin* rules, by contrast, word error rates are much higher – our conjecture is that a β value of 2 is is too small since these versions are "softer" than *smin* and *psmin*. None of the joint training results as yet surpasses basic product rule combination of individually trained classifiers.

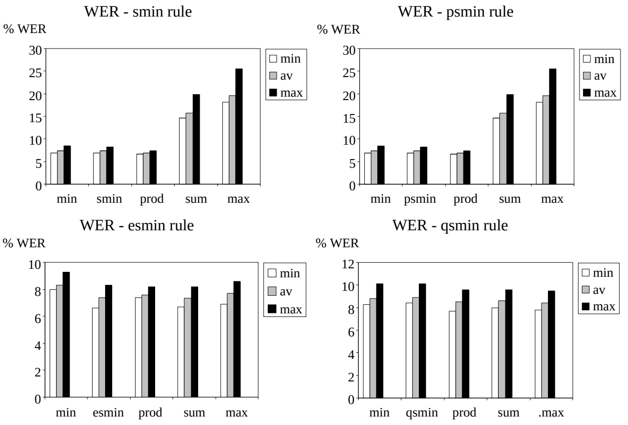

Figure 4: Joint training using new soft minimum functions and tested with various combination strategies.

WER	smin	psmin	esmin	qsmin
min	5.7%	**4.8%**	5.6%	6.7%
av	5.9%	**5.2%**	6.1%	7.2%
max	6.3%	**6.0%**	6.8%	7.8%

Table 2: Word error rates for different min rules after embedded training. Boldface numbers show the best result overall.

For embedded training, only the "AND" combination rules were used. The word error rates obtained by the four different min rules are shown in Table 2. The *psmin* joint training scheme used together with the product combination rule significantly ($p < 0.002$) outperforms the baseline product combination scheme. It should be noted that, at the time of writing, many possible combinations of β values and rules have not been explored in our embedded joint training scheme, so larger gains might still be achievable.

In order to verify our initial assumption that combination improves when the errors of the individual classifiers are uncorrelated, we computed error correlation on the outputs of the individually-trained networks and the jointly-trained networks. The correlation coefficients (Table 3) indeed show that error correlation is much lower for the jointly trained classifiers, as expected.

	baseline	smin	psmin	esmin	qsmin
ρ	0.68	0.57	0.49	0.60	0.52

Table 3: Correlation coefficients for classifier outputs.

7. DISCUSSION

In this paper, we have presented new techniques that generalize previously used combination rules, such as the mean, product, *min*, and *max* functions. These new continuous and differentiable forms be used both for combination of independently trained classifiers and also as objective functions in new joint classifier training schemes. We demonstrated the application of these rules to both combination and joint training using different input features, and we analyze their effects on word recognition accuracy. We found a significant word-error improvement over the product rule using a joint training scheme and product rule combination.

In future work, we plan on more thoroughly investigating the effect of the different rules and β values on combination and joint training performance for a variety of tasks.

Acknowledgements: We would like to thank the International Computer Science Institute for the use of their neural network training software. We would also like to thank Gang Ji for useful discussions regarding this work.

REFERENCES

[1] N.E. Sharkey A.J.C. Sharkey and G.O Chandroth. Neural nets and diversity. In *Proceedings of the 14th International Conference on Computer Safety, Reliability and Security*, pages 375–389, 1995.

[2] J. Bilmes. *Natural Statistic Models for Automatic Speech Recognition*. PhD thesis, U.C. Berkeley, Dept. of EECS, CS Division, 1999.

[3] J. Bilmes, N. Morgan, S.-L. Wu, and H. Bourlard. Stochastic perceptual speech models with durational dependence. *Intl. Conference on Spoken Language Processing*, November 1996.

[4] J.A. Bilmes. Dynamic Bayesian Multinets. In *Proceedings of the 16th conf. on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, 2000.

[5] J.A. Bilmes and K. Kirchhoff. Directed graphical models of classifier combination: Application to phone recognition. In *Proc. Int. Conf. on Spoken Language Processing*, Beijing, China, 2000.

[6] C. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.

[7] Herve Bourlard and Stephane Dupont. A new ASR approach based on independent processing and recombination of partial frequency bands. In *Proceedings ICSLP*, volume I, pages 426–427, 1996.

[8] R.A. Cole, M. Noel, T. Lander, and T. Durham. New telephone speech corpora at CLSU. *Eurospeech 95*, pages 821–824, 1995.

[9] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Wiley, 1991.

[10] S. Dupont and J. Luettin. Using the multi-stream approach for continuous audio-visual speech recognition: experiments on the M2VTS database. In *Proceedings ICSLP-98*, pages 1283–1286, 1998.

[11] J.G. Fiscus. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER). *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding*, 1997.

[12] A.K. Halberstadt and J.R. Glass. Heterogeneous measurements and multiple classifiers for speech recognition. *Proceedings ICSLP-98*, pages 995–998, 1998.

[13] S. Hashem. Optimal linear combinations of neural networks. *Neural Networks*, 10(4):599–614, 1997.

[14] G. E. Hinton. Training products of experts by minimizing contrastive divergence. Technical report, Gatsby Computational Neuroscience Unit, 2000.

[15] Tin Kam Ho, Jonathan J. Hull, and Sargur N. Srihari. Decision combination in multiple classifier systems. *PAMI*, 16(1):66–75, January 1994.

[16] R.A. Jacobs. Methods for combining experts' probability assessments. *Neural Computation*, 7:867–888, 1995.

[17] R.A. Jacobs, M.I. Jordan, S.J. Nowland, and G.E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87, 1994.

[18] L. Jiang and X. Huang. Unified decoding and feature representation for improved speech recognition. *Proceedings of Eurospeech-99*, 1999.

[19] B.E.D. Kingsbury and N. Morgan. Recognizing reverberant speech with RASTA-PLP. *Proceedings ICASSP-97*, 1997.

[20] K. Kirchhoff. Combining acoustic and articulatory information for speech recognition in noisy and reverberant environments. In *Proceedings of the International Conference on Spoken Language Processing*, 1998.

[21] K. Kirchhoff and J. Bilmes. Dynamic classifier combination in hybrid speech recognition systems using utterance-level confidence values. *Proceedings ICASSP-99*, pages 693–696, 1999.

[22] J. Kittler, M. Hataf, R.P.W. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[23] A. Krogh and J. Vedelsby. Neural network ensembles, cross validation, and active learning. In *Advances in Neural Information Processing Systems 7*. MIT Press, 1995.

[24] L. Lam and C.Y. Suen. Optimal combinations of pattern classifiers. *Pattern Recognition Letters*, 16:945–954, 1995.

[25] P. McMahon, P. Court, and S Vaseghi. Discriminative weighting of multi-resolution sub-band cepstral features for speech recognition. *Proceedings ICSLP-98*, pages 1055–1058, 1998.

[26] N. Mirghafori and N. Morgan. Combining connectionist multi-band and full-band probability streams for speech recognition of natural numbers. In *Proceedings of the International Conference on Spoken Language Processing*, pages 743–746, 1998.

[27] G. Potamianos and H.P. Graf. Discriminative training of HMM stream exponents for speech recognition. *Proceedings ICASSP-98*, pages 3733–3736, 1998.

[28] J. R. Quinlan. Bagging, boosting, and C4.5. In *Proc. 14th National Conf. on AI*, 1996.

[29] G. Rogova. Combining the results of several neural network classifiers. *Neural Networks*, 7:777–781, 1994.

[30] A.J.C. Sharkey. Multi-net systems. In *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems*, pages 3–30. Springer, 1999.

[31] K. Tumer and J. Ghosh. Analysis of decision boundaries in linearly combined neural classifier. *Pattern Recognition*, 29:341–348, 1996.

[32] Su-Lin Wu, Michael L. Shire, Steven Greenberg, and Nelson Morgan. Integrating syllable boundary information into speech recognition. In *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, volume 1, Munich, Germany, April 1997. IEEE.