Explicitly Generating Complementary Systems for Large Vocabulary Continuous Speech Recognition

C. Breslin and M.J.F. Gales

Cambridge University Engineering Department
Trumpington Street, Cambridge, CB2 1PZ, UK
{cb404,mjfg}@eng.cam.ac.uk

Abstract

Large Vocabulary Continuous Speech Recognition (LVCSR) systems often use a multi-pass recognition framework where the final output is obtained from a combination of multiple models. Previous systems within this framework have normally built a number of independently trained models, before performing multiple experiments to determine the optimal combination. For two models to give improvements upon combination, it is clear that they must be *complementary*, i.e. they must make different errors. While independently trained models often do give improvements when they are combined, it is not guaranteed that they will be complementary. This paper presents a new algorithm, Minimum Bayes Risk Leveraging (MBRL), for explicitly generating systems that are complementary to each other. This algorithm is based on Minimum Bayes Risk training, but within a boosting-like iterative framework. Experimental results are reported on a Broadcast News Mandarin task. These experiments show small but consistent gains when combining complementary systems using confusion network combination.

1. Introduction

Large Vocabulary Continuous Speech Recognition systems, such as those developed at Cambridge University for Broadcast News transcription [1, 2], typically use a multi-pass recognition framework, where a number of independently trained models are combined in the final stage. The system combination is performed using schemes such as ROVER [3] and CNC [4]. Howver, the models that are combined are normally not guaranteed to be complementary, and often the gains achieved from combination are small. Complementary system generation has been well documented in the context of machine learning [5] and there are many existing algorithms for generating complementary systems. Due to the increased complexity of the task however, most need some modifications before they are applicable to ASR. Complementary system generation for ASR has received growing attention in recent years. The most common approach for creating diverse speech recognition systems is simply to use a number of different acoustic modelling techniques to build several independent models. The models might use different frontends, segmentations, or phone sets. Independently trained systems often give gains when combined together, but there is no guarantee that this will be the case. The major drawback of this approach is that it's not possible to predict which systems have complementary errors without actually performing the combination. Hence, a number of experiments, such as those in [2, 6], must be performed to select the optimal combination. This is time-consuming, and becomes increasingly impractical as the training set size and number of alternative systems increases.

An alternative way of creating diversity is to introduce randomness at some point in the training algorithm. In [7] randomness was added into the state clustering decision tree algorithm. Again, it is not guaranteed that the resulting systems will in fact be complementary, and only small gains in performance were obtained from combining multiple random systems.

Boosting is a machine learning technique that is specifically designed for generating a series of complementary systems; AdaBoost [8] is the most popular boosting algorithm. It maintains a distribution over the training set, giving increased weight to poorly modelled training examples. Training is performed with respect to this distribution, and so that as it progresses, the distribution evolves so later classifiers focus on the 'difficult' examples. The resulting classifiers are then combined together with a weighted voting scheme, with weights predicted by the boosting algorithm. AdaBoost is only suitable for classification tasks involving a finite number of classes. For continuous speech recognition there can be an infinite number of classes, and so several approximations are needed before boosting is suitable for ASR. Boosting-like (or leveraging) algorithms for continuous speech recognition have previously been applied at the frame [9] or utterance level [10].

For training an ensemble of systems for ASR, it would be preferable to use a training algorithm that is explicitly tuned to the final combination scheme; this is the approach adopted in this paper. The combination scheme used is CNC [4], and the approach described, Minimum Bayes Risk Leveraging (MBRL), is based on modifying the Bayes loss function to reflect the errors in combination. This algorithm is described in detail in the next section, followed by preliminary results on a Broadcast News Mandarin system, before conclusions are drawn.

2. Minimum Bayes Risk Leveraging

Minimum Bayes Risk Leveraging (MBRL) is an approach to training complementary systems based on Minimum Bayes Risk training, but with a modified loss function to reflect the fact that multiple systems will be combined together. The standard expression for Minimum Bayes Risk (MBR) training [11] is

$$\mathcal{F}(\mathcal{M}) = \sum_{r=1}^{R} \sum_{H_w \in \mathcal{H}} P(H_w|\mathcal{O}_r;\mathcal{M})\mathcal{L}(H_w,\tilde{H}) \qquad (1)$$

C. Breslin is funded jointly by the EPSRC and Toshiba Research Ltd. This work made use of equipment kindly supplied by IBM under a SUR award

where $\tilde{H}$ is the correct hypothesis for data $\mathcal{O}_r$, $\mathcal{H}$ is the set of all possible hypotheses and $\mathcal{M}$ is the current model. This objective function is a generalisation of many existing discriminative criteria, such as Minimum Phone Error (MPE) and Maximum Mutual Information (MMI) [12]. In common with many discriminative criteria, there is no simple closed-form update approach to minimising this expression, so a range of approaximations have been developed; see for example [11, 12].

MBRL uses the same general form of objective function, but also considers a number of *previous* classifiers when computing the expected loss. There are two ways to introduce the dependency on previous models, $\mathcal{M}^{(0)}...\mathcal{M}^{(s-1)}$, into the objective function. One option is to use the posterior probability of a word dependent on all previous systems. This yields

$$\mathcal{F}(\mathcal{M}) = \sum_{r=1}^{R} \sum_{H_w \in \mathcal{H}} P(H_w|\mathcal{O}_r; \mathcal{M}^{(1)}...\mathcal{M}^{(s-1)}, \mathcal{M})\mathcal{L}(H_w, \tilde{H}) \tag{2}$$

Unfortunately this form of combination is computationally expensive. Alternatively, the dependency can be introduced via a modified loss function, which gives

$$\mathcal{F}(\mathcal{M}) = \sum_{r=1}^{R} \sum_{H_w \in \mathcal{H}} P(H_w|\mathcal{O}_r; \mathcal{M})\tilde{\mathcal{L}}(H_w, \tilde{H}|\mathcal{M}^{(1)}...\mathcal{M}^{(s-1)}) \tag{3}$$

Having determined the form of the objective function, it is necessary to evaluate precisely how the previous models, $\mathcal{M}^{(0)}, \ldots, \mathcal{M}^{(S-1)}$, should alter the value of the modified loss function. The loss function reflects whether the training data is well modelled or not by the previous systems. If earlier systems correctly classified a word, then it should be allocated minimal loss. As CNC [4] is used for system combination, it should also be used to determine the effect of the previous systems on this loss. Thus, the loss function is calculated at a word level; this is in keeping with the original motivation for MBR training, which was to have training criterion which reflects word error rate as the assessment criterion. To calculate the loss function, confusion networks are generated from the training data using the existing models. These confusion networks are then combined together using CNC (the individual system word posteriors are simply averaged in CNC), and aligned with the reference transcription. This yields the posterior probability for each of the hypothesis words given all the existing models.

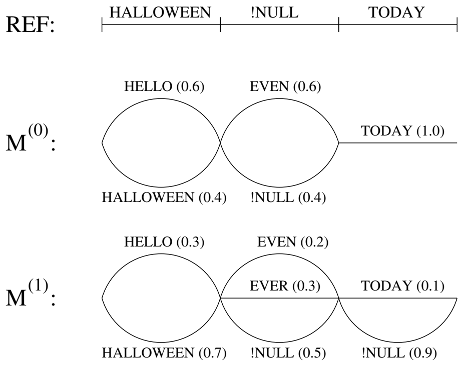

Figure 1: *Complementary Systems Example.*

Figure 1 shows an example of confusion networks from two complementary systems aligned with a reference transcription. Each word is marked with a posterior probability. Consider training $\mathcal{M}^{(1)}$ so that it is complementary to $\mathcal{M}^{(0)}$. With respect to $\mathcal{M}^{(0)}$, the word *TODAY* is well modelled, while the words *HALLOWEEN*, *HELLO* and *EVEN* are poorly modelled. By assigning a high value of loss to the incorrectly modelled words, and a low value to the correctly modelled words, $\mathcal{M}^{(1)}$ can focus on the mistakes made by $\mathcal{M}^{(0)}$.

Two simple methods of using this posterior, $P(W_m|\boldsymbol{O}; S)$, to alter the loss function may then be used. First, the posterior for each word may be used directly in the loss function calculation. Thus the modified loss function for building the S^{th} model, $\tilde{\mathcal{L}}(W_m, \tilde{W}; S)$, becomes

$$\tilde{\mathcal{L}}(W_m, \tilde{W}; S) = \begin{cases} P(W_m|\boldsymbol{O}; S) & W_m \neq \tilde{W} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

The problem with this approach is that the posteriors produced by ASR systems are not normally reliable. To reduce the effects of this, a simple threshold approach may be used. Thus

$$\tilde{\mathcal{L}}(W_m, \tilde{W}; S) = \begin{cases} 1 & P(W_m|\boldsymbol{O}; S) > \alpha, W_m \neq \tilde{W} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

This threshold-based approach may be viewed as form of training data pruning; words that are well modelled by earlier systems are not used to train latter systems. This form of thresholding has some similarities to active training [13], but with the view to building multiple systems, instead of one single best system. The threshold form of modified loss function is used in this paper.

Initialise:
From an initial model, $\mathcal{M}^{(0)}$ generate a set of training
data confusion networks

For: s= 1:S
Combine confusion networks from $\mathcal{M}^{(0)}...\mathcal{M}^{(s-1)}$ with
the reference transcription
Train a model $\mathcal{M}$ to give $\mathcal{M}^{(s)}$ by minimising
a cost function based on:

$$\mathcal{F}(\mathcal{M}^{(s)}) = \sum_{r=1}^{R} \sum_{H_w \in \mathcal{H}} P(H_w|\mathcal{O}_r; \mathcal{M}^{(s)})\tilde{\mathcal{L}}(H_w, \tilde{H}|\mathcal{M}^{(0)}...\mathcal{M}^{(s)})$$

The model $\mathcal{M}$ may differ from previous models by having,
for example, a different frontend, covariance model,
decision tree, topology etc.
Generate training data confusion networks for the
new system $\mathcal{M}^{(s)}$

Output:
The final hypothesis is based on CNC using models
$\mathcal{M}^{(0)}...\mathcal{M}^{(S)}$

Figure 2: Minimum Bayes Risk Leveraging Algorithm

The Minimum Bayes Risk Leveraging algorithm is given in figure 2. In comparison to standard discriminative training which trains a single best system, the aim of this algorithm is to train a number of systems which may perform poorly individually, but which perform well in combination. Similarities can also be drawn between this algorithm and boosting; both algorithms aim to train an ensemble of classifiers which perform well when combined. The loss function in MBRL has the same purpose as the distribution over the training data in boosting. In contrast to boosting however, the form of system combination is left open and no classifier weights are calculated as part of the algorithm. Also, there is no need to alter the training algorithm or resample the training set to take account of the weighting on the training data; this is done implicitly by the MBRL objective function.

Previous work on weighting training data at a smaller granularity than the utterance level (e.g. [9]) has relied on force-aligning the data in order to determine which *frames* correspond to a particular word or phone. Force-aligning is not guaranteed to be accurate, and can lead to errors at the word or phone boundaries. This problem is avoided here by mapping word losses to *states* rather than to frames. Furthermore, [9] uses a confidence measure based on word posteriors to determine a weighting over the training data. Word posteriors are not always well correlated with correctness. The alternative proposed here, aligning confusion networks with the reference transcription, is fast and it provides an easy way to determine word correctness.

Under certain conditions, maximum likelihood (ML) training is optimal. ML-MBRL is a maximum likelihood form of MBRL. It optimises the standard ML objective function, but the state occupation counts are weighted by a loss function. As the state occupation counts are weighted, and the loss function calculated at the word level, the effect on the update formulae is minimal. For example, if state θ belongs to reference word $\tilde{W}$, the modified mean update is given by

$$\boldsymbol{\mu}_\theta = \frac{\sum_{t=1}^{T} l(W_m, \tilde{W}|\mathcal{M}^{(0)}...\mathcal{M}^{(s)})\gamma_\theta(t)\mathbf{o}_t}{\sum_{t=1}^{T} l(W_m, \tilde{W}|\mathcal{M}^{(0)}...\mathcal{M}^{(s)})\gamma_\theta(t)} \qquad (6)$$

where $\gamma_\theta(t)$ is the occupation count for state θ at time t. The variance and prior updates are affected in a similar manner. Again, there are many forms for this loss function, but the threshold function is used for this work.

3. Experimental Results

Experiments were performed on a Broadcast News Mandarin task. The baseline systems were trained using 148 hours of data; 28 hours of Hub-4 data released by the Linguistic Data Consortium (LDC) with accurate transcriptions, and 140 hours of TDT4 data with only closed-caption references provided. Light supervision techniques were used on the latter portion. The feature vector consists of 13 PLP features with 1st, 2nd and 3rd derivatives appended. An HLDA transform is used to map this vector to 39 dimensions, and then pitch and its derivatives are added. Thus, the final feature vector has 42 dimensions. Results are given on two test sets: `dev04f` consists of 0.5 hours of CCTV data from shows broadcast in November 2003, and `eval04` includes 1 hour of data from CCTV, RFA and NTDTV broadcast in April 2004. This system is fully described in [1]. In contrast to [1], these experiments use an ML trained baseline with no speaker adaptation.

Two baseline systems, `G0` and `H0`, were built. `H0` was built using the standard HLDA frontend described above, while `G0` used a Gaussianised frontend [1]. Both systems have on average 16 components per state, and approximately 6000 unique states, after decision-tree based state clustering. From these starting systems, standard ML training using all of the training data was performed to give two further systems, `G1` and `H1`. Also, two complementary systems, `G1c` and `H1c` were built using ML-MBRL. Both of these systems were built to be complementary to `G0` using a threshold of $\alpha = \exp(-1)$ in the loss function (equation 5). This corresponded to 32% of the training data words. In addition, the state boundaries were fixed during training as initial experiments showed that these could drift significantly when using ML-MBRL training.

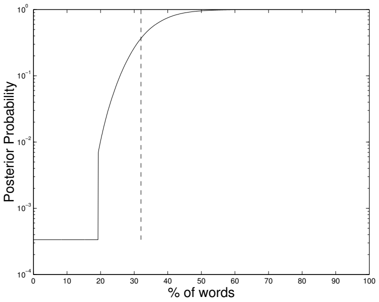

Figure 3: *Training set word posteriors,* `G0`

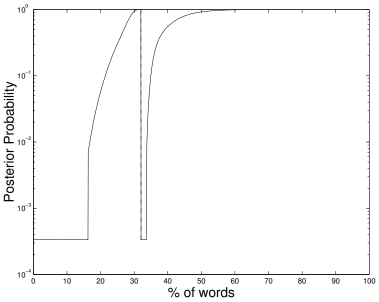

Figure 4: *Training set word posteriors,* `G1c`

Figure 3 shows the distribution of training set word posteriors obtained using `G0`; the threshold of $\exp(-1)$, corresponding to 32% of words is also marked. Approximately 20% of words have zero posterior probability with respect to `G0`. Figure 4 shows how these word posteriors change after carrying out ML-MBRL training. This graph shows the distribution of word posteriors obtained using `G1c`, for both portions of training data (i.e. the 32% used for training and the remaining 68%). It can be seen that the algorithm has increased the posterior probability of many previously badly recognised words, but has also had the effect of decreasing the posteriors for previously well recognised words.

Table 1 shows the confusion network decoding results for the individual baseline and complementary models. Performing further iterations of ML training has very little effect on the error rate; for example `G0` and `G1` both have an error rate of 14.3% on the `dev04f` set. However, performing ML-MBRL training degrades the individual system results; the error rate for `G1c` is 14.7%. This effect is seen for both complementary models, on both test sets.

The results from confusion network combination are also given in table 1. Combining two independent models does give improvements in error rate, as has been seen in previous work. For example, `G0` and `H1` have individual error rates of 14.3% and 14.4% on `dev04f`, and their combination decreases the error rate to 13.8%. However, combining two complementary models can give greater improvements. For example, `G0` and `H1c` have indi-

Model	System	CER (%)	
		dev04f	eval04
G0	GAUSS	14.3	22.9
H0	HLDA	14.4	23.2
G1	GAUSS	14.3	22.8
G1c		14.7	23.0
H1	HLDA	14.4	23.2
H1c		14.6	23.2
G0 + G1	CNC	14.1	22.6
G0 + G1c		14.0	22.4
G0 + H1	CNC	13.8	22.3
G0 + H1c		13.4	22.0

Table 1: Individual System and CNC Results (CER %)

Model	dev04f		eval04	
	CNC	IDEAL	CNC	IDEAL
G0 + G1	14.1	14.0	22.6	22.5
G0 + G1c	14.0	13.3	22.4	21.9

Table 2: IDEAL and CNC combination Results (CER %)

vidual error rates of 14.3% and 14.7%, but their combination gives an error rate of 13.4%. This is a gain of 0.4% absolute over the independent system combination, despite the fact that the individual error rate for H1c is 0.3% worse than for H1. As expected due to their similarity, the gains got from combining two GAUSS systems is small, while larger gains are seen from combining an HLDA and a GAUSS system.

It is interesting to consider the potential gains that can be achieved using system combination, and hence an *ideal* combination scheme was also implemented. This scheme first aligns the reference transcription with the confusion networks in order to determine whether either, or both, of the systems are correct. If the first system selects the correct word, then the second system is ignored and only the first system is used. If the first system is incorrect, then the combination with the second system is performed. This form of combination mirrors the threshold scheme used in training. Table 2 compares the results of this ideal scheme with standard CNC for the combination of G0, G1 and G1c. For the combination of G0 and G1, standard confusion network combination gives error rates of 14.1% and 22.6% on dev04f and eval04 respectively, compared to the ideal scheme results of 14.0% and 22.5%. That is, confusion network combination is an almost optimal combination scheme for combining these systems. In contrast, for combination with the complementary system, the results from CNC are significantly worse than for the ideal scheme. CNC results using G0 and G1c are 14.0% and 22.4% on the two test sets, but the ideal combination results are 13.3% and 21.9%. Hence, a more sophisticated combination scheme might be more suitable for combining these complementary systems.

4. Conclusions

This paper has presented a new algorithm, Minimum Bayes Risk Leveraging, for explicitly building systems that are complementary to each other. This algorithm differs from previous boosting-like algorithms as it is based on Minimum Bayes Risk training, and relies on confusion network combination in training to accurately determine a weighting on the training data. It also differs from discriminative training and active learning in its aim to build an ensemble of classifiers, rather than one single best model.

It was found that building systems to be complementary to another degraded the error rate of the individual systems when compared to standard training. However, combining complementary systems led to improvements over combining independently trained systems. This is in contrast to previous work with CNC, which has found that optimal combination is obtained when the systems being combined have similar error rates. However, the results of an *ideal* combination scheme indicate that standard CNC is not an optimal method of combination for complementary systems, and that an alternative form of combination is needed to fully take advantage of the complementary nature of these models.

5. References

[1] Sinha, R., Gales, M.J.F., Kim, D.Y., Liu, X.A., Sim, K.C. and Woodland, P.C. “The CU-HTK Mandarin Broadcast News Transcription System”, ICASSP 2006.

[2] Gales, M.J.F., Kim, D.Y., Woodland, P.C., Chan, H.Y. Mrva, D., Sinha, R. and Tranter, S.E., “Progress in the CU-HTK Broadcast News Transcription System”, IEEE Trans. Speech and Audio Processing, to appear.

[3] Fiscus, J.G. “A Post-processing System to Yield Reduced Word Error Rates: Recogniser Output Voting Error Reduction (ROVER)”, Proc. IEEE ASRU Workshop, pages 347-352, 1997.

[4] Evermann, G. and Woodland, P.C. “Posterior Probability Decoding, Confidence Estimation and System Combination”, Proc. of the NIST Speech Transcription Workshop, 2000.

[5] Dietterich, T.G. “Ensemble Methods in Machine Learning”, Lecture Notes in Computer Science, vol 1857, pp 1-15, 2000.

[6] Liu, X., Gales, M.J.F, Sim, K.C. and Yu, K. “Investigation of Acoustic Modeling Techniques for LVCSR Systems”, ICASSP 2005.

[7] Siohan, R., Ramabhadran, B. and Kingsbury, B. “Constructing Ensembles of ASR Systems using Randomised Decision Trees”, ICASSP 2005.

[8] Freund, Y. and Schapire, R.E. “A Decision-Theoretic Generalisation of Online Learning and an Application to Boosting”, Journal of Computer and System Sciences, 55(1):119-139, 1997.

[9] Zhang, R. and Rudnicky, A.I. “A frame level boosting training scheme for acoustic modelling”, ICSLP, 2004.

[10] Meyer, C. “Utterance Level Boosting of HMM Speech Recognisers”, ICASSP, 2002.

[11] Doumpiotis, V. and Byrne, W. “Lattice Segmentation and Minimum Bayes Risk Discriminative Training for Large Vocabulary Continuous Speech Regognition”, Speech Communication, (2):142-160, 2005.

[12] Povey, D. “Discriminative Training for Large Vocabulary Continuous Speech Recognition”, PhD Thesis, University of Cambridge, 2004.

[13] Arslan, L.M. and Hansen, J.H.L. “Selective Training for Hidden Markov Models with Applications to Speech Classification”, IEEE Trans. Speech and Audio Processing, Vol. 7 (1):46-54, 1999.