

Sub-Problem Selection for Acoustic Code-Breaking

Veera Venkataramani
Center for Language and Speech Processing
The Johns Hopkins University
3400 N. Charles Street, Baltimore, MD 21218, U.S.A.
veera@jhu.edu

William Byrne
Cambridge University Engineering Department
Trumpington Street, Cambridge, CB2 1PZ, U.K.
wjb31@cam.ac.uk

Technical Report No. CUED/F-INFENG/TR.563

December 2006

Abstract

We have developed lattice rescoring procedures which apply specialized acoustic and language models, e.g. discriminatively trained HMMs and Support Vector Machines, to improve the transcription performance of general purpose ASR systems. Initial investigations showed that large gains in small vocabulary tasks are possible, but that large improvements in large vocabulary recognition are harder to achieve. In this paper we analyze the feasibility of the entire approach. We find that sub-problems be identified reliably, and that the approach has potential, but that the selection of sub-problems for solution should consider additional factors, such as the availability of training data.

This report analyzes a particular previously developed decoding framework [1, 2, 3] designed to identify possible recognition errors in a first-pass recognition hypothesis; Support Vector Machines were then used to improve the original ASR hypotheses. Here we analyze the improvements obtained to better understand the decoding framework. We also present methods for selecting a larger number of sub-problems in large vocabulary tasks that offer scope for additional improvement.

The work reported here was performed at The Johns Hopkins University Center for Language and Speech Processing with support from the NSF (U.S.A) under the Information Technology Research (ITR) program, NSF IIS Award No. 0122466. This report is based on a submission to Interspeech 2005.

1 Introduction

Acoustic code-breaking is a novel approach to rescoring in automatic speech recognition which involves both search-space refinement and specialized acoustic and language models. The overall approach is first to perform an unsupervised analysis of first-pass recognition lattices generated by a good HMM-based ASR system. This identifies ‘regions of confusion’ in the lattices, where the baseline hypothesis might be weak. Within the overall recognition task, these regions of confusion can be considered as recognition sub-problems, in that they are limited to a small and finite set of word hypotheses. Using training data subsets chosen for the purpose, we construct specialized classifiers over the hypotheses in each of these distinct sub-problems, and finally apply these classifiers in rescoring the original lattices.

The goal is to obtain small-vocabulary recognition performance on large vocabulary continuous speech recognition tasks, and to do so by transforming the original large vocabulary problems into a series of smaller problems that are more easily solved. We and others have reported positive results on both small and large vocabulary recognition tasks [4, 2, 3, 5] using specialized classifiers based on both HMMs and Support Vector Machines (SVMs).

The approach is quite general. We obtain specialized classifiers by training general-purpose systems, such as HMMs or SVMs, on training data sets constructed for each specific sub-problem. Motivated by the observation that small vocabulary decoders may perform better on small tasks than general purpose large vocabulary recognizers, we hope to improve the performance of the baseline large ASR system within these specific sub-problems. For example, the baseline system must be general enough to distinguish between (say) ‘B’, ‘V’, and ‘E’, but if ‘E’ can be eliminated as an alternative, a classifier capable only of distinguishing between ‘B’ and ‘V’ could resolve the remaining confusion. Code-breaking also allows us to use classifiers that would otherwise be unsuitable for continuous speech recognition. We are able to use SVMs because we can reliably find sub-problems consisting of word pairs, which we can resolve with binary classifiers such as SVMs.

In this paper we analyze the improvements obtained in our previous work with the specific goal of determining what gains can be expected from this approach to ASR. The main question is whether it is possible to identify and solve enough sub-problems to achieve significant performance improvements in large vocabulary continuous speech recognition. The first concern is what effect the transformation of first-pass lattices into sequential sub-problems has on the overall Lattice Word Error Rate (LER), which is the upper bound on improvement in rescoring. Ideally, we should attempt to solve as many sub-problems as possible, however this results in an explosion in the number of specialized classifiers required. The problem lies not the total number of classifiers, but in whether there is adequate data to train all of them reliably. The second line of analysis in this paper is whether we can cluster recognition sub-problems to ensure adequate training data.

2 Experiments in Code-Breaking

Given a baseline ASR system, we use an unsupervised lattice cutting algorithm [6] to identify ASR sub-problems within test set lattices. Lattice cutting first aligns all lattice paths to the baseline hypothesis (Fig. 1, b), and then collapses all subpaths in the lattice so that they align with words in the baseline hypothesis (Fig. 1, c). This converts the lattice into a sequence of smaller sub-lattices. Note that no paths are lost in this process; in fact, new paths are added and the oracle Lattice Error Rate (LER) of the lattice in Fig. 1, c is typically much lower than that of the original. Pruning of arcs in sublattices can be performed using posterior probabilities derived from the baseline lattice (Fig. 1, d). The baseline systems rely on HMMs trained under ML, MMI, or Pinched Lattice MMI (PLMMI) [4] criteria. PLMMI is MMI performed over training set lattices which are aligned to the reference transcriptions and pruned as above.

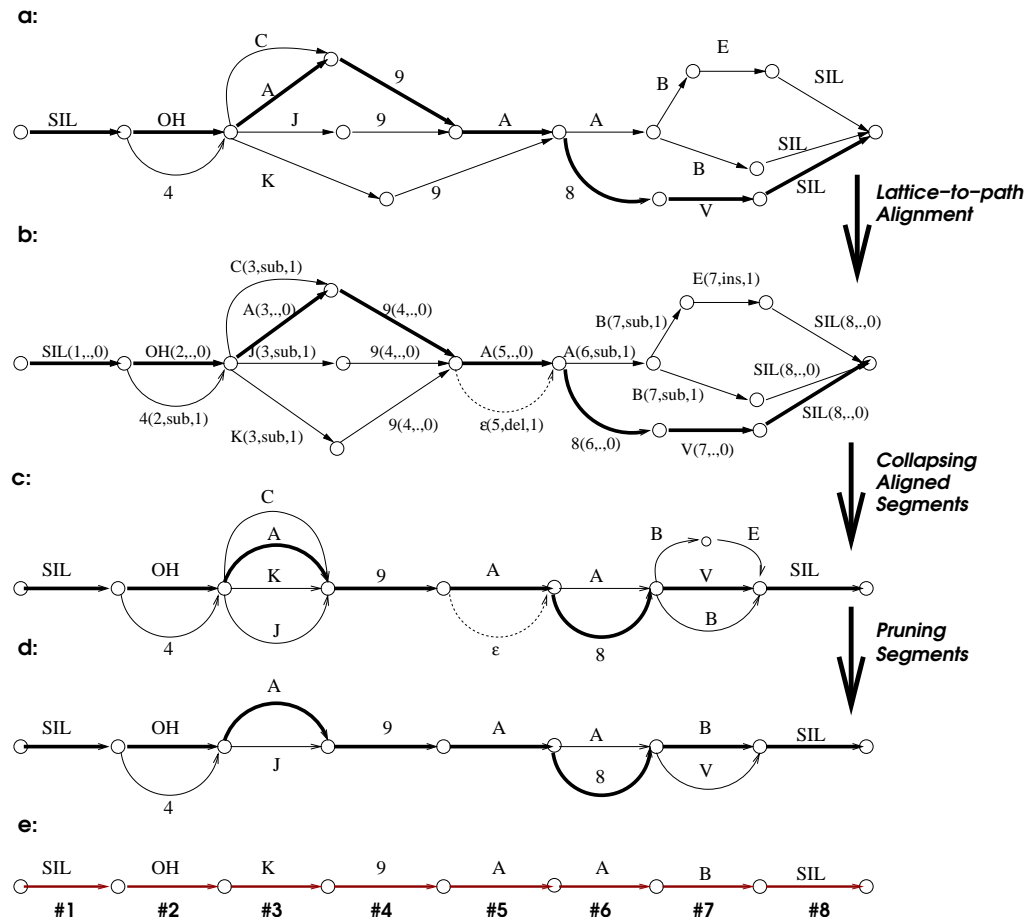


Figure 1: Lattice Segmentation. *a*: First-pass lattice with MAP path in bold; *b*: Aligned lattice; *c*: Collapsed segment sets; *d*: Refined Search Space. *e*: The truth.

We have used both word-level HMMs and SVMs as our specialized classifiers. Since this analysis extracts small vocabulary sub-problems (e.g. Fig. 1, d), we can use MMI as in small vocabulary tasks to train HMMs for use in specialized classifiers. To train SVMs, we use the score-space approach developed by Smith *et al.* [7]; here, the patterns used by the SVMs are derived directly from these word level HMMs, and this provides a fixed-dimensional representation for both SVM training and classification. We have evaluated these techniques over the OGI-Alphadigit corpus [4, 2] and the MALACH spontaneous Czech conversational domain [8, 3].

2.1 Code-Breaking in Small Vocabulary ASR

The OGI-Alphadigit is a small vocabulary corpus of 26 letters and 10 numbers; utterances are random six word strings. Our experiments are based on ~50000 training and ~3000 test utterances. The acoustics were parametrized as 39 dimensional MFCC vectors with delta and acceleration coefficients. Each word was modeled by a left-to-right HMM with ~20 states; these were trained with MMI. For each model set, the AT&T decoder was used to generate lattices. We then pinched and restricted the lattices to the 50 most frequently occurring confusion pairs (*CPairs*); typical confusions were {B,V} and {F,S}. Table 1 reports the overall word error rates of the baseline HMM systems,

HMM Training	HMM	SVM
ML	10.73	8.63
MMI	9.07	8.10
PLMMI	7.98	8.13

Table 1: WERs of HMM and SVM Code-Breaking Systems on the Alphadigit Task.

and results using specialized SVMs to select the path through each test set CPair [2]. For ML and MMI baselines, SVMs yield substantial improvements. However, it is interesting that SVMs failed to improve the PLMMI HMM baseline; we investigate this in Sections 3.1 and 3.2.

2.2 Code-Breaking in Large Vocabulary ASR

The Czech language baseline system for MALACH is based on continuous mixture density, tied state, cross-word, gender-independent, triphone HMMs trained with HTK using 40 hours of transcribed speech (24065 utterances) parameterized as MFCC vectors with delta and acceleration coefficients. The AT&T Large Vocabulary Decoder was used to generate lattices over the test set with a bigram language model based on a 83000 word vocabulary. Lattice-based MMI [4] was then performed. The test set consisted of ~ 8400 utterances spoken by ten held-out speakers (~ 25 hours of speech). Unsupervised MLLR transforms for each of the test-set speakers were estimated on a 1000 utterance test subset. The baseline system produced test set lattices with WER of 45.6% and a LER of 22.3%. These lattices were then pinched (Fig. 1,c). We prune away all paths in a segment with a posterior < 0.10 and then only keep behind binary confusions alone; if a segment set survived pruning but contained more than two hypotheses, it was pruned back to the MAP hypothesis. This yields what we call *natural* confusion pairs (naturally occurring binary confusion sets).

We trained word level MMI HMMs for each confusion pair, and SVMs were trained in the score-spaces of these HMMs. Unlike Section 2.1, we did not train the SVMs using training set CPairs extracted from lattices. We instead used *pseudo-ambiguity*: if a CPair contains the words $\{w_1, w_2\}$, find all instances of w_1 and w_2 in training, and gather statistics from every utterance under both hypotheses [4, 2], and in this way create a training set for supervised discriminative training for each CPair.

During decoding, we interpolate the posterior distribution estimated by the SVM and the baseline HMM system to choose a word in a segment set by voting. This results in a WER improvement of 0.1% with a significance test p-value of less than 0.001. We found that SVMs were able to improve over the baseline hypothesis within most of the confusion sets. The overall influence on WER is small, of course, given the small number of confusion sets studied.

3 Behavior of PLMMI HMMs and SVMs

We now analyze the Alphadigits systems to understand the behavior of the PLMMI HMMs and PLMMI SVMs.

3.1 Segment Set Selection Bias

We demonstrate that the selection of segment sets within lattices is tied to the baseline models which generate them. In Table 2, LER varies as the baseline lattices are pruned to binary pair lattices, and then further restricted to contain only the 50 most frequent binary CPairs found within the MMI lattices. Fixing the CPairs allows us to study performance over a static set of sub-problems.

Baseline HMMs	Baseline Lattices	All Pairs	50 Pair Subset
ML	2.00	4.87	5.60
MMI	1.70	4.07	4.65
PLMMI	1.80	3.93	4.68

Table 2: LER of Baseline and Binary Pinched Lattices. The most common CPairs under MMI are the fixed ‘50 Pair Subset’.

Note first that the LERs of the original lattices generated from the ML and the MMI models are comparable (2.00% *vs.* 1.70%), but that LER for the MMI models are significantly better (4.87% *vs.* 4.07%). The binary pinched MMI lattices are better than the binary pinched ML lattices. We counted the total number of binary confusions that appeared in each set of lattices, and found that each set contains ~ 8400 confusions. The improvement in LER is not due to a higher number of identified confusions by the MMI models; rather it is due to the better quality of confusions identified. We conclude that the MMI models are simply better at generating binary confusions, in that the confusions identified are more likely to contain the truth as one of the alternatives.

Note also that the performance of the PLMMI models are significantly better than that of the MMI models (7.98% *vs.* 9.07%). However, there is not a corresponding reduction in LER (4.65% *vs.* 4.68%). This can be partially explained by the observation that the number of CPairs drops after PLMMI training: there were ~ 8400 instances for the MMI case and ~ 5400 instances for the PLMMI case, and thus less scope for improvement. This result can be further explained when we consider lattices containing all possible binary confusions. Here, we see that the PLMMI lattices do give a better search space than the MMI lattices (4.07% *vs.* 3.93%). This suggests why the PLMMI SVMs were unable to improve the PLMMI baseline: the PLMMI SVMs had been trained to resolve the confusions of the MMI models, not the confusions of the PLMMI models. We conclude that sub-problems need to be selected for each baseline system, especially as the underlying models improve.

3.2 Training data sparsity for PLMMI SVM systems

We investigate whether training data sparsity may account for the inability of the PLMMI SVM classifiers to reduce the PLMMI WER. PLMMI training of the underlying HMMs lowers the training set WER relative to MMI. If the PLMMI SVMs are trained using the CPair instances found in the PLMMI training set, there will simply be less training data relative to MMI: the PLMMI SVMs were trained with $\sim 80,000$ CPair instances, compared to $\sim 120,000$ instances for the MMI SVMs.

Figure. 2 shows how the overall WER is influenced by the amount of training data available for each CPair SVM. Starting from the PLMMI WER baseline, we incrementally apply the SVM classifiers to test set CPairs, and plot the overall WER as the SVM classifiers are applied. The SVM application order is based on the amount of training data for each SVM; the SVM with the largest training set is applied first. We see that the WER gains come from the SVMs with the most training data, and that by the application of the 20th SVM (which has 1500 training tokens), the improvements over the baseline have vanished. We also include for comparison the MMI SVM performance over these test set CPairs; the MMI SVMs are significantly poorer than the PLMMI SVMs when evaluated over the same test data.

The results of Sections 3.2 and 3.1 suggest two important issues in selecting sub-problems. First, the sub-problems need to be matched to the errors being made by the baseline HMM system, since the error patterns change as the baseline systems improve. Second, the selection process should consider whether there is adequate training data.

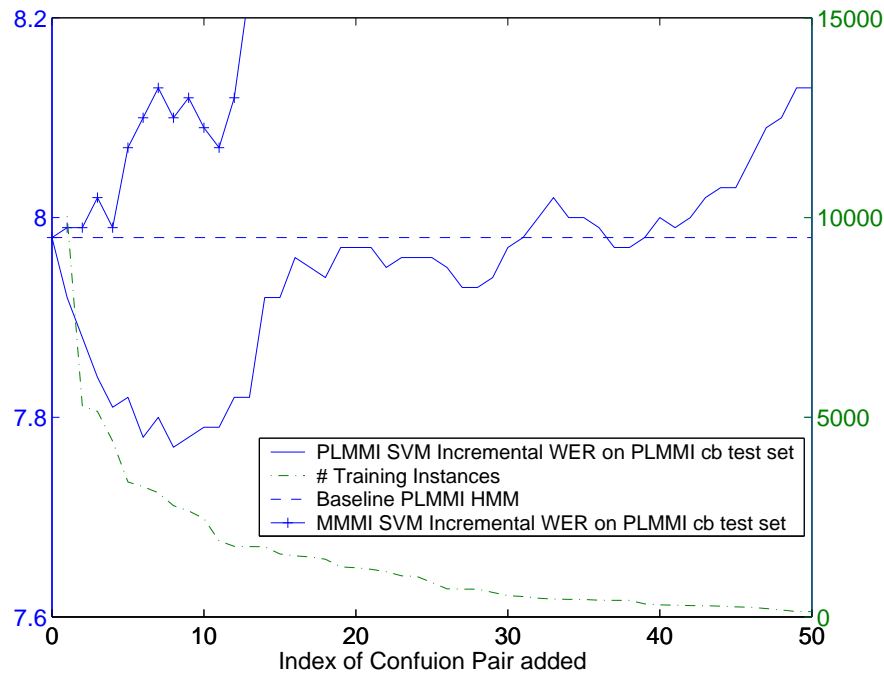


Figure 2: PLMMI SVM WER and Training Set Size.

4 Enlarging the Search Space

We have shown that applying SVMs to the frequently occurring confusions found in an LVCSR task can produce small yet statistically significant improvements [3]. These improvements were modest by design, since the code breaking test set was restricted to confusion pairs that occurred at least 100 times in the test set. This restriction was enforced to provide enough test data to evaluate each classifier. However it is not an inherent limitation, and we now discuss methods to identify larger code-breaking test sets with greater scope for improvement.

4.1 Word Pair Diversity

Figure 3 shows how LER varies as we filter the test set confusion pairs by their frequency of occurrence. We first prune lattice arcs by discarding those with a posterior probability of less than 0.1; then we count the number of occurrences of each natural CPair; then we measure LER by discarding pairs based on frequency of occurrence. If we retain all pairs (threshold value of 1), the LER is 41.6%, which provides a potential $\sim 4\%$ reduction from the baseline WER of 45.6%. We see a great increase in the types of CPairs as the observation count threshold decreases, and we note that much of the LER improvement is due to the CPairs observed least frequently in the test set.

We wish to develop an unsupervised classification procedure that applies specialized classifiers only to those segments for which we are confident that the baseline system hypothesis is in error; in other words, we wish to be confident that we are not introducing errors into hypotheses that are already correct. To assess the feasibility of this, we characterize each segment set as either ‘MAPCOR’ or ‘MAPERR’, corresponding to whether or not the MAP hypothesis within a segment is correct. This is measured simply by aligning the pinched and restricted lattice (Fig. 1,d) to the truth (Fig. 1,e) and counting the errors. In a similar way, we also label segment sets as CPOC or

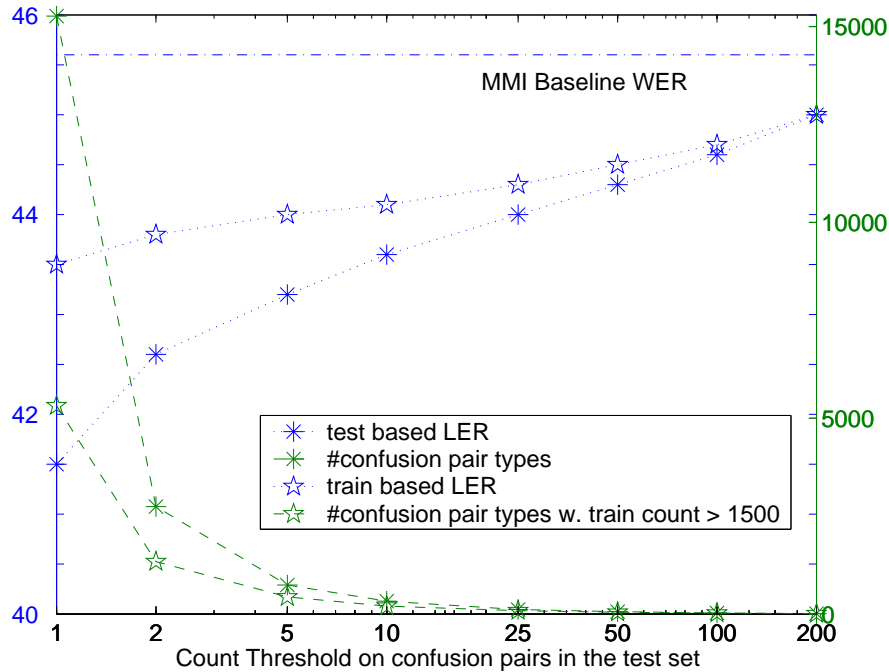


Figure 3: LVCSR LER and CPair Test Set Frequency.

CPOE (Confusion Pair Oracle Correct/Error) based on whether a segment contains the truth. Ideally, we will find many CPOC sub-problems with a relatively high ratio of MAPERR to MAPCOR hypotheses; in these, the baseline is wrong and the alternative hypothesis is correct.

Results are listed in Table 3. In focusing on Oracle Correct confusion pairs which occur 100 times or more in the test set, we find the ratio of MAPERR to MAPCOR to be 0.48. This is encouraging, since this is the operating point at which we have demonstrated the feasibility of using such classifiers to produce statistically significant WER improvements [3]. We note that the pruning step of (Fig. 1,d) is crucial; without it, the MAPERR/MAPCOR ratio is ~ 0.15 , and in our experience it is very difficult to improve the baseline at this operating point, given the great risk of degrading the baseline at this operating point.

Referring to the Figure 3 at the count threshold of 1, we can see that if we could apply 6802 word confusion classifiers robustly, then there is scope for an improvement of 4% over the 45.6% WER of the baseline system. While it would be possible, although difficult, to apply the thousands of classifiers needed to resolve the least frequently occurring binary confusions, the real obstacle is doing so is not computational; the difficulty is to find enough training data for all these classifiers.

We now consider robust classifier training. In the analysis of Figure 2, we saw that we needed ~ 1500 instances of training for each SVM. With this as a target training set size, we measure the LER of lattices containing only sub-problems which are observed at least 1500 times in the acoustic training data. As shown in Fig. 3, we see that we still have an opportunity of a 2.1% WER improvement over the baseline system when we focus on problems with rich training sets. We note also that (Table 3) that the ratio of MAPERR to MAPCOR is still healthy within these sub-problems.

Test Count	All		Training Count ≥ 1500	
	MAPERR	R	MAPERR	R
1	6802	0.51	3558	0.47
10	3324	0.50	2439	0.47
50	2191	0.49	1859	0.46
100	1695	0.48	1528	0.45
200	1063	0.43	1063	0.43

Table 3: Analysis of CPOC CPairs. MAPERRs and R(= #MAPERR/#MAPCOR) for All CPairs, and for CPairs Observed More Than 1500 Times in the Training Set.

Phone Edit Distance	Phone Edit		Word CPair		
	Tokens	Types	Types	R	LER
1	11243	236	2878	0.56	43.8
≤ 2	18591	344	3010	0.54	42.9
≤ 3	23003	289	2868	0.47	42.5

Table 4: CPairs Clustered by Phone-Edit-Distance Between the Word Hypotheses. LER of lattices when pruned back to contain CPairs with specified phone edit distance.

4.2 Sub-Problem Phonetic Equivalence Classes

The preceding analysis shows that we can identify a collection of sub-problems which offer significant improvement over the baseline and for which we can find adequate training data. However the number of distinct word-level classifiers is still unwieldy. Fortunately there is redundancy among these distinct problems. In particular, many word pair confusions can be solved by the same *phonetic classifier*. As a trivial example, a {p/b} classifier can resolve both of these word pair confusions: {PAT/BAT} and {PACK/BACK}.

We assign word confusion pairs to equivalence classes based on their phone errors. For example, the above two word pairs would belong to the same CPair. We achieve this by replacing each word label in a CPair instance by the phone string hypothesized by the baseline ASR system. We then measure the phone edit distance between these phone strings.

We give the distribution of the CPairs as a function of the number of edit operation between the phone strings in Table 4; for example, /p a t/ and /b a t/ differ by 1, while /p a t/ and /b a k/ differ by 2. We see that this analysis identifies 236 binary phone confusions; if we train classifiers for these, we can apply them to 11243 confusion instances with the potential of reducing the baseline WER from 45.6% to 43.8%. The 236 phonetic classifiers needed is a dramatic reduction from the 11243 classifiers that would be needed to solve these problems at the word level. If we can train classifiers to distinguish between two phone strings which differ by two of their constituents, we can further reduce the WER to 42.9%.

4.3 Greedy Selection of N-Hypothesis Sub-Problems

Much of the above analysis was motivated by the application of SVMs to LVCSR. In their simplest formulation, SVMs are binary classifiers. It is possible to obtain a more promising code-breaking test set if we relax our criteria to include more confusions, for example {PAT, BAT, HAT} as a 3-way sub-problem. Specifically, we derive confusion sets containing more than one alternative hypothesis path by considering the n-most-likely alternatives. We prune the original lattices at posterior threshold of 0.05, which raises the original lattice LER to 35.4%, and we retain

all confusions, not just the ‘natural’ ones. We also allow each hypothesis to contain more than one word string, e.g. {SAMOROST, SME MU DOST}. Table 5 shows that simply by focusing on ternary word confusions, there is the possibility of an almost 10% WER reduction from the baseline, and there is little need to consider more than 4-hypothesis problems. For $N=2$, we find an R value of 0.43. This suggests that our previous scheme of selecting “naturally” occurring CPairs is conservative.

Hypotheses (N)	2	≤ 3	≤ 4	≤ 5	all
LER	38.0	36.2	35.6	35.4	35.4

Table 5: LER for N-Hypothesis Sub-Problems.

5 Conclusions

We have analyzed the performance of two code-breaking systems. In studying a small vocabulary system, we found that sub-problems should be selected from the errors being made by the best system; keeping sub-problems fixed as models improve appears to be suboptimal. We also found a dependence on the amount of training data associated with the sub-problems. In large vocabulary systems, we investigated several methods to increase the potential gains in code-breaking. We find that there is considerable potential improvement to be had by training a relatively small number of sub-word (phonetic) classifiers which can be each be shared over a relatively large number of diverse word-level sub-problems.

References

- [1] V. Venkataramani, S. Chakarabarty, and W. Byrne, “*Gini* support vector machines for segmental minimum Bayes risk decoding of continuous speech,” *Computer, Speech, and Language*, Published online by Elsevier Ltd., 2 October 2006.
- [2] V. Venkataramani, S. Chakarabarty, and W. Byrne, “Support vector machines for segmental minimum Bayes risk decoding of continuous speech,” in *ASRU*, 2003.
- [3] V. Venkataramani and W. Byrne, “Lattice segmentation and support vector machines for large vocabulary continuous speech recognition,” in *ICASSP*, 2005.
- [4] V. Doumptiotis, S. Tsakalidis, and W. Byrne, “Lattice segmentation and minimum Bayes risk discriminative training,” in *Eurospeech*, 2003.
- [5] M. J. F. Gales and M. Layton, “Svms, score-spaces and maximum margin statistical models,” in *ATR Beyond HMMs Workshop*, 2004.
- [6] V. Goel, S. Kumar, and W. Byrne, “Segmental minimum Bayes-risk decoding for automatic speech recognition,” *IEEE Trans. on Speech and Audio Processing*, May 2004.
- [7] N. D. Smith and M. J. F. Gales, “Using SVMs and discriminative models for speech recognition,” in *ICASSP*, 2002.
- [8] W. Byrne et al., “Automatic recognition of spontaneous speech for access to multilingual oral history archives,” *IEEE Trans. on Speech and Audio Processing*, July, 2004.