LATTICE SEGMENTATION AND SUPPORT VECTOR MACHINES FOR LARGE VOCABULARY CONTINUOUS SPEECH RECOGNITION

Veera Venkataramani [1] *and William Byrne* [1,2]

Center for Language and Speech Processing, The Johns Hopkins University,
3400 N. Charles Street, Baltimore, MD 21218, U.S.A. [1]
Cambridge University Engineering Department
Trumpington Street, Cambridge, CB2 1PZ, U.K. [2]
veera@jhu.edu , wjb31@cam.ac.uk

ABSTRACT

Lattice segmentation procedures are used to spot possible recognition errors in first-pass recognition hypotheses produced by a large vocabulary continuous speech recognition system. This approach is analyzed in terms of its ability to reliably identify, and provide good alternatives for, incorrectly hypothesized words. A procedure is described to train and apply Support Vector Machines to strengthen the first pass system where it was found to be weak, resulting in small but statistically significant recognition improvements on a large test set of conversational speech.

1. INTRODUCTION

Acoustic Code-breaking is a divide-and-conquer approach to Automatic Speech Recognition (ASR). The process starts by analyzing word lattices generated by a good ASR system to pick out 'regions of uncertainty'. These are portions of the lattice (sublattices) where the first-pass ASR system is less than certain about the words in its primary hypothesis (also referred to as the MAP lattice path). Moreover, these sublattices contain words and phrases which can be considered as likely alternatives to the primary hypothesis. The original acoustic model (or language model) is weak over these regions in that the system was unable to pick a clear winner from among the competing hypotheses; these sub-lattices essentially define sub-problems that remain after the first recognition pass. The idea behind acoustic code-breaking is to attack these problems using special-purpose models trained specifically to find the truth in these sets of competing hypotheses.

This approach is motivated by several considerations. Even powerful discriminative training procedures such as MMI do not address all errors uniformly well. It is possible, for example, for MMI to improve the overall word error rate even while performance over some individual error-types actually degrades [1, 2]. Code-breaking is a way to avoid that problem (see also [3]).

Another consideration is that different word recognition problems require different types of decisions. The first-pass acoustic model has little *a priori* knowledge about the speech to be recognized, and so it must be capable of choosing between (say) 'A' and '8' and also 'A' and 'J'. Code-breaking makes it possible to use a specialized decoder capable of distinguishing between 'A' and 'J' without having to distinguish between 'A' and '8'. Without the refinement of the first-pass hypotheses, such a simple choice would never occur in large vocabulary continuous speech recognition.

Specialized HMMs can be trained to solve the problems identified by code-breaking [4], however the approach can also employ novel classifiers, such as Support Vector Machines (SVMs) [5, 6, 7, 8], to resolve acoustic confusion identified by the first-pass decoder.

SVMs are essentially binary pattern classifiers, and to use them in this context, we restrict the regions of confusion to be word pairs, called *confusion pairs*. This approach of using SVMs has been demonstrated and validated on small vocabulary, continuous speech recognition tasks [9, 10]. In this paper we demonstrate the code-breaking is a general approach through which powerful but simple classifiers can be incorporated into large vocabulary speech recognition systems.

All the steps in code-breaking become more challenging in a large vocabulary recognition task. If we select a confusion pair to be 'fixed', we need to be fairly confident that (a) the MAP hypothesis is actually wrong in that pair, and that (b) the other hypothesis is actually the truth. While we have found that both these issues are manageable in small vocabulary tasks, in large vocabulary tasks, we face sparsity issues due to the diversity of word confusions that arise; it may be difficult to ensure the presence of the truth in a confusion pair. We will study the degree to which we can ensure (a) and (b). To demonstrate that the overall approach is feasible, we will show that we can employ SVMs to reduce error rates within the confusion pairs so as to produce statistically significant improvements relative to the baseline MMI ASR system.

2. UNSUPERVISED IDENTIFICATION OF ASR SUBPROBLEMS VIA LATTICE PINCHING

Lattice segmentation converts a first-pass lattice into a sequence of smaller sub-lattices through a Levenshtein alignment of the lattice to a reference path [11]. Here, test set lattices (Fig. 1, a) are aligned to the primary hypothesis so that word sequences from the lattice are aligned with words in the primary hypothesis (Fig. 1, b). In Period-1 lattice cutting [11], all competing paths are collapsed with respect to a single word in the reference path. This produces *segment sets*, which are groups of substrings from the lattice identified as alternatives to words in the primary hypothesis (Fig. 1, c). Note that no lattice paths are discarded yet. In fact, new lattice paths are usually introduced; the oracle Lattice Error Rate (LER)

This work was supported by the NSF (U.S.A) under the Information Technology Research (ITR) program, NSF IIS Award No. 0122466.

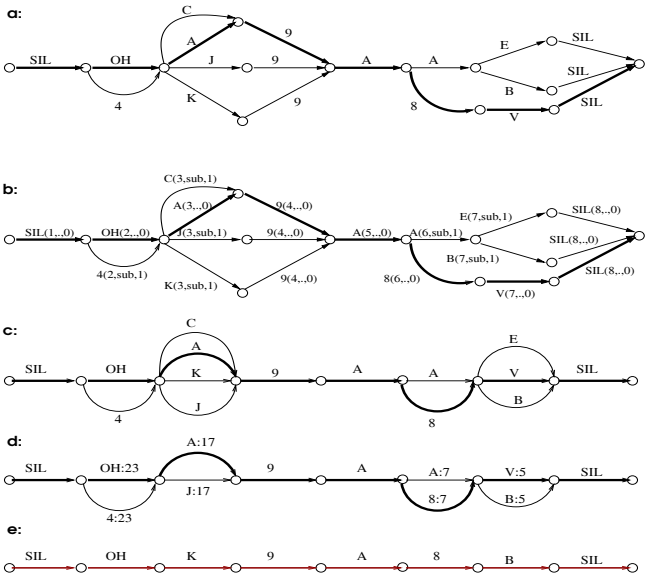

Fig. 1. Lattice Segmentation. *a*: First-pass lattice with MAP path in bold; *b*: Alignment of the lattice to the MAP path under the Levenshtein distance; the link labels give the word hypothesis, segment index, edit operation, and its alignment cost); *c*: Collapsed segment sets; *d*: Refined Search Space consisting of binary segment sets. Word hypotheses are tagged so specialized models can be used in lattice rescoring. *e:* The truth.

of the lattice in Fig. 1, c is typically much lower than that of the original lattice in Fig. 1, a.

Some of the segment sets can contain NULL links. These are contributed by lattice paths that are shorter than the reference. The presence of a NULL requires answering the question "Should the word in the primary hypothesis be deleted?". Since the MAP hypothesis could easily contain a wrongly inserted word or phrase, this is clearly a problem of interest and specialized detectors could be built to attack it. But in this paper we ignore this problem. We simply discard the NULL links knowing an increase in the LER will result.

While the segment sets are defined by alignment under the Levenshtein distance, the joint acoustic and language model scores in the initial lattice are also preserved so that posterior distributions can be defined over the segment sets. This allows us to prune the segment sets to finally obtain confusion pairs (Fig. 1, d).

This process of alignment, segmentation, and pruning defines the sub-problems that will be attacked in code-breaking. We refer to the overall process as *lattice pinching* and we now analyze how well it performs in (a) identifying weaknesses in the primary hypothesis and (b) providing useful alternatives. We begin by describing our large vocabulary ASR task.

3. EFFECTIVENSS OF LATTICE PINCHING

We evaluate our approach in the MALACH spontaneous Czech conversational domain [12]. The system consists of speaker independent continuous mixture density, tied state, cross-word, gender-independent, triphone HMMs trained with HTK using 40 hours of transcribed speech (24065 utterances). The speech was parameterized into 39-dimensional, MFCC coefficients, with delta and acceleration coefficients. The AT&T Large Vocabulary Decoder was used to generate lattices over the training and test sets with a bigram language model based on a 83000 word vocabulary. Lattice-based MMI [13, 1] was then performed. The test set consisted of approx. 8400 utterances spoken by ten held-out speakers (approx. 25 hours of speech). Unsupervised MLLR transforms for each of the test-set speakers were estimated on a 1000 utterance subset of the test set. The baseline system produced a test set lattices with WER of 45.6% and 22.3% LER.

Pruning Threshold	LER	Avg. # Hyps. / Segment Set	Segment Sets	
			Types	Tokens
0.00	27.3	11.65	94029	1393099
0.05	35.3	2.82	49837	212852
0.10	37.9	2.35	35278	134252
0.20	41.1	2.06	17132	63267
0.30	43.2	2.00	7288	26913
0.40	44.7	2.00	2249	7930
0.50	45.6	-	0	0

Table 1. Lattice Pinching and LER. The average number of hypotheses per segment set, number of distinct segment sets, and total number of segment sets after posterior-based pruning. Threshold 0.0 corresponds to Fig. 1 c with NULL hypotheses discarded.

We first analyze the change in oracle Lattice Error Rate that results from pinching the test set lattices. Table 1 shows that discarding the NULL hypotheses from the segmented lattices, without any pruning, increases the LER to 27.3%. If we were to process these lattices with special-purpose classifiers, these classifiers would need to be able to distinguish between 11.65 hypotheses on average, and if these classifiers were to perform perfectly, they would lower the WER from 45.6% to 27.3%. Since we wish to apply binary classifiers, the analysis at the 0.3 pruning threshold of 0.3 is relevant, since 'on average' pinching produces binary confusion pairs. While the best performance that can be obtained is a WER of 43.2%, we stress that this is improvement over a well-trained large vocabulary ASR system on a very difficult test set.

We consider only those confusion pairs that occur in the test data at least 100 times. This is not a necessary restriction and it does further limit our potential improvement, but it simplifies our analysis in that there are enough instances of each pair to reliably measure recognition performance over each of them. Referring to Figure 1 d, only these frequently occurring confusion pairs are retained, and all others are pruned back to the primary hypothesis.

To further analyze the confusion pairs, we Levenshtein-align the pinched lattices (Fig 1 d) to the truth (Fig 1 e). We first count the number of Confusion Pair Errors (CPERR), which are confusion pairs that don't contain the truth. For example, in Fig. 1 d, (A:17, J:17) is classified as CPERR since it does not contain the true word 'K'; the other sets are classified as Confusion Pair Oracle Correct (CPOC). While it is desirable to produce as few CPERR sets as possible, those CPERR instances that do occur can be ignored. These are 'lost causes', where lattice pinching failed to provide a good alternative and further processing can pick randomly from the confusion pair without any meaningful effect on the overall WER. Of course, if a set can be guessed to be CPERR, there is the opportunity to add hypotheses, perhaps to fix OOV problems.

The CPOC segments are those which we are interested in. Within the CPOC segments we can distinguish those in which the

Pruning Threshold	#CPOC/ #CPERR	#MAPERR/ #MAPCOR	Segment Sets	
			Types	Tokens
0.00	14.30	0.24	22	7324
0.05	4.7	0.64	26	8022
0.10	3.3	0.92	26	6860
0.20	3.2	1.17	17	3831
0.30	4.2	1.15	6	1405
0.40	11.0	1.04	2	337
0.50	-	-	0	0

Table 2. Ratio of #CPOC/#CPERR segments and #MAPERR/#MAPCOR segments for the confusion pairs observed at least 100 times in the 25 hour test set.

MAP path agrees with the oracle path (MAPCOR) and those in which the MAP path is in error (MAPERR). In Fig. 1, d the pair (V:5, B:5) is classified as MAPERR, and the pairs (OH:23, 4:23) and (A:7, 8:7) are MAPCOR;both these sets are CPOC.

We further process the pinched lattices constructed from the frequently occurring confusion pairs. We renormalize these lattices to define the posterior distribution over these binary confusion pairs, and again apply a posterior-based pruning to these instances of the confusion pairs. The results are as reported in Table 2. At a pruning threshold of 0.4, the surviving confusion pairs are high quality: the CPERR pairs occur far less frequently than CPOC pairs; and within these the the MAPERR count is about equal to the MAPCOR count, so about half the MAP hypotheses are incorrect. Unfortunately, there are only two distinct confusion pairs and pruning eliminates all but 337 instances of them. In the subsequent experiments, we prune at a threshold of 0.1. At this level, we still have three times as many CPOC pairs as CPOERR, the system is still making errors roughly half the time (MAPERR $\approx$ MAPCOR), and we have a diverse test set of 6860 observations of 26 distinct confusion pairs. While the 0.1 threshold value reported here was chosen in a supervised fashion, we subsequently verified that 0.1 is also the optimal threshold found by splitting the test set by speakers and using one half as a held-out set and the other half as the test set. We conclude that this threshold can be determined robustly as part of the modeling process.

Lattice pinching can also be used to identify weaknesses in the underlying language model. For example if a confusion pair contains homonyms, acoustic-phonetic information cannot easily be used to distinguish them. This homonym problem is very severe in Czech due to the mixing of standard and colloquial Czech in conversational speech [12], as with the Czech words 'BYLI' and 'BYLY', both pronounced `B I L I`. It is possible to train specialized language models for such homonym confusion pairs, but we focus here on acoustic code-breaking experiments. Restricting ourselves to non-homonym confusion pairs further reduces the number of confusion pairs to 21 with 2991 total observations.

To recap the selection of test set confusion pairs, we prune from the collapsed segment sets any path whose posterior probability is less than 0.10. After pruning, we keep only confusion pairs: any confusion set with more than two hypotheses is pruned back to the primary hypothesis. We then restrict the confusion pairs to those that occur atleast 100 times. Finally, homonym confusion pairs are also pruned back to the primary hypothesis.

The relative increase in the number of MAPERRs as the threshold increases strongly suggests that code-breaking should be done so that the baseline posterior distribution over the confusion pairs is considered in the decoding process. We have developed simple voting procedures for this [9, 10], to be described in Sec. 4.3.

4. CODE-BREAKING

4.1. Acoustic HMMs for Confusion Pairs

We begin by training special purpose HMMs for the words in the confusion pairs. A set of multiple Gaussian mixture monophone HMMs are trained over the acoustic training set, and these models are also used to align the training set. Whole-word acoustic models for the words in confusion pairs are initialized with these monophone models and they are then then reestimated using Baum Welch over word segments extracted from the aligned training set.

We next clone these whole-word models for the confusion pairs, e.g. the model for the word 'A' is replicated so that A:17 and A:7 are two different whole-word HMMs. To train the models for the confusion pair (A:7, 8:7), an acoustic training subset is created by extracting all the acoustic segments for 'A' and '8' from the training data. MMI is then used to further train the models A:7 and 8:7 over this training subset. This process is repeated for all of the confusion pairs, and in this way, the models are specialized to discriminate between the words in the confusion pairs. Throughout all this we keep track of pronunciation variation. For example, the word 'TAK' has pronunciations `T A K` and `T A G`, and the word 'PAK' has only the pronunciation `P A K`. Models are trained for all three instances, and `T A K` *vs.* `P A K` and `T A G` *vs.* `P A K` would be considered as two distinct confusion pairs.

4.2. Acoustic SVMs for Confusion Pairs

We now have all that is needed to train acoustic SVMs for the binary confusion pairs. We use the score-space approach developed by Smith and Gales [14, 9, 10]. Statistics derived from the likelihood of speech segments under the HMMs are used to convert a variable-length sequence into a static fixed-dimensional representation which can be used in SVM training and classification; the dimension is derived from the number of parameters in each HMM. We do not review this approach, other than to stress that the use of whole-word models allows us to fix the score space for every instance of each confusion pair. If we were to derive the score space from triphones, say, the score space would then depend on context within which the word pair occurs. This may in fact be desirable, but it would greatly complicate the modeling approach.

SVMs were trained using the $Gini$SVM toolkit [15] for 21 non-homonym confusion pairs. The MMI trained word HMMs were used to generate mean and likelihood-ratio scores. All SVMs were trained in 20% of the most informative dimensions (chosen by a Fisher-like criterion [14]). We noticed that performace of the SVMs was similar for a range of dimensionalities of the score-space used (15% to 25%). We used a tanh kernel and a global SVM trade-off parameter of 1.0. More details of the SVM training procedure can be found in Venkataramani *et al.* [9].

4.3. SVM-MAP Hypothesis Combination

For each of the 21 confusion pairs, Figure 2 reports performance of the baseline HMM system, the SVM decoders, and a hybrid decoder combining the two. The baseline performance over each confusion pair is the left-most of each of the three bars. The decision is made simply by picking the most likely alternative under the lattice posterior; this likelihood is based on the triphone HMM acoustic score, with MLLR, and the bigram language model. An

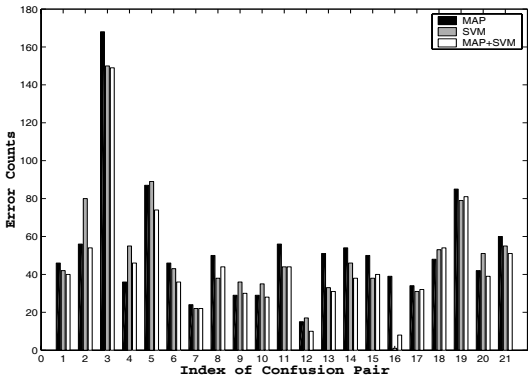

Fig. 2. Error counts over individual confusion pairs.

error occurs when picking the wrong word relative to the Levenshtein alignment of the pinched lattice to the truth; e.g. in Fig. 1 d, picking V:5 would count as an error.

For each confusion pair instance, the appropriate discriminatively trained whole-word was used to create score-space features for use in classification by the SVM trained for that pair. The performance over each of the 21 confusion pairs is given in the center bars of the plots in Fig. 2. Performance relative to the MAP baseline is mixed; there are not consistent improvements.

To combine the SVM and MAP decisions, a posterior distribution over the SVM decisions was estimated by logistic regression [10]. This associates a confidence (estimated likelihood of being correct) with each SVM choice. For a particular instance of a confusion pair with words (w_1, w_2), let $p_h(w)$ be the MAP posterior over the pinched lattices, and $p_s(w)$ be the SVM confidence in each decision. A simple linear interpolation

$p_\lambda(w_i) = \lambda p_h(w_i) + (1-\lambda)p_s(w_i)$ for $0 \leq \lambda \leq 1$

gives a combined likelihood over the word pair. With $\lambda = 0.5$, the performance over the 21 pairs by this SVM-MAP combination system is given in the third of the bars in Fig. 2. Under this combination, the error count decreases in 18 of the 21 pairs.

The influence of these reductions on the overall WER over the complete 25 test set is necessarily limited, for the reasons already discussed. The main reason is that the 2991 words in the code-breaking test set are only a small portion of the complete 25 hour test set. Under the MAP-SVM combination system, the baseline MAP WER is reduced from 45.6% to 45.5%. However small, these gains are statistically significant and stable with respect to λ: we obtained this performance improvement for $\lambda = 0.4, 0.5, 0.6,$ and $, 0.7$, and in all instances the significance test p-values [16] were less than 0.001.

5. CONCLUSIONS

We have presented a general modeling approach for incorporating special purpose classifiers into a large vocabulary recognition system. Possible errors in the first-pass recognition hypotheses are identified by lattice pinching, and specialized decoders are trained and applied to fix these errors. We have shown that SVM binary classifiers can in this way be gainfully added to a large vocabulary ASR system. We constructed our experiments so that the potential gains are modest, but this does not reflect any insurmountable limitation in the approach. Expanding the code-breaking test set, perhaps by more permissive pruning or acoustic clustering of confusion sets, will provide opportunity for greater improvements.

Acknowledgments. Thanks to AT&T for the AT&T decoder and FSM libraries. Baseline MMI HMM models were trained by V. Doumpiotis; S. Chakrabartty provided the *Gini*SVM toolkit and much advice, as did S. Kumar.

6. REFERENCES

[1] V. Doumpiotis, S. Tsakalidis, and W. Byrne, "Lattice segmentation and minimum Bayes risk discriminative training," in *Eurospeech*, 2003.

[2] V. Doumpiotis and W. Byrne, "Lattice segmentation and minimum Bayes risk discriminative training for large vocabulary continuous speech recognition," *Speech Communication*, *Submitted*.

[3] P. C. Woodland and D. Povey, "Minimum phone error and I-smoothing for improved discriminative training," in *Proc. ICASSP*, 2002.

[4] V. Doumpiotis, S. Tsakalidis, and W. Byrne, "Discriminative training for segmental minimum Bayes risk decoding," in *ICASSP*, 2003.

[5] V. Vapnik, *The Nature of Statistical Learning Theory*, chapter 5, Springer-Verlag, New York, Inc., 1995.

[6] A. Ganapathiraju, J. Hamaker, and J. Picone, "Advances in hybrid SVM/HMM speech recognition," in *GSPx / International Signal Processing Conference*, 2003.

[7] S. E. Golowich and D. X. Sun, "A support vector/hidden Markov model approach to phoneme recognition," in *ASA Proceedings of the Statistical Computing Section*, 1998, pp. 125–130.

[8] J. Salomon, S. King, and M. Osburne, "Framewise phone classification using support vector machines," in *Proc. ICSLP*, 2002.

[9] V. Venkataramani and W. Byrne, "Support vector machines for segmental minimum Bayes risk decoding of continuous speech," in *ASRU*, 2003.

[10] V. Venkataramani, S. Chakarabarty, and W. Byrne, "*Gini*support vector machines for segmental minimum Bayes risk decoding of continuous speech," *Comp. Spch. Lang.*, *Submitted*.

[11] V. Goel, S. Kumar, and W. Byrne, "Segmental minimum Bayes-risk decoding for automatic speech recognition," *IEEE Transactions on Speech and Audio Processing*, May 2004.

[12] W. Byrne et al., "Automatic recognition of spontaneous speech for access to multilingual oral history archives," *IEEE Trans. Speech and Audio Proc.*, July, 2004.

[13] P. C. Woodland and D. Povey, "Large scale discriminative training for speech recognition," in *Proc. ITW ASR*, 2000.

[14] N. D. Smith and M. J. F. Gales, "Using SVMs and discriminative models for speech recognition," in *ICASSP*, 2002.

[15] S. Chakrabartty, *The giniSVM toolkit, Version 1.2*, Available: http://bach.ece.jhu.edu/svm/ginisvm/, 2003.

[16] D. Pallett, W. Fisher, and J. Fiscus., "Tools for the analysis of benchmark speech recognition tests.," in *ICASSP*, 1990.