Building Multiple Complementary Systems using Directed Decision Trees

C. Breslin and M.J.F. Gales

Cambridge University Engineering Department, UK
{cb404,mjfg}@eng.cam.ac.uk

10.21437/Interspeech.2007-419

Abstract

Large vocabulary speech recognition systems typically use a combination of multiple systems to obtain the final hypothesis. For combination to give gains, the systems being combined must be complementary, i.e. they must make different errors. Often, complementary systems are chosen simply by training multiple systems, performing all combinations, and selecting the best. This approach becomes time consuming as more potential systems are considered, and hence recent work has looked at explicitly building systems to be complementary to each other. This paper considers building multiple complementary systems based on directed decision trees, and combining them within a multi-pass adaptive framework. The tree divergence is introduced for easy comparison of trees without having to build entire systems. Experiments are presented on a Broadcast News Arabic task, and show that gains can be achieved by using more than one complementary system.

Index Terms: automatic speech recognition, system combination, complementary systems

1. Introduction

State-of-the-art large vocabulary speech recognition (LVCSR) systems often use a multi-pass framework for decoding. Initial lattices and 1-best transcriptions are generated and used to perform speaker adaptation. The lattices are next rescored by several different systems and the outputs combined to give the final hypothesis. Methods such as ROVER [1] or Confusion Network Combination (CNC) [2] are used to perform this combination. Improvements can only be seen if the models being combined are complementary. That is, if they make different errors.

The most straightforward way to find systems that are complementary is to independently train a variety of systems using, for example, different frontends or covariance modelling, and evaluate all possible combinations. This approach becomes time-consuming as the number of potential systems increases, and it is not guaranteed that any will in fact be complementary. Previous work has found that the best results are obtained when combining independently trained systems that have comparable error rates [3]. Having similar error rates is not a necessary requirement for systems to be complementary, and this observation suggests that the types of independently trained system that can be successfully combined in practice is limited.

Ensembles of classifiers have proven to be successful in the machine learning community [4], and recent work has looked at explicitly training complementary systems for automatic speech recognition. However, many machine learning algorithms require alterations before they can be applied to ASR. Boosting is one algorithm that has successfully been adapted to speech recognition [5]. Boosting introduces a weighting on the training data so that in successive iterations, more emphasis is placed on harder to recognise training data. Minimum Bayes Risk Leveraging [6] takes a similar approach to weighting training data to reflect confusions.

As an alternative to weighting the training data, [7] proposes building directed decision trees that bias the trees towards becoming complementary. Another approach to generating complementary decision trees is to add randomness into the decision tree generation [8]. This paper builds on previous directed decision tree work by building multiple complementary systems, decoding within a multi-pass adaptive framework, and introducing a divergence measure to judge how close decision trees are to each other.

The following sections describe decision trees, a framework for building multiple complementary systems, and a measure for tree divergence. Experimental results are then presented and discussed on a Broadcast News Arabic task.

2. Decision Trees

Decision trees are binary trees used to cluster states of triphone HMMs, so that parameters can be shared [9]. They contain questions, typically concerning triphone context, at their nodes, and states clustered at their leaves. A top-down clustering algorithm is used to build the tree, and there are three stages involved in decision tree generation. First, statistics are gathered for each state of each triphone in the training data. Second, the tree is built by repeatedly splitting the states into smaller clusters. For each split, all possible questions are ordered according to the likelihood of the data after the split, and the question selected to split the states is the one which maximises the likelihood. Thirdly, the tree stops growing when the data likelihood falls below a threshold.

Decision trees are widely used for parameter tying because they provide an elegant way to cluster triphones that aren't seen in the training data and the size of the tree can easily be controlled by altering the stopping criterion threshold. However, states that are clustered together share the same output distribution, and so higher level information such as the language model or context is needed to tell them apart.

Splitting the data in decision tree generation is a locally optimal operation, and small changes to the question selection can lead to very different decision trees. Random decision trees [8] take advantage of this fact to build different decision trees by altering the question selection stage so a random question from the top N is chosen, rather than simply picking the best.

3. Directed Decision Trees

Directed decision trees [7] aim to build complementary systems by biasing the decision tree generation towards separating states which are confusable. As well as obtaining the standard statistics for decision tree generation, a second set of weighted statistics is also obtained. These statistics are weighted so as to give

a higher weight to poorly recognised parts of the training data. The question selection stage is then altered so that best question is chosen with respect to these weighted statistics rather than the original statistics. The original statistics are used for the stopping criterion so that the directed trees are a similar size to the original trees.

In this paper, the following method is used for weighting the training data. First, Confusion Networks are obtained for every utterance in the training data, and are aligned with the reference transcription. Words in confusion networks are annotated with posterior probabilities, so it is trivial to obtain a posterior probability for each word in the reference. Then, a loss for each training data word $\tilde{W}$ given a model $\mathcal{M}$ is calculated, $l(\tilde{W}|\mathcal{M})$, so that loss is inversely correlated with the word posterior, $P(\tilde{W}|\boldsymbol{O};\mathcal{M})$:

$$l(\tilde{W}|\mathcal{M}) = \left(1 - P(\tilde{W}|\boldsymbol{O};\mathcal{M})\right)^{\alpha}$$

Now, when calculating the statistics, state level occupation counts ($\gamma_j(t)$) are multiplied by the loss ($l_j(t)$) of the word that state belongs to

$$\sum_t \gamma_j(t)\boldsymbol{o}_t \rightarrow \sum_t l_j(t)\gamma_j(t)\boldsymbol{o}_t$$

Hence states which are often confused in the training data will have higher weights than those which are well recognised.

Previous work using directed decision trees has concentrated on building just one complementary system. It is easy to extend the idea to building multiple complementary systems using an iterative framework such as that used in boosting. First, it is necessary to consider how the loss function calculation is changed when training a system to be complementary to a number of previous models $\mathcal{M}^1 \cdots \mathcal{M}^{(S)}$. Training data confusion networks are generated for all previous systems, and aligned with the reference, as in figure 1. Then, it is straightforward to obtain an *average word posterior* for each reference word. This average posterior is used in the loss function, which becomes

$$l(\tilde{W}|\mathcal{M}^{(1)} \cdots \mathcal{M}^{(S)}) = \left(1 - P_{avg}(\tilde{W}|\boldsymbol{O};\mathcal{M}^{(1)} \cdots \mathcal{M}^{(S)})\right)^{\alpha}$$

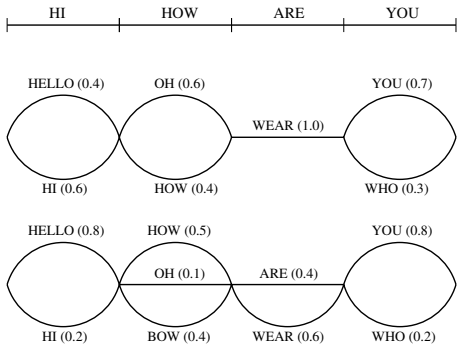


Figure 1: *Calculating the loss function with multiple previous systems*

A system, $\mathcal{M}^{(S+1)}$, trained using this modified loss function will be complementary to models $\mathcal{M}^{(1)} \cdots \mathcal{M}^{(S)}$, rather than to just one other system. Next, this can be used in an iterative framework to build multiple complementary systems. Figure 2 shows this framework. First, one system is trained to be complementary to a baseline. Then, a second system is trained complementary to both the baseline and the first complementary system. This can be repeated to obtain a number of systems, all complementary to each other. The advantage of this iterative framework is that the order of combination is now given as the order the systems were built in.

This is a general framework for training complementary systems, and can be used with any complementary training algorithm. Although Confusion Networks are used as the combination method here, any other combination algorithm can be used, and the combination method used in decoding can be matched with that used in training.

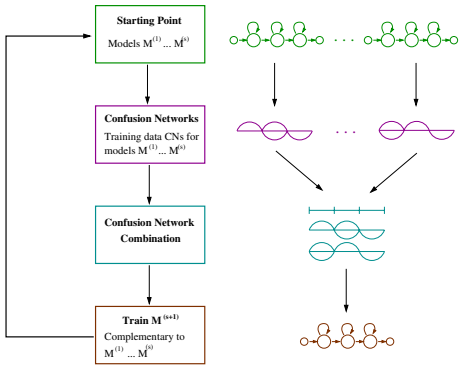


Figure 2: *Framework for generating multiple complementary systems*

4. Decision Tree Divergence

The decision tree generation stage typically occurs early in the process of building a speech recognition system. For the experiments presented in section 5, decision tree clustering is performed on a triphone system which has one Gaussian component per state output distribution. Then, ML training is performed while increasing the number of components to 16, an HLDA transform is computed and MPE training performed. Only after the system has been trained is it possible to evaluate its performance. Hence, some measure for comparing decision trees earlier in the process is useful when building systems based on multiple decision trees.

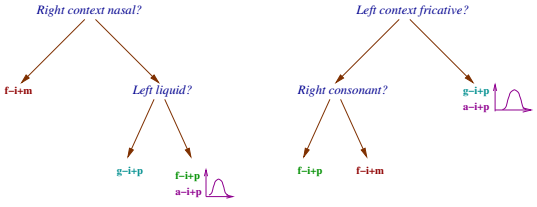


Figure 3: *Comparing Decision Trees*

Figure 3 shows an example of two decision trees with different clusterings. It is possible to compare these clusterings directly using, for example, a cluster similarity measure like that in [10]. However, a cluster comparison measure typically uses many pairwise comparisons between clustered elements, and so proves expensive in practice.

The divergence measure used here relies on the fact that all states clustered at a node share a common Gaussian distribution. Hence, for each state of each triphone, it is possible to

obtain a Gaussian distribution from both trees. If these are close together, then it is likely that the clustering is similar, and if they are further apart the clustering is likely to be different. Hence, a measure of tree divergence can be calculated as in figure 4. The symmetric Kullback-Leibler (KL) divergence is used as a measure of distance between Gaussians from the two trees, and can be calculated for each state of each triphone in the training data. The tree divergence is an average of these KL divergences, weighted by the state occupation count.

```
divergence = 0.0
For each state of each triphone:
    γ = training data state occupation count
    N1 = state output distribution in tree 1
    N2 = state output distribution in tree 2
    d = Symmetric-Kullback-Leibler(N1, N2)
    divergence += γ * d
end
```

Figure 4: *Calculating Decision Tree Divergence*

5. Results

Experiments were performed on a Broadcast News Arabic task. Each system was trained using 101.8 hours of data and a PLP frontend. After decision tree clustering, the systems were mixed up to 16 components per state using ML training. Next, an HLDA transform was computed to map from 52 dimensions (12 PLP coefficients plus energy, 1st, 2nd and 3rd derivatives) to 39 dimensions before the number of components per state was reordered to be proportional to the state occupation count. An average of 16 components per state is maintained. Finally, MPE training is performed using an MMI prior and gender dependent models were built. A baseline system, S0, was first built, and D1 was built to be complementary to S0 by using a directed decision tree. Next, D2 was built, also using a directed decision tree, to be complementary to both S0 and D1. D1 and D2 were built in exactly the same way as S0, the only difference being the decision tree generation.

Results are given on three test sets: `bnat05` (5.72 hours) and `bnat06` (2.76 hours) are broadcast news data, with the former being more closely matched to the training data. `bcat06` (2.81 hours) consists of broadcast conversation shows, and hence is the least closely matched. As short vowels are missed out of Arabic transcriptions, pronunciations are generated using Buckwalter morphological rules, and pronunciation probabilities are used in decoding. A trigram language model with a 65k wordlist was used, and trigram lattices were converted to confusion networks in order to perform confusion network decoding for the individual system results. This allows the gains achieved by combination to be easily seen. Decoding was performed in a multi-pass framework using CMLLR adaptation. This framework is explained in further detail below, and further information about Broadcast News transcription can be found in [3].

The decision tree divergence was measured for the three systems, and figure 5 shows how the tree divergence varies with α in the loss function calculation. D1 was compared to just the baseline, S0, while D2 was compared both to S0 and to D1. The divergence tends to increase with α, as more emphasis is placed on harder to recognise training data. Both D1 and D2 are a similar distance from the S0, but D1 and D2 are much closer together. Thus the tree divergence is used to judge an appropriate value for α without having to build the entire system. For the results presented below, D1 was built with $\alpha = 1$ and D2 was built with $\alpha = 16$.

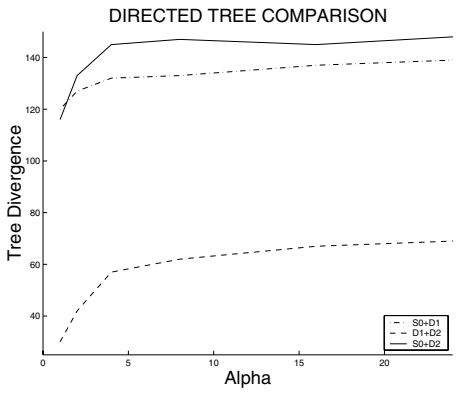


Figure 5: *Decision Tree Divergence with α when comparing S0+D1 (dotted line), S0+D2 (solid), and D1+D2 (dashed)*

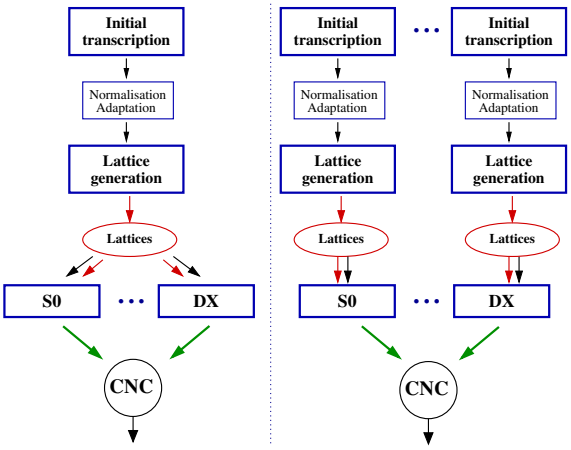


Figure 6: *Multipass framework with (a) common lattice generation, (b) different lattice generation passes*

Experiments were first run in the simple multi-pass framework shown to the left in figure 6. In this framework, an initial transcription was generated using gender independent S0 models, so normalisation and adaptation could be performed, before lattices were generated. These lattices were rescored using S0, D1 and D2 gender dependent models, and their outputs combined using Confusion Network Combination. The results obtained are shown in table 1.

For the most closely matched testset, `bnat05`, the individual performance of the directed tree systems is slightly worse (0.2% absolute) than the baseline, but improvements are seen on the other two sets. On the most mismatched set, `bcat06`, an improvement of 0.8% absolute is seen, from the baseline error rate of 42.7% down to 41.9%.

When the combination of S0 and D1 is performed, gains are seen for all test sets. The smallest gain is again on `bnat05`, with an improvement of 0.3% absolute over the baseline, and the largest gain of 0.9% absolute is seen on `bcat06`. When combining all three systems, further gains are only seen on the

bcat06 set, but a final performance on this set of 41.4% WER is 1.3% absolute better than S0 alone.

	System	bnat05	bnat06	bcat06
BASELINE	S0	20.2	31.4	42.7
DIRECTED	D1	20.4	31.0	42.0
	D2	20.4	30.7	41.9
CNC	S0+D1	**19.9**	**30.5**	41.8
	S0+D1+D2	19.9	30.5	**41.4**

Table 1: *WER (%) results using shared adaptation and lattice generation*

Decoding was also run in the alternative multi-pass framework to the right in figure 6. In this framework, normalisation, adaptation and lattice generation were performed independently for each system using gender independent models, before being rescored using gender dependent models. Table 2 shows the results obtained with this setup.

The individual system results obtained for D1 and D2 are slightly worse than for the previous setup. A gain of 0.2% absolute can be seen for D2 on bnat06, and a gain of 0.6% absolute for bcat06. However, the results obtained from combination are better. A gain of 0.5% absolute is seen on bnat05 when combining S0 with D1, although no gain is seen when adding a third system too. Gains of 1.0% and 1.1% absolute are seen for bnat06 and bcat06 respectively when combining S0 and D1, and further small gains of 0.2% and 0.4% are seen when incorporating D2.

	System	bnat05	bnat06	bcat06
BASELINE	S0	20.2	31.4	42.7
DIRECTED	D1	20.7	31.4	42.4
	D2	20.7	31.2	42.1
CNC	S0+D1	**19.7**	30.4	41.6
	S0+D1+D2	19.8	**30.2**	**41.2**

Table 2: *WER (%) results using independent adaptation and lattice generation*

The results in tables 1 and 2 show clearly the cross adaptation effect that can be achieved when using one system to perform adaptation and generate lattices, and another to rescore them. The effect is demonstrated by the improved individual system performances when using a shared adaptation and lattice generation pass (table 1) over using independent passes (table 2). It is interesting then that these gains don't follow through when system combination is performed. It is the framework with no cross-adaptation effect that gives the best results in combination. This shows that individual system error rate is not necessarily a good indicator of whether two systems are complementary or not, and, as such, simply aiming to optimise individual system performances may not lead to optimal performance in combination. The first framework is more efficient for decoding, and it may be the case that relaxing the pruning on the first passes to give more diverse lattices for rescoring will combine the efficiency benefits of the first framework with the performance gains of the second.

A second observation concerns the relative gains obtained when combining the baseline with either one or two complementary systems. Although the largest gains over the baseline are obtained by combining two complementary systems and the baseline, the contribution from adding the second complementary system is small compared to that of just the first. This could be because the two directed decision trees aren't diverse enough, as seen with the tree divergence measure in figure 5.

A final interesting observation is that the performance gains obtained both for the individual directed tree systems and in combination are directely correlated with training set mismatch.

6. Conclusions

This paper has extended previous work on directed decision trees by building multiple complementary systems inside an iterative framework, and decoding within a multi-pass framework using speaker adaptation. The results show that gains can be achieved using more than one complementary system, but that gains from a second complementary system are smaller than from the first. Tree divergence was introduced as a way of comparing decision trees without having to train speech recognition systems. Future work will look at optimising the multi-pass framework both in terms of performance gains and time efficiency, and alternative ways of introducing diversity into the second complementary system to further improve results.

7. Acknowledgements

C. Breslin is jointly funded by the EPSRC and Toshiba Research Ltd.

8. References

[1] J.G. Fiscus, "A post-processing system to yield reduced word error rates: Recogniser output voting error reduction (ROVER)," in *Proc. IEEE ASRU Workshop, pages 347-352*, 1997.

[2] G. Evermann and P. C. Woodland, "Posterior probability decoding, confidence estimation and system combination," in *Proceedings Speech Transcription Workshop*, College Park, MD, 2000.

[3] M.J.F. Gales, D.Y. Kim, P.C. Woodland, H.Y. Chan, D. Mrva, R. Sinha, and S.E. Tranter, "Progress in the CU-HTK broadcast news transcription system," *IEEE Trans. Speech and Audio Processing*, vol. 14, pp. 1513–1525, 2006.

[4] T.G. Dietterich, "Machine learning research: Four current directions," *The AI Magazine*, 1997.

[5] R. Zhang and A. I. Rudnicky, "A frame level boosting training scheme for acoustic modelling," in *Proceedings ICSLP*, 2004.

[6] C. Breslin and M.J.F. Gales, "Generating complementary systems for speech recognition," in *Proceedings ICSLP*, 2006.

[7] C. Breslin and M.J.F. Gales, "Complementary system generation using directed decision trees," in *Proceedings ICASSP*, 2007.

[8] O. Siohan, B. Ramabhadran, and B. Kingsbury, "Constructing ensembles of ASR systems using randomized decision trees," in *Proceedings ICASSP*, 2005.

[9] J. Odell, *The use of context in large vocabulary speech recognition*, Ph.D. thesis, 1995.

[10] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the Americal Statistical Association*, vol. 66, pp. 846–850, 1971.