Hill-Climbing Feature Selection for Multi-Stream ASR

David Gelbart[1], *Nelson Morgan*[1,2], *Alexey Tsymbal*[3]

[1]International Computer Science Institute, USA
[2]EECS Department, University of California at Berkeley, USA
[3]Siemens AG, Germany
david.gelbart@gmail.com, morgan@icsi.berkeley.edu, alexey.tsymbal@siemens.com

Abstract

We performed automated feature selection for multi-stream (i.e., ensemble) automatic speech recognition, using a hill-climbing (HC) algorithm that changes one feature at a time if the change improves a performance score. For both clean and noisy data sets (using the OGI Numbers corpus), HC usually improved performance on held out data compared to the initial system it started with, even for noise types that were not seen during the HC process. Overall, we found that using Opitz's scoring formula, which blends single-classifier word recognition accuracy and ensemble diversity, worked better than ensemble accuracy as a performance score for guiding HC in cases of extreme mismatch between the SNR of training and test sets.

Our noisy version of the Numbers corpus, our multi-layer-perceptron-based Numbers ASR system, and our HC scripts are available online.

Index Terms: speech recognition, feature selection, ensemble

1. Introduction

The usual architecture of a multi-stream automatic speech recognition (ASR) system is a set of classifiers acting in parallel on the same classification problem, producing parallel streams of classifier output which are then combined. Such an architecture is often referred to as an ensemble of classifiers in the pattern recognition literature. Diversity (disagreement between members of the ensemble) is necessary for an ensemble to perform better than its best individual member. A popular way to create diversity in an ensemble ASR system is to provide a different feature vector to each classifier. This leads to the problem of ensemble feature selection (EFS): given a pool of available features, what features should be used by each classifier?

Published work on EFS for ASR that we are aware of has dealt with indivisible blocks of features, so that a feature vector must contain either all the features in a block or none of them (see section 2.3 of [1] for a survey). In this work, we performed EFS at the level of individual features. For example, the feature vector of each classifier in an ensemble may contain both some MFCC features and some PLP features, while leaving out other features, and a feature may appear in the feature vectors of more than one classifier.

We carried out EFS using a hill-climbing (HC) approach previously used outside the ASR field [2]. In this approach, initially chosen feature vectors are iteratively improved by adding or removing a single feature at a time to or from a feature vector if the change improves a performance score. The process is stopped when no more performance improvement can be achieved by the algorithm.

This paper presents key results from [1][3], while [1] also included experiments using the ISOLET corpus to evaluate the random subspace method [4] and HC.

2. Data sets and ASR approach

2.1. The OGI Numbers corpus

The OGI Numbers corpus [5] consists of strings of spoken numbers collected over telephone connections. We used version 1.3 of the corpus, divided into a 6 hour training set and two 2 hour test sets. We called the first test set the *development set*, since we used it as test data for guiding the HC process, and the second test set the *evaluation set*, since we used it as held-out data for final performance evaluation.

2.2. Our noisy version of the Numbers corpus

We created a noisy version of the corpus by adding one of ten different noise types to each utterance at one of six different signal-to-noise ratios. We will refer to this version of the corpus as *noisy* and to the original corpus as *clean*. Four noise types were used for all three sets (training, development, and evaluation), and each of the other noise types was only used in one set (with two of these noise types per set). Thus, there was a mix of matched noise types (shared across all sets) and mismatched noise types (only found in one set).

The design of our noisy corpus was heavily influenced by the Aurora 2 benchmark [6], with the additional goal of separating development test data used for feedback to improve systems from held out evaluation data used to report final results [7].

We have provided scripts and noise recordings [8] other researchers can use to make their own exact copy of our noisy version of Numbers, starting from a copy of the clean corpus.

2.3. Our use of multi-layer perceptrons (MLPs)

We used a "hybrid connectionist" ASR approach [9], in which MLPs are used for acoustic modeling within an HMM. While Gaussian mixture model based systems are more common than MLP based systems, MLPs have often handled novel feature types particularly well [10]. Our results using MLPs are made more relevant to researchers using GMM-based systems by past work on the "tandem" approach, which can be used as a bridge between them [11][12].

We used one MLP per stream. We combined the outputs of the MLPs by combining posterior probabilities at the frame level. We did this using a common approach: taking the geometric mean of the posterior probabilities for each phone across MLPs [13] (for numerical reasons we calculated this through taking the arithmetic mean of logarithmic probabilities).

2.4. Our Numbers ASR system

Our scripts and configuration files for Numbers ASR using multi-layer perceptrons, based on the open source Quicknet toolkit and noway decoder [14], can be downloaded at [8].

We used part of the training set as cross-validation data for early stopping during the MLP training process and as a test set for tuning decoder parameters using a grid search. We repeated decoder parameter tuning whenever we made a change to feature vectors, MLP topology, or the training condition (i.e., changing between clean and noisy training).

As an experimental control, we always chose the number of MLP hidden units so that the total number of acoustic model parameters in each system was about 806,400 (corresponding to 3600 hidden units for a single-MLP MFCC or PLP system). For a system with more than one MLP, this total was the sum of the number of parameters for all the MLPs, and the number of hidden units for each MLP was chosen so that the number of parameters for each MLP was approximately equal. When feature vector sizes changed during HC, we adjusted hidden layer sizes to satisfy these requirements.

Partway through this project, we noticed that we could improve ASR accuracy by making some adjustments to MLP training and duration modeling [1][8], but for comparability we kept our original settings for all experiments.

3. Hill-climbing procedure

The pseudocode below, titled HILL_CLIMBING, corresponds to the version of HC used in this paper.

In some of our experiments, HC scored candidate feature vectors by ensemble word recognition accuracy. In the other experiments, it scored them by the formula $fitness_s = acc_s + \alpha * div_s$, where s identifies a stream, acc_s is the single-stream (i.e., non-ensemble) accuracy of that stream, and div_s is that stream's contribution to ensemble diversity. (In the pseudocode listing, scoring is represented by the CALCULATE_SCORE function.) The α parameter provides an adjustable trade-off between accuracy and diversity, but for simplicity we always used $\alpha = 1$. This formula was introduced by Opitz [15].

At first glance Opitz's formula may be less intuitive than ensemble accuracy. The authors of [16] used ensemble accuracy to guide ensemble feature selection, and mentioned that overfitting was sometimes a problem; they suggested the inclusion of a diversity term in the scoring formula as a response. While Opitz's formula is presumably still prone to overfitting of individual classifiers, it may have an advantage in that an ensemble of overfitted classifiers is not necessarily an overfitted ensemble. In fact, some work (e.g. [17][18][19]) suggests that overfitting of individual classifiers may actually help ensemble performance in some situations. We set acc_s to the single-stream word recognition accuracy of stream s. We calculated div_s by calculating the pairwise diversity between stream s and each other stream, and then averaging the pairwise diversities. We defined pairwise diversity as the number of word hypotheses that differ divided by the total number of words.

In all HC experiments, our feature pool contained 39 MFCC features, 39 PLP features, and 28 MSG (Modulation-filtered SpectroGram) features. In order to see the effect of varying ensemble size, in some experiments we used three MLPs per ensemble and in some we used five. When using three MLPs we sometimes started off by assigning features to MLP randomly using the random subspace method (RSM) [4]. In the other cases we started off with each of the three MLPs using a particular feature extraction algorithm. That meant one MLP using the MFCCs, another using the PLP features, and a third using the MSG features. When the three initial feature vectors were chosen by RSM we also used 39 features for the first two MLPs and 28 features for the third MLP. When we used five MLPs, the MLPs initially used 13 static MFCCs, 26 dynamic (delta and delta-delta) MFCC features, 13 static PLP features, 26 dynamic PLP features, and 28 MSG features respectively.

```
HILL_CLIMBING($FS, S, N$)
▷ FS: the set of feature vectors (already initialized)
▷ S: the number of feature vectors (streams) in FS
▷ N: the number of features in the feature pool

▷ Perform hill-climbing for each stream in turn
for $s \leftarrow 1$ to $S$
        ▷ Initialize this stream's score.
        $score \leftarrow$ CALCULATE_SCORE($FS, s$)
        repeat
                $improvement \leftarrow false$
                ▷ For each feature in the feature pool
                for $i \leftarrow 1$ to $N$
                        ▷ If feature $i$ is in stream $s$, remove it.
                        ▷ If feature $i$ is not in stream $s$, add it.
                        SWITCH($i, FS[s]$)

                        ▷ If the change improved the score
                        $newScore \leftarrow$
                                CALCULATE_SCORE($FS, s$)
                        if $newScore > score$
                          then
                                $score \leftarrow newScore$
                                $improvement \leftarrow true$
                          else
                                ▷ Undo the change.
                                SWITCH($i, FS[s]$)
          until $improvement = false$
```

To speed up HC, we parallelized it using a speculative execution technique, using one four-core machine for MLP training and MLP forward pass and another one for decoding. Despite this, each of the ten HC experiments took weeks to run, largely because decoder parameter tuning was very time consuming. We repeated the tuning in every iteration of the innermost of the three loops in the HC algorithm, which might have been more than necessary.

4. Results and discussion

We used a two-tailed matched pairs sign test to check whether performance differences are statistical significant (which we will abbreviate to "stat. sig." below). For full details on statistical significance of results see [1][3]. We used a probability of the null hypothesis of 0.05 as the significance threshold.

4.1. Main baseline and hill-climbing results

We will now discuss our main baseline results, which are in Table 1, and our main HC results, which are in Tables 2 and 3. The three-stream ensembles chosen by HC are better than most of the baselines but there is no stat. sig. difference between them and the all-features-concatenated baseline (row (d) of Table 1). In the clean case, the five-stream system chosen by HC has lower word error rate (WER) than any of the baselines, and this is stat. sig. In the noisy case, the difference between it and the best baseline (all features concatenated) is not stat. sig.

HC always lowered WER on the evaluation set, compared to the initial ensemble that HC started with. In nine of ten cases (all but row (b) of Table 2) this improvement was stat. sig.

For three-stream HC, we tried both scoring formulas (Opitz's and ensemble accuracy) and both approaches to choosing initial features (random or non-random). For clean data with RSM initialization, Opitz's formula was better then ensemble accuracy (4.4% final WER vs. 4.6%) but there was no stat. sig. difference in the other cases. For a given scoring formula used to guide HC, it did not make a stat. sig. difference whether or not we chose the initial feature vectors for HC randomly. So in our five-stream experiments we used Opitz's formula and non-random initial feature vectors. Five-stream initial WERs were better than three-stream initial WERs, and the same for final WERs, and this was stat. sig. in most cases.

Experiment	Clean train and test	Noisy train and test
(a) MFCC	6.5	21.4
(b) PLP	5.0	17.5
(c) MSG	7.3	16.3
(d) All features concatenated	4.5	14.7
(e) MFCC, PLP, MSG (three MLPs)	4.9	15.7
(f) RSM (three MLPs)	4.8	17.2
(g) Five MLPs	4.5	15.6

Table 1: Baseline evaluation set WERs. In row (d), all features in the feature pool are concatenated into a single feature vector. Rows (e)-(g) were starting points for HC.

Experiment	Clean train and test		
	Changes	Initial WER	Final WER
(a) Hill-climbing (HC)	4	4.9	4.6
(b) HC, RSM initialization	2	4.8	4.6
(c) HC using $\alpha = 1$	14	4.9	4.5
(d) HC using $\alpha = 1$, RSM initialization	17	4.8	4.4
(e) HC, 5 streams, using $\alpha = 1$	45	4.5	4.2

Table 2: HC results for the clean Numbers corpus. Rows (a)-(d) use three streams and row (e) uses five streams. The "Changes" columns give the number of features changed (added to or deleted from a feature vector) during HC. The "Initial WER" columns give the initial ensemble WER on the evaluation set before the HC algorithm has made any changes to the feature vectors. The "Final WER" columns give the ensemble WER on the evaluation set once HC has finished. If a value is given for α it means that Opitz's formula was used to guide HC. "RSM" means that initial feature vectors were chosen randomly.

Experiment	Noisy train and test		
	Changes	Initial WER	Final WER
(a) Hill-climbing (HC)	5	15.7	14.8
(b) HC, RSM initialization	17	17.2	14.8
(c) HC using $\alpha = 1$	20	15.7	14.8
(d) HC using $\alpha = 1$, RSM initialization	15	17.2	14.9
(e) HC, 5 streams, using $\alpha = 1$	61	15.6	14.2

Table 3: HC results for the noisy Numbers corpus.

4.2. Hill-climbing and overfitting

To investigate whether overfitting of the ensemble affected HC performance, we compared development set ensemble accuracy to evaluation set ensemble accuracy at each stage of HC (see Section 6.5 of [1] and Section 1.6 of [3]). We concluded that performance on the Numbers evaluation set was not significantly lowered by overfitting [7][20] to the development set during the HC process, except perhaps in the case of the noisy corpus with five streams.

4.3. Hill-climbing performance for unseen noises

Did HC improve evaluation set performance for all six noise types in the evaluation set, or only for the four shared noise types (see section 2.2)? Table 4 compares evaluation set WERs for the four shared noises and the two evaluation-only noises. In the three-stream case, HC improved on the systems that it started from even for the evaluation-only noises. In each case, the improvement was stat. sig. for both the shared noises and the evaluation-only noises. The differences in WER between the four three-stream systems chosen by HC were not stat. sig. In the five-stream case, the difference between initial and final WERs for five-stream HC is stat. sig. for the shared noises, but not for the evaluation-only noises. This difference from the three-stream case may be because the initial five-stream WER for evaluation-only noises is 22.4%, considerably lower than for the three-stream systems.

Experiment	Shared noises	Evaluation-only noises
(a) Initial 3 streams, non-RSM	14.1	25.5
(b) Initial 3 streams, RSM	15.0	28.8
(c) Hill-climbing (HC)	13.3	23.4
(d) HC, RSM initialization	13.3	23.6
(e) HC using $\alpha = 1$	13.1	24.2
(f) HC using $\alpha = 1$, RSM init.	13.1	24.5
(g) Initial 5 streams	15.0	22.4
(h) HC, 5 streams, using $\alpha = 1$	12.7	22.7

Table 4: Evaluation set WERs for the noisy Numbers corpus, divided into shared noises and evaluation-only noises. Rows (a)-(f) use three streams.

4.4. Testing the features chosen by hill climbing in a heavily mismatched condition

Are the features chosen by HC simply better features in general, or only better if used for the condition (clean or noisy) that HC was performed for? Tables 5 and 6 show the results when we used the clean Numbers corpus for feature selection, MLP training and decoder parameter tuning, and then tested on the evaluation set from the noisy Numbers corpus.

All three-stream systems chosen by HC performed better in this highly mismatched situation than the initial three-stream systems HC started from, and this was stat. sig. in each case. This is further evidence that improvements in performance from HC can carry over to situations not seen during HC. Using Opitz's formula with non-RSM initialization gave the best performance.

Both the final three-stream system in row (c) of Table 6 and the initial five-stream system in row (e) performed better than the final five-stream system chosen by hill-climbing (this was stat. sig.). However, the final five-stream system chosen by hill-climbing performed better than the systems in the remaining rows of the table (also stat. sig.).

See [1] for further mismatched condition results.

5. Conclusions

For both clean and noisy data sets, HC usually improved performance on held out data compared to the initial system it started with, even for noise types that were not seen during the HC process. On the other hand, HC was quite time consuming. Perhaps it could be sped up by doing less decoder parameter tuning.

In most cases, error rates were roughly comparable whether we guided HC with Opitz's scoring formula or with ensemble accuracy. However, Opitz's formula provided a large advantage when there was an extreme mismatch between training and test.

It should be noted that the error rate of our baselines was well above the best known Numbers result (2.0% for Numbers version 1.0; see section 3.10 of [1]). However, it was comparable to other published OGI Numbers results.

Finally, our HC scripts can be downloaded at [21], along with diagrams of the initial and final feature vectors. We may use the same location in the future if we need to publish updates or corrections regarding our work.

Experiment	WER
(a) MFCC	33.2
(b) PLP	38.0
(c) MSG	30.4
(d) All features concatenated	32.1
(e) MFCC, PLP, MSG (three MLPs)	30.5
(f) RSM (three MLPs)	30.5
(g) Five MLPs	22.9

Table 5: Baseline WERs on the evaluation set in a highly mismatched condition: the clean training set was used with the noisy evaluation set. The feature vectors and MLP hidden layer sizes are the same as in Table 1.

Experiment	Initial WER	Final WER
(a) Hill-climbing (HC)	30.5	29.0
(b) HC, RSM initialization	30.5	28.2
(c) HC using $\alpha = 1$	30.5	23.4
(d) HC using $\alpha = 1$, RSM init.	30.5	27.9
(e) HC, 5 streams, using $\alpha = 1$	22.9	24.1

Table 6: HC results on the evaluation set in a highly mismatched condition: the clean training and development sets used with the noisy evaluation set.

6. References

[1] D. Gelbart, "Ensemble feature selection for multi-stream automatic speech recognition," Ph.D. dissertation, University of California, Berkeley, 2008, online at http://www.icsi.berkeley.edu.

[2] A. Tsymbal, M. Pechenizkiy, and P. Cunningham, "Diversity in search strategies for ensemble feature selection," *Information Fusion*, vol. 6, no. 1, 2005.

[3] D. Gelbart, "Hill-climbing ensemble feature selection with a larger ensemble," International Computer Science Institute, Tech. Rep. 09-001, 2009, online at http://www.icsi.berkeley.edu.

[4] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, 1998.

[5] Corpora group at OGI Center for Spoken Language Understanding, "Numbers version 1.3," http://www.cslu.ogi.edu/corpora.

[6] H. Hirsch and D. Pearce, "The AURORA experimental framework for the performance evaluation of speech recognition systems under noisy conditions," in *ASR2000*, Paris, France.

[7] J. Reunanen, "Overfitting in making comparisons between variable selection methods," *Journal of Machine Learning Research*, vol. 3, 2003.

[8] "Noisy Numbers corpus and Numbers ASR scripts," http://www.icsi.berkeley.edu/Speech/papers/gelbart-ms/numbers.

[9] N. Morgan and H. Bourlard, "Continuous speech recognition: an introduction to the hybrid HMM/connectionist approach," *IEEE Signal Processing Magazine*, May 1995.

[10] H. Wang *et al.*, "The value of auditory offset adaptation and appropriate acoustic modeling," in *INTERSPEECH*, 2008.

[11] H. Hermansky, D. Ellis, and S. Sharma, "Tandem connectionist feature stream extraction for conventional HMM systems," in *ICASSP*, 2000.

[12] Q. Zhu, B. Chen, N. Morgan, and A. Stolcke, "On Using MLP Features in LVCSR," in *ICSLP*, 2004.

[13] D. Ellis, "Stream combination before and/or after the acoustic model," International Computer Science Institute, Tech. Rep. 00-007, 2000, online at http://www.icsi.berkeley.edu.

[14] "SPRACHcore speech recognition tools," http://www.icsi.berkeley.edu/∼dpwe/projects/sprach/sprachcore.html.

[15] D. Opitz, "Feature selection for ensembles," in *Sixteenth National Conference on Artificial Intelligence (AAAI)*, 1999.

[16] L. Kuncheva and L. Jain, "Designing classifier fusion systems by genetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, 2000.

[17] W. Street and Y. Kim, "A streaming ensemble algorithm (SEA) for large-scale classification," in *KDD*, 2001.

[18] J. Hansen and A. Krogh, "A general method for combining predictors tested on protein secondary structure prediction," in *Artificial Neural Networks in Medicine and Biology (ANNIMAB-1)*, 2000.

[19] P. Granitto, P. Verdes, H. Navone, and H. Ceccatto, "Aggregation algorithms for neural network ensemble construction," in *Proceedings of the VII Brazilian Symposium on Neural Networks (SBRN'02)*, 2002.

[20] J. Loughrey and P. Cunningham, "Using early-stopping to avoid overfitting in wrapper-based feature selection employing stochastic search," Trinity College Dublin, Tech. Rep. 2005-37, 2005.

[21] "Numbers hill-climbing scripts and feature vectors," http://www.icsi.berkeley.edu/Speech/papers/gelbart-ms/numbers-hillclimb.