DETECTION OF UNSEEN WORDS IN CONVERSATIONAL MANDARIN

Ivan Bulyko, Owen Kimball, Man-Hung Siu, José Herrero and Dan Blum

Raytheon BBN Technologies, Cambridge, MA 02138, USA
{ibulyko,okimball,msiu,jherrero,dblum}@bbn.com

ABSTRACT

We present a Mandarin keyword search system that uses a large vocabulary recognizer to generate consensus networks at various resolutions: word, character, syllable and phone. In order to achieve fast and accurate search, we propose the use of an efficient approximate-match dynamic programming algorithm that finds the best alignment between the target query and the consensus network. Experiments with Mandarin conversational telephone speech show that the approximate-match search improves detection accuracy by more than 10% for rare words that are not present in the recognizer's dictionary (OOV terms). We also found OOV terms to benefit most from system combination, where we observe a roughly 10% improvement relative to the best single system.

Index Terms— Spoken term detection, OOV, Mandarin

1. INTRODUCTION

Spoken term detection (STD) or keyword search (KWS) is an important application of speech recognition technology. A substantial research literature supports the use of large vocabulary continuous speech recognition (LVCSR) for accurate keyword search. Specifically, using lattices generated by an LVCSR system has been shown to be crucial for achieving high recall and improved accuracy in KWS [1,2]. In addition to accuracy, fast retrieval speed is important in KWS, but searching lattices presents a computational challenge. Various methods of efficient indexing and search of lattices have been proposed, e.g. [3,4]. Consensus networks [5] offer a compact representation of the lattice, encapsulating the full richness of recognition hypotheses in a structured linear form, which can be exploited by search algorithms. In the past consensus networks have been shown [6] to perform as well or better than lattices in KWS tasks. The keyword search system described in this paper uses consensus networks because of their computational advantages.

However, even deep lattices can have inadequate recall. This problem is particularly serious when we search for long phrases. One solution is to expand the search space by allowing search errors, such as insertions, deletions and substitutions. Various proposals have been made that include using syllable-level pronunciation automata [7], a phonetic confusion matrix [8] and query expansion [9] to recover from recognition errors. In this paper, we describe a new approximate match algorithm based on a simple filter followed by dynamic programming. We present experiments showing that the algorithm is very effective at detecting out-of-vocabulary words and is computationally efficient.

Out-of-vocabulary (OOV) words, i.e., those not in the recognizer's dictionary, can never appear in a lattice and therefore must be retrieved using subword units such as syllables and phones. Word lattices can be converted to their corresponding subword units, as we do in this paper, or one can build a subword or a hybrid word-subword recognition system [10] that generates lattices with subword units. Since Mandarin does not have well defined "words", the notion of OOV words is a bit unclear. Large vocabulary recognizers contain all individual characters in addition to many multi-character words. This gives the word recognizer an ability to generate any sequence of characters, thus making it similar in function to a hybrid word-subword recognizer. In our experiments, a query term is labeled as OOV when the underlying sequence of characters (judged by a human to be a single word) does not appear in the recognizer's dictionary.

Word-based recognition is known to give better accuracy in Mandarin STT compared to using character-based models. This can be attributed to better phonetic context modeling as well as a more powerful n-gram language model when words are being used. However, the ambiguous word segmentation in Mandarin forces us to segment character strings into words using automatic tools based on a predefined dictionary. This leads to biasing the recognition model to a word segmentation that may not be optimal for keyword detection. Users of a Mandarin keyword search system look for character sequences, unlike in other languages where users look for words or word phrases. Therefore performance must be evaluated with the character sequence matching criterion, rather than being constrained to "words" per a given segmentation. For example, if "China National Bank" is marked as one word in the reference, its location, returned as a search result for "China," should be scored as correct since the two characters corresponding to "China" are correctly detected. Since KWS performance is evaluated based on character sequence matching, a word-based index is not optimal for Mandarin keyword search. In this paper we evaluate keyword search accuracy with units at various granularities: word, syllable, and phone.

System combination is an effective tool in improving keyword search accuracy. Various forms of system combination have been explored in the past: a) outputs from different recognizers are combined into a single lattice or consensus network [6]; b) lattices from different recognizers are searched independently and the search results are combined [11]; c) lattices from a single recognizer are searched at various resolutions (e.g. word and subword) or using different search methods and the search results are combined [12]; and d) any combination of the above [13]. In this paper we use a single word-based recognizer to produce consensus networks of different granularities: word, syllable and phone. We then search each resulting network and combine the results, achieving significant improvement in KWS of OOV terms.

2. KEYWORD SEARCH SYSTEM

Our keyword search is based on BBN's Byblos LVCSR system [14], which uses state-of-the-art discriminatively trained acoustic models and performs MLLR speaker adaptation. The output of the

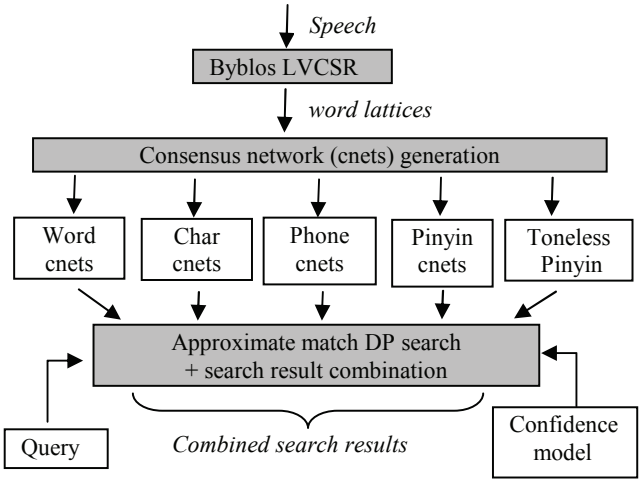

Figure 1. Mandarin keyword search system.

recognizer is a word lattice, which we transform into a consensus network [5] for the purpose of running keyword search (Figure 1). In Mandarin, we convert the word lattices to character-, syllable- or phone-level lattices by splitting each word arc to a sequence of character or phone arcs that correspond to this word. We then convert the resulting sub-word lattices to consensus networks.

The keyword search is accomplished by finding the query term (represented by a sequence of units, i.e. words, characters, phones or syllables) in the consensus network. A match is returned when we find a sequence of arcs in the consensus network that corresponds to the target.

Unless the original word lattices are heavily pruned, the consensus networks capture the acoustic confusability between words or sub-word units quite well. Oracle error rate of a lattice is typically a factor of two or more lower than the one-best error rate. However, when OOV words occur, word recognizers tend to make errors that propagate around the OOV region. The resulting consensus networks of subword units tend to be biased toward the existing vocabulary and some previously observed word sequences. This makes detection of OOV terms more difficult, because the sequence of subword units that corresponds to the OOV term is much less likely to be present in the consensus network. Even for IV terms, when a query is long (e.g. multi-word phrases), finding the whole sequence of words in the consensus network becomes less likely, leading to high precision, but low recall.

Recall can be improved significantly by allowing approximate matches within the consensus network to be returned, where insertions, deletions or substitutions within the target are evaluated with some predefined probability of mismatch, *Pm*. As we later show in the experiments, allowing approximate matches does not overwhelm the system with false alarms. We align the target query represented by a sequence of units (phones, characters, etc.) to the consensus network using dynamic programming (DP), an algorithm analogous to the one used in computing word error rate. The algorithm returns the best alignment, measured by the product of 1) consensus network arc posteriors (including skip arcs) used by the alignment, and 2) probability of mismatch *Pm* for each instance of deletion, insertion or substitution.

Our implementation of the DP algorithm is very efficient, making only a single pass through the consensus network. Its complexity is O(nm), where n is the length of the query and m is the number of nodes in the consensus network. Furthermore, we make it possible to identify utterances that are not likely to contain a match before we run the DP search and then skip those utterances. Specifically, for each consensus network we store n-gram sequences (order 3) that occur. At search time we check the percentage of n-grams from the target query that are present in the consensus network and skip the utterance if the number is below a user-defined threshold that can be changed to set the tradeoff between high recall and high speed. When we used a setting that required at least half of the target trigrams to be present in the utterance, we were able to double the search speed, reaching hundreds of hours of speech per second of search, with minimal loss in performance (around 0.01 loss in AUC).

Each hit returned by the DP search is assigned a confidence score which is computed using a Generalized Linear Model (GLM). The input to the GLM is a fixed-length vector of numerical features. For each search result, the GLM computes a weighted sum of the input features and applies the sigmoid function to the sum. We used features that are language-independent and easy to compute from the consensus networks during the DP search: number of phonemes in the target; the geometric mean of confidences and the product of confidences in the best alignment, either including or excluding phoneme errors; the number and fraction of target phonemes that were matched; and the number and fraction of phonemes that did not align to the consensus network. We have also tried using phonetic and lexical features but did not see any significant benefit.

3. EXPERIMENT SETUP

In these experiments we used the Hub5 Dev04 Mandarin CTS test set (about 3 hours of speech). The recognizer was trained with 250 hours of transcribed speech available from LDC. The training transcripts were automatically segmented into words using a 24K-word dictionary. The phone set contained 78 phonemes that included tones [14]. The recognizer's performance was measured as follows: a) 1-best character error rate (CER) was 34.2%, b) lattice oracle CER was 14.7%, and c) 1-best phone error rate (PER) was 25.9%.

The test set reference transcripts were segmented into words by a human annotator. This gave us a set of meaningful query terms, some of which were not present in the recognizer's dictionary. For query terms, we used all words in the test set with the exception of the 100 most frequent words. The query terms were divided into two sets: in-vocabulary (IV) and out-of-vocabulary (OOV) based on the recognizer's dictionary. Table 1 shows various characteristics of the two query lists. As one can see, OOV terms are about 10 times less likely to occur than IV terms. Also OOV terms on average are about 30% longer than IV terms as measured in characters or phonemes.

	IV	**OOV**
Number of terms	1827	502
Average occurrences per term	17.2	1.8
Average term length in characters	1.8	2.4
Average term length in phonemes	5.1	6.8

Table 1. Characteristics of the query lists.

The query terms, defined as character sequences, must be converted to phone sequences or pinyin before we can search in

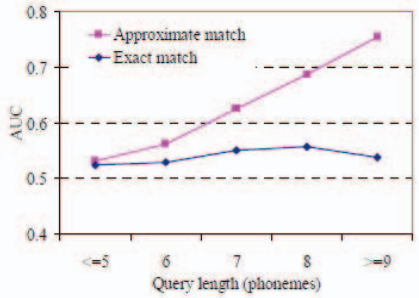

Figure 2. Benefits of the approximate match search for OOV terms at different query lengths.

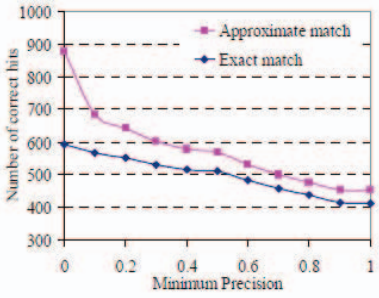

Figure 3. Approximate match search improves recall on OOVs while generating more high precision hits.

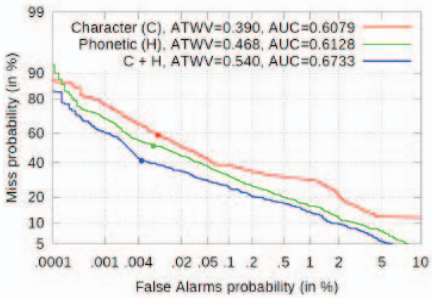

Figure 4. Term-weighted DET curves show improvement in OOV detection accuracy from system combination.

System	Approx match?	IV terms			OOV terms		
		AUC	ATWV	OTWV	AUC	ATWV	OTWV
W Word-based	N	0.614	0.437	0.559	N/A	N/A	N/A
H' Phonetic	N	0.626	0.402	0.557	0.536	0.430	0.563
H Phonetic	Y	0.626	0.395	0.559	**0.613**	**0.468**	**0.646**
C' Character	N	0.709	**0.497**	**0.645**	0.524	0.412	0.544
C Character	Y	**0.712**	0.490	**0.645**	0.608	0.390	0.600
P' Pinyin	N	0.655	0.427	0.579	0.507	0.406	0.536
P Pinyin	Y	0.658	0.413	0.585	0.595	0.410	0.624
T' Toneless P	N	0.605	0.392	0.529	0.538	0.433	0.576
T Toneless P	Y	0.605	0.383	0.538	0.591	0.433	0.633
C + W		0.723	0.488	0.643	N/A	N/A	N/A
C + H		**0.725**	**0.515**	**0.647**	0.673	0.540	0.693
C + P		0.711	0.497	0.633	0.639	0.520	0.664
C + T		0.718	0.505	0.643	0.646	0.477	0.670
C + H + P		0.723	0.512	0.645	0.676	**0.541**	0.695
C + H + T		0.722	0.501	0.646	0.684	0.485	0.681
C + H + P + T		0.722	0.499	0.644	**0.685**	0.460	**0.697**

Table 2. KWS accuracy with various indexing/searching methods and their combinations on Mandarin Dev04 CTS test set (CER = 34%).

phonetic or pinyin consensus networks. We used a trainable text-to-phoneme (T2P) model to do this conversion. The model is trained in three steps (see [15] for details): 1) the characters and phonemes (or pinyin) in the training dictionary are aligned using an iterative EM-based algorithm, 2) contextual features are extracted from the alignments and 3) a decision tree mapping characters to phonemes is built based on the contextual features. The resulting model's T2P accuracy was 95% at both phoneme and pinyin levels, measured on the full 24K-word dictionary using 5-fold cross-validation, with the vast majority of the errors attributable to incorrect tone values.

4. EVALUATION

For both sets of query lists, we report keyword search accuracy in terms of three different metrics: AUC, ATWV (Actual Term Weighted Value) with automatically set confidence threshold, and OTWV (ATWV with optimally set threshold).

AUC, which stands for "area under curve", is a variant of the mean average precision (MAP) metric, but unlike typical MAP, it penalizes queries with no returned hits. It is defined as:

$$AUC = \frac{1}{|Q|}\sum_{q\in Q}\frac{1}{N(q)}\sum_{r\in R(q)}P(q,r) \qquad (1)$$

where Q is the set of queries with non-empty references, $N(q)$ is the number of possible hits for query q, $R(q)$ is the set of ranks of correct hits for q, and $P(q,r)$ is precision for query q at rank r.

AUC measures the quality of search result rankings; it is not sensitive to the absolute value of hit confidences and it does not require setting a confidence threshold. AUC, which takes on values in the range [0,1], is high when correct hits are ranked high in the list of results. Note that AUC is not weighted by the word frequency, i.e. both rare and common words contribute equally to the overall score.

ATWV, which stands for "actual term-weighted value" was introduced by NIST in the 2006 STD evaluation [16]. It requires the system to make a hard decision by setting a confidence threshold. Setting the threshold correctly for each query term (see [1] for details) requires an accurate estimate of the expected count for that term as well as accurate estimation of word confidences.

OTWV (or optimal TWV) is the maximum ATWV that can be achieved with the given search results when the threshold is set optimally. It can also be thought of as the value of the search results when the system is not required to make a hard decision.

5. RESULTS

The 3-hour Dev04 test set was decoded once using word-based models and a 24K-word phonetic dictionary, yielding 1-best character error rate (CER) of 34.2% and lattice oracle CER of 14.7%. Word lattices were converted to word-based, phonetic, character, pinyin, and toneless pinyin consensus networks. Keyword search was performed over the consensus networks with and without approximate match. We used a fixed probability of mismatch, Pm=0.0001 for all types of insertions, deletions and substitutions. This probability was set to such a low value to ensure that alignment errors do not compete with actual matched arcs in the consensus network. Since the approximate search has the flexibility of returning a very large number of results, the hits were always limited to the top 1000 from each system.

Table 2 shows keyword search accuracy for each system with and without approximate match search. As one can see, there appears to be small if any benefit from the approximate match search on the IV query terms. However, when we apply approximate match search to the OOV queries, we improve AUC and OTWV substantially. ATWV scores show mixed results which are not well correlated with OTWV, likely because the automatically computed threshold is very sensitive to the absolute value of confidence for individual hits as well as their aggregate confidence, which represents the expected count. The approximate

match search may introduce noise into the confidence computation that negatively affects the accuracy of the threshold.

For IV queries, the best individual KWS system is character-based, far superior to all other systems. Character-based search outperforms word-based search because in Mandarin performance is evaluated with the character sequence matching criterion, as described in Section 1. For OOV queries, phonetic system gives the best accuracy, with other systems that use approximate match following close behind. We also evaluated various combinations of search results, where search results from two or more systems were merged with their confidences linearly interpolated. The interpolation weights were set manually without tuning. Confidence weights for IV terms were set to 0.9 for the character-based system and 0.1 uniformly split among the other systems. For OOV terms we used uniform interpolation weights.

There appears to be minimal benefit to IV query terms from the system combination. In contrast, OOV detection accuracy improves substantially, with an AUC increase for the best combined result of 0.07 from the best single system, a more than 10% relative improvement. The best AUC and OTWV on OOV terms are obtained by combining all four sub-word systems, although using just the character and phonetic systems was nearly as good. ATWV scores for OOV terms showed some degradation (despite improvements in OTWV) with the combinations that involved the toneless pinyin system. This indicates that the confidence values produced by the toneless pinyin system are less accurate, thus negatively affecting the accuracy of the automatic confidence threshold.

In Figure 2 we compare AUC scores on OOV terms for the phonetic system with and without the approximate match search for various length queries. One can see that longer OOV queries (measured in phonemes) benefit more from the approximate match search. As the target phone sequence gets longer, the likelihood of the whole sequence being unaltered in the consensus network goes down. This effect is particularly strong for rare terms and OOVs where the language model will not favor these sequences, which leads to lower recall when we try to match these sequences exactly.

Improved recall is not the only benefit that we get from the approximate match search. Figure 3 offers another comparison between approximate and exact matches. Here, for each level of minimum precision we plot the total number of correct hits with at least that level of precision. At the left side of the figure, the higher value of the approximate match curve corresponds to that system's increased overall recall compared with exact match. The right side of the figure indicates that approximate match also generates about 10% more hits, even when considering only a high-precision operating point.

In Figure 4 we plot term-weighted DET curves that illustrate the improvements in keyword search accuracy on OOV terms from system combination. Here we merged sets of search results from character-based and phonetic consensus networks, averaging their confidences. These DET curves show the combination to be clearly superior to either one individual system.

6. CONCLUSIONS AND FUTURE WORK

In this paper we propose the use of an efficient approximate-match dynamic programming search in consensus networks. The algorithm, which is new to the task of keyword search, finds the best alignment between the target query and the consensus network, while allowing errors such as insertions, deletions and substitutions. Experiments show that the approximate-match search improves detection accuracy by more than 10% for OOV terms, leading to higher recall as well as a 10% increase in high-precision hits.

We measured keyword search accuracy in conversational Mandarin using different indexing units as well as different combinations of search results. We found that searching character consensus networks gives the best accuracy on in-vocabulary terms compared to phonetic, pinyin, toneless pinyin and word consensus networks. Combining search results from different systems was particularly helpful for words not in recognizer's dictionary (OOV terms) where detection accuracy was improved by around 10% relative to the best single system.

The probability of mismatch used by the DP algorithm was set to a constant value in our experiments. While this leads to a simple and language-independent solution, having unit-dependent probabilities for different types of errors will likely improve accuracy, particularly for longer units, such as syllables. In future work we plan to explore the use of a confusion matrix and/or other unit distance measures.

7. REFERENCES

[1] D. Miller, M. Kleber, C.-L. Kao, O. Kimball, T. Colthurst, S. Lowe, R. Schwartz and H. Gish, "Rapid and accurate spoken term detection", in Proceedings of InterSpeech, pp 314-317, 2007.

[2] F. Seide, P. Yu, C. Ma and E. Chang. "Vocabulary-Independent search in spontaneous speech", in Proceedings of ICASSP, 2004.

[3] P. Yu, K. Chen, C. Ma and F. Seide. "Vocabulary-independent indexing of spontaneous speech", IEEE Trans. on Speech and Audio Processing. Vol 12, No 5, 2005.

[4] K. Thambiratnam and S. Sridharan, "Rapid yet accurate speech indexing using dynamic match lattice spotting". IEEE Trans. on Speech and Audio Processing. Vol 15, No 1, 2007.

[5] L. Mangu, E. Brill and A. Stolcke, "Finding consensus in speech recognition: word error minimization and other applications of confusion networks", Computer Speech and Language, 14(4), 373-400, 2000.

[6] T. Hori, I. L. Hetherington, T. J. Hazen, and J. Glass, "Open-vocabulary spoken utterance retrieval using confusion networks", in Proceedings of ICASSP, 2007.

[7] T. Mertens, D. Schneider and J. Kohler, "Merging Search Spaces for Subword Spoken Term Detection", in Proceedings of InterSpeech, 2127-2130 ,2009.

[8] C. W. Han, S. J. Kang, C. M. Lee, and N. S. Kim, "Phone Mismatch Penalty Matrices for Two-Stage Keyword Spotting Via Multi-Pass Phone Recognizer", in Proceedings of InterSpeech, 202-205, 2010.

[9] J. Mamou, B. Ramabhadran, "Phonetic Query Expansion for Spoken Document Retrieval", in Proc. of InterSpeech, 2106-2109, 2008.

[10] I. Szőke, L. Burget, J. Černocký, M. Fapšo, "Sub-word modeling of out of vocabulary words in spoken term detection", In Proceedings of IEEE Workshop on Spoken Language Technology, India, 2008.

[11] J. Tejedor, D. Wang, S. King, J. Frankel and J. Colas, "A posterior probability-based system hybridization and Combination for Spoken Term Detection", in Proceedings of InterSpeech, 2009.

[12] S. Meng, P. Yu, F. Seide, J. Liu, "A study of lattice-based spoken term detection for Chinese spontaneous speech", ASRU, 2007.

[13] S. Meng, W.-Q. Zhang, J. Liu, "Combining Chinese Spoken Term Detection Systems via Side-information Conditioned Linear Logistic Regression", in Proceedings of InterSpeech, 685-689, 2010.

[14] S. Abdou, et al, "The 2004 BBN Levantine Arabic and Mandarin CTS Transcription Systems", in Proc. of EARS Workshop, 2004.

[15] R. Prasad, S. Tsakalidis, I. Bulyko, C. Kao, P. Natarajan, "Pashto Speech Recognition with Limited Pronunciation Lexicon", in Proceedings of ICASSP, 5086-5089, 2010.

[16] http://www.nist.gov/speech/tests/std/