*i*ROVER: Improving System Combination with Classification

D. Hillard[†], B. Hoffmeister[‡], M. Ostendorf[†], R. Schlüter[‡], H. Ney[‡]

†SSLI, Electrical Engineering Dept., University of Washington, Seattle, WA

{hillard,mo}@ee.washington.edu

‡Informatik 6, Computer Science Dept., RWTH Aachen University, Aachen, Germany

{hoffmeister,schlueter,ney}@cs.rwth-aachen.de

Abstract

We present an improved system combination technique, *i*ROVER. Our approach obtains significant improvements over ROVER, and is consistently better across varying numbers of component systems. A classifier is trained on features from the system lattices, and selects the final word hypothesis by learning cues to choose the system that is most likely to be correct at each word location. This approach achieves the best result published to date on the TC-STAR 2006 English speech recognition evaluation set.

1 Introduction

State-of-the-art automatic speech recognition (ASR) systems today usually include multiple contrasting systems, which are ultimately combined to produce the final hypothesis. There is consensus that improvements from combination are usually best when systems are sufficiently different, but there is uncertainty about which system combination method performs the best. In addition, the success of commonly used combination techniques varies depending on the number of systems that are combined (Hoffmeister et al., 2007). In this work, we develop a system combination method that outperforms all previously known techniques and is also robust to the number of component systems. The relative improvements over ROVER are particularly large for combination when only using two systems.

The aim of system combination for ASR is to minimize the expected word error rate (WER) given multiple system outputs, which are ideally annotated with word confidence information. The most widely used system combination approach to date is ROVER (Fiscus, 1997). It is a simple voting mechanism over just the top hypothesis from each component system. Two alternatives that incorporate information about multiple hypotheses and leverage word posterior probabilities are confusion network (CN) combination (Mangu et al., 2000; Evermann and Woodland, 2000) and minimum Time Frame Word Error (min-fWER) decoding (Hoffmeister et al.,

2006), discussed further in the next section. Previous work found that among ROVER, CN combination, and min-fWER combination, no one method was consistently superior across varying numbers and types of systems (Hoffmeister et al., 2007).

The main contribution of this work is to develop an approach that always outperforms other possible system combination methods. We train a classifier to learn which system should be selected for each output word, using features that describe the characteristics of the component systems. ROVER alignments on the 1-best hypotheses are used for decoding, but many of the features are derived from the system lattices. The classifier learns a selection strategy (i.e. a decision function) from a development set and then is able to make better selections on the evaluation data then the current 1-best or lattice-based system combination approaches.

Next, Section 2 describes previous work in system combination techniques. Section 3 describes our approach, and Section 4 provides experiments and results. Finally, we summarize the approach and findings in Section 5.

2 Previous Work

Previous work in speech recognition system combination has produced significant improvements over the results possible with just a single system. The most popular, and often best performing method is ROVER (Fiscus, 1997), which selects the word that the most systems agree on at a particular location (majority voting). An extended version of ROVER also weights system votes by the word confidence produced by the system (confidence voting).

Further improvements have been achieved by including multiple system alternatives, with methods such as Confusion Network Combination (CNC) (Evermann and Woodland, 2000), or N-Best ROVER (Stolcke et al., 2000), which is a special case of CNC. Alternatively, the combination can be performed at the frame level (min-fWER) (Hoffmeister et al., 2006). Recent work found that the best system combination method depended on the number of systems being combined (Hoffmeister et al., 2007). When only two systems are available, approaches considering multiple alternatives per system were bet-

ter, but as the number of systems increased the standard ROVER with confidence scores was more robust and sometimes even better than CNC or min-fWER combination.

Another approach (Zhang and Rudnicky, 2006) used two stages of neural networks to select a system at each word, with features that capture word frequency, posteriors at the frame, word, and utterance level, LM back-off mode, and system accuracy. They obtained consistent but small improvements over ROVER: between 0.7 and 1.7% relative gains for systems with about 30% WER.

3 Approach

We develop a system that uses the ROVER alignment but learns to consistently make better decisions than those of standard ROVER. We call the new system iROVER, where the i stands for improved results, and/or intelligent decisions. The following sections discuss the components of our approach. First, we emulate the approach of ROVER in our lattice preprocessing and system alignment. We then introduce new methods to extract hypothesis features and train a classifier that selects the best system at each slot in the alignment.

3.1 Lattice Preparation

Our experiments use lattice sets from four different sites. Naturally, these lattice sets differ in their vocabulary, segmentation, and density. A compatible vocabulary is essential for good combination performance. The main problems are related to contractions, e.g. "you've" and "you have", and the alternatives in writing foreign names, e.g. "Schröder" and "Schroder". In ASR this problem is well-known and is addressed in scoring by using mappings that allow alternative forms of the same word.

Such a mapping is provided within the TC-STAR Evaluation Campaign and we used it to normalize the lattices. In case of multiple alternative forms we used only the most frequent one. Allowing multiple parallel alternatives would have distorted the posterior probabilities derived from the lattice. Furthermore, we allowed only one-to-one or one-to-many mappings. In the latter case we distributed the time of the lattice arc according to the character lengths of the target words.

In order to create comparable posterior probabilities over the lattice sets we pruned them to equal average density. The least dense lattice set defined the target density: around 25 for the development and around 30 for the evaluation set.

Finally, we unified the segmentation by concatenating the lattices recording-wise. The concatenation was complicated by segmentations with overlapping regions, but our final concatenated lattices scored equally to the original lattice sets. The unified segmentation is needed for lattice-based system combination methods like frame-based combination.

3.2 System Alignments

In this work we decided to use the ROVER alignment as the basis for our system combination approach. At first glance the search space used by ROVER is very limited because only the first-best hypothesis from each component system is used. But the oracle error rate is often very low, normally less than half of the best system's error rate.

The ROVER alignment can be interpreted as a confusion network with an equal number of arcs in each slot. The number of arcs per slot equals the number of component systems and thus makes the training and application of a classifier straightforward.

For the production of the alignments we use a standard, dynamic programming-based matching algorithm that minimizes the global cost between two hypothesis. The local cost function is based on the time overlap of two words and is identical to the one used by the ROVER tool. We also did experiments with alternative local cost functions based on word equalities, but could not outperform the simple, time overlap-based distance function.

3.3 Hypothesis Features

We generate a cohort of features for each slot in the alignment, which is then used as input to train the classifier. The features incorporate knowledge about the scores from the original systems, as well as comparisons among each of the systems. The following paragraphs enumerate the six classes of feature types used in our experiments (with their names rendered in italics).

The primary, and most important feature class covers the *basic* set of features which indicate string matches among the top hypotheses from each system. In addition, we include the systems' frame-based word confidence. These features are all the information available to the standard ROVER with confidences voting.

An additional class of features provides extended *confidence* information about each system's hypothesis. This feature class includes the confusion network (CN) word confidence, CN slot entropy, and the number of alternatives in the CN slot. The raw language model and acoustic scores are also available. In addition, it includes a frame-based confidence that is computed from only the acoustic model, and a frame-based confidence that is computed from only the language model score. Frame-based confidences are calculated from the lattices according to (Wessel et al., 1998); the CN-algorithm is an extension of (Xue and Zhao, 2005).

The next class of features describes *durational* aspects of the top hypothesis for each system, including: character length, frame duration, frames per character, and if the word is the empty or null word. A feature that normalizes the frames per character by the average over a window of ten words is also generated. Here we use characters as a proxy for phones, because phone information is not available from all component systems.

We also identify the system dependent *top error* words for the development set, as well as the words that occur to the left and right of the system errors. We encode this information by indicating if a system word is on the list of top ten errors or the top one hundred list, and likewise if the left or right system context word is found in their corresponding lists.

In order to provide *comparisons* across systems, we compute the character distance (the cost of aligning the words at the character level) between the system words

and provide that as a feature. In addition, we include the confidence of a system word as computed by the frame-wise posteriors of each of the other systems. This allows each of the other systems to 'score' the hypothesis of a system in question. These cross-system confidences could also act as an indicator for when one system's hypothesis is an OOV-word for another system. We also compute the standard, confidence-based ROVER hypothesis at each slot, and indicate whether or not a system agrees with ROVER's decision.

The last set of features is computed relative to the combined *min-fWER* decoding. A confidence for each system word is calculated from the combined frame-wise posteriors of all component systems. The final feature indicates whether each system word agrees with the combined systems' min-fWER hypothesis.

3.4 Classifier

After producing a set of features to characterize the systems, we train a classifier with these features that will decide which system will propose the final hypothesis at each slot in the multiple alignment. The target classes include one for each system and a null class (which is selected when none of the system outputs are chosen, i.e. a system insertion).

The training data begins with the multiple alignment of the hypothesis systems, which is then aligned to the reference words. The learning target for each slot is the set of systems which match the reference word, or the null class if no systems match the reference word. Only slots where there is disagreement between the systems' 1-best hypotheses are included in training and testing.

The classifier for our work is Boostexter (Schapire and Singer, 2000) using real Adaboost.MH with logistic loss (which outperformed exponential loss in preliminary experiments). Boostexter trains a series of weak classifiers (tree stumps), while also updating the weights of each training sample such that examples that are harder to classify receive more weight. The weak classifiers are then combined with the weights learned in training to predict the most likely class in testing. The main dimensions for model tuning are feature selection and number of iterations, which are selected on the development set as described in the next section.

4 Experiments

We first perform experiments using cross-validation on the development set to determine the impact of different feature classes, and to select the optimal number of iterations for Boostexter training. We then apply the models to the evaluation set.

4.1 Experimental setup

In our experiments we combine lattice sets for the English task of the TC-STAR 2006 Evaluation Campaign from four sites. The TC-STAR project partners kindly provided RWTH their development and evaluation lattices. Systems and lattice sets are described in (Hoffmeister et al., 2007).

Table 1 summarizes the best results achieved on the single lattice sets. The latter columns show the results of

	Viterbi		min-fWER		CN	
	dev	eval	dev	eval	dev	eval
1	10.5	9.0	10.3	8.6	10.4	8.6
2	11.4	9.0	11.4	9.5	11.6	9.1
3	12.8	10.4	12.5	10.4	12.6	10.2
4	13.9	11.9	13.9	11.8	13.9	11.8

Table 1: *WER[%] results for single systems.*

CN and min-fWER based posterior decoding (Mangu et al., 2000; Wessel et al., 2001).

4.2 Feature analysis on development data

We evaluate the various feature classes from Section 3.3 on the development set with a cross validation testing strategy. The results in Tables 2 and 3 are generated with ten-fold cross validation, which maintains a clean separation of training and testing data. The total number of training samples (alignment slots where there is system disagreement) is about 3,700 for the 2 system case, 5,500 for the 3 system case, and 6,800 for the 4 system case.

The WER results for different feature conditions on the development set are presented in Table 2. The typical ROVER with word confidences is provided in the first row for comparison, and the remainder of the rows contain the results for various configurations of features that are made available to the classifier.

The *basic* features are just those that encode the same information as ROVER, but the classifier is still able to learn better decisions than ROVER with only these features. Each of the following rows provides the results for adding a single feature class to the *basic* features, so that the impact of each type can be evaluated.

The last two rows contain combinations of feature classes. First, the best three classes are added, and then all features. Using just the best three classes achieves almost the best results, but a small improvement is gained when all features are added. The number of iterations in training is also optimized on the development set by selecting the number with the lowest average classification error across the ten splits of the training data.

Features	2 System	3 System	4 System
ROVER	10.2%	8.8%	9.0%
basic	9.4%	8.6%	8.5%
+confidences	9.3%	8.7%	8.4%
+durational	9.2%	8.6%	8.4%
+top error	9.0%	8.5%	8.4%
+comparisons	8.9%	8.6%	8.4%
+min-fWER	8.5%	8.5%	8.4%
+top+cmp+fWER	8.3%	8.3%	8.2%
all features	8.3%	8.2%	8.2%

Table 2: WER results for development data with different feature classes.
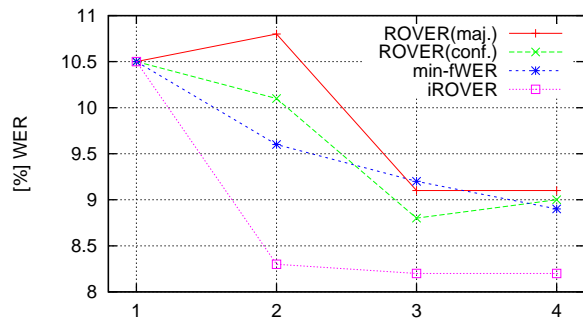
	2 System	3 System	4 System
ROVER (maj.)	10.8%	9.1%	9.1%
ROVER (conf.)	10.1%	8.8%	9.0%
min-fWER	9.6%	9.2 %	8.9 %
*i*ROVER	8.3%	8.2%	8.2%
oracle	6.5%	5.4%	4.7%

Table 3: *WER[%] results for development data with manual segmentation, and using cross-validation for iROVER.*

4.3 Results on evaluation data

After analyzing the features and selecting the optimal number of training iterations on the development data, we train a final model on the full development set and then apply it to the evaluation set. In all cases our classifier achieves a lower WER than ROVER (statistically significant by NIST matched pairs test). Table 3 and Table 4 present a comparison of the ROVER with majority voting, confidence voting, frame-based combination, and our improved ROVER (*i*ROVER).

5 Conclusions

In summary, we develop *i*ROVER, a method for system combination that outperforms ROVER consistently across varying numbers of component systems. The relative improvement compared to ROVER is especially large for the case of combining two systems (14.5% on the evaluation set). The relative improvements are larger than any we know of to date, and the four system case achieves the best published result on the TC-Star English evaluation set. The classifier requires relatively little training data and utilizes features easily available from system lattices.

Future work will investigate additional classifiers, classifier combination, and expanded training data. We are also interested in applying a language model to decode an alignment network that has been scored with our classifier.

References

G. Evermann and P. Woodland. 2000. Posterior probability decoding, confidence estimation and system combination. In *NIST Speech Transcription Workshop*.
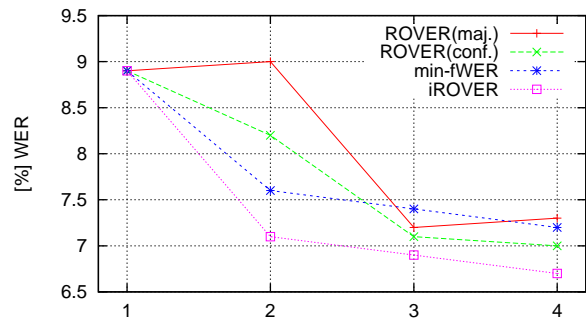
	2 System	3 System	4 System
ROVER(maj.)	9.0%	7.2%	7.3%
ROVER(conf.)	8.2%	7.1%	7.0%
min-fWER	7.6 %	7.4 %	7.2 %
*i*ROVER	7.1%	6.9%	6.7%
oracle	5.2%	4.1%	3.6%

Table 4: *WER[%] results for evaluation data.*

J.G. Fiscus. 1997. A post-processing system to yield reduced word error rates: Recognizer Output Voting Error Reduction (ROVER). In *Proc. ASRU*.

B. Hoffmeister, T. Klein, R. Schlüter, and H. Ney. 2006. Frame based system combination and a comparison with weighted ROVER and CNC. In *Proc. ICSLP*.

B. Hoffmeister, D. Hillard, S. Hahn, R. Schülter, M. Ostendorf, and H. Ney. 2007. Cross-site and intra-site ASR system combination: Comparisons on lattice and 1-best methods. In *Proc. ICASSP*.

L. Mangu, E. Brill, and A. Stolcke. 2000. Finding consensus in speech recognition: word error minimization and other applications of confusion networks. *Computer Speech and Language*, 14:373–400.

R. E. Schapire and Y. Singer. 2000. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168.

A. Stolcke, H. Bratt, J. Butzberger, H. Franco, V. Gadde, M. Plauche, C. Richey, E. Shriberg, K. Sonmez, J. Zheng, and F. Weng. 2000. The SRI March 2000 Hub-5 conversational speech transcription system. In *NIST Speech Transcription Workshop*.

F. Wessel, K. Macherey, and R. Schlüter. 1998. Using word probabilities as confidence measures. In *Proc. ICASSP*.

F. Wessel, R. Schlüter, and H. Ney. 2001. Explicit word error minimization using word hypothesis posterior probabilities. In *Proc. ICASSP*, volume 1.

Jian Xue and Yunxin Zhao. 2005. Improved confusion network algorithm and shortest path search from word lattice. In *Proc. ICASSP*.

R. Zhang and A. Rudnicky. 2006. Investigations of issues for using multiple acoustic models to improve continuous speech recognition. In *Proc. ICSLP*.