FUSING MULTIPLE SYSTEMS INTO A COMPACT LATTICE INDEX FOR CHINESE SPOKEN TERM DETECTION

Sha Meng, Peng Yu, Jia Liu*, and Frank Seide*

*Department of Electronic Engineering, Tsinghua University, 100084 Beijing, P.R.C.
mengs04@mails.tsinghua.edu.cn, liuj@tsinghua.edu.cn
Microsoft Research Asia, 5F Beijing Sigma Center, 49 Zhichun Rd., 100080 Beijing, P.R.C.
{rogeryu,fseide}@microsoft.com

ABSTRACT

We examine the task of spoken term detection in Chinese spontaneous speech with a lattice-based approach. We first compare lattices generated with different units: word, character, tonal and toneless syllables, and also lattices converted from one unit to another unit. Then we combine lattices from multiple systems into a single lattice. By fully exploiting the redundant information in the combined lattice with a time-based node/arc merging, we achieve the result of a compact lattice index with the accuracy improved to 79.2% from 73.9% using the best subsystem.

Index Terms— spoken term detection, keyword spotting, lattice, system combination

1. INTRODUCTION

Improving accessibility for overwhelming amounts of speech data available today necessitates the development of robust Spoken Term Detection (STD, also known as keyword spotting) and Spoken Document Retrieval (SDR) techniques. However, speech data interesting for an STD or SDR task, including online lectures, conversations, and voicemails, are still a challenge for today's speech-recognition technology, which achieves word accuracies of only 50-70%. To deal with the high recognition error rate, a lattice-based approach has been widely used for SDR and STD tasks [1, 2, 3, 4, 5, 6], which was found to improve the search accuracy significantly.

For English systems, both word-based lattices and phonetic lattices have been used. Usually, word-based systems provide better precisions than phonetic systems due to stronger language models, but suffer from the Out-Of-Vocabulary (OOV) problem, which phonetic systems can handle nicely. Research [1, 3, 7] has also found combining the two systems results in significant improvements.

The (Mandarin) Chinese language has a monosyllabic structure, in which a closed set of syllables has satisfying vocabulary coverage, and syllable ngrams provide reasonably efficient language modeling. Most previous research [4, 6, 8] have involved building syllable-based systems. In [5], word-based lattices were converted and stored in a syllable-based index. [9] proposed multi-scale indexing, and showed an improvement by combining word-based transcriptions with syllable-based lattices.

Our research examines lattice-based spoken term detection for Chinese spontaneous speech. Specifically, we compare lattices generated with word and different sub-word units (character, tonal and toneless syllables), and also discuss methods to convert lattices generated with a higher-level unit, i.e., more semantic-based like word, to a lower-level unit, i.e. more phonetic-based like syllable. Some of the results have already been presented in [10], while in this paper, we fuse multiple systems by a lattice-level combination followed by a time-based node/arc merging, and result in a compact lattice index with a significantly better accuracy than all subsystems.

The rest of paper is organized as follows. Section 2 introduces the lattice-based spoken term detection algorithm. Section 3 describes the methods of lattice generation and conversion. Section 4 discusses how to merge multiple lattices into a compact lattice index. Section 5 shows the results and section 6 concludes the paper.

2. LATTICE-BASED SPOKEN TERM DETECTION

We first recapitulate the method of lattice-based spoken term detection. A lattice $\mathcal{L} = (\mathcal{N}, \mathcal{A}, n_{\text{start}}, n_{\text{end}})$ is a directed acyclic graph (DAG) with $\mathcal{N}$ being the set of nodes, $\mathcal{A}$ being the set of arcs, and n_{start}, $n_{\text{end}} \in \mathcal{N}$ being the unique initial and unique final node, respectively.

Each node $n \in \mathcal{N}$ has an associated time $t[n]$ and possibly an acoustic or Language-Model (LM) context condition. Arcs are 4-tuples $a = (S[a], E[a], I[a], w[a])$. $S[a]$, $E[a] \in \mathcal{N}$ denote the start and end node of the arc. $I[a]$ is the word (or sub-word) identity. Last, $w[a]$ shall be a weight assigned to the arc by the recognizer. Specifically, $w[a] = p_{\text{ac}}(a)^{1/\lambda} \cdot P_{\text{LM}}(a)$ with acoustic likelihood $p_{\text{ac}}(a)$, LM probability P_{LM}, and LM weight λ. [1] Normally the recognizer will also tell the

[1] Despite its name, the function of the LM weight is now widely consid-

best pronunciation for each arc, when multiple pronunciations exist for the word $I[a]$.

In addition, we define a *path* $\pi = (a_1, \cdots, a_K)$ as a *sequence* of connected arcs, and use the symbols S, E, I, and w for paths as well to represent the respective properties for entire paths, i.e. the path start node $S[\pi] = S[a_1]$, end node $E[\pi] = E[a_K]$, label sequence $I[\pi] = (I[a_1], \cdots, I[a_K])$, and total path weight $w[\pi] = \prod_{k=1}^{K} w[a_k]$.

It was found in [2] that an alternative but equivalent representation of lattices, which we call *posterior lattices*, is more convenient in many cases.

For the posterior lattice, we define the *arc posteriors* $P_{\text{arc}}[a]$ and *node posteriors* $P_{\text{node}}[n]$ [2] as

$$P_{\text{arc}}[a] = \frac{\alpha_{S[a]} \cdot w[a] \cdot \beta_{E[a]}}{\alpha_{n_{\text{end}}}} \;;\; P_{\text{node}}[n] = \frac{\alpha_n \cdot \beta_n}{\alpha_{n_{\text{end}}}},$$

with *forward-backward probabilities* α_n, β_n defined as:

$$\alpha_n = \sum_{\pi: S[\pi]=n_{\text{start}} \wedge E[\pi]=n} w[\pi] \quad ; \quad \beta_n = \sum_{\pi: S[\pi]=n \wedge E[\pi]=n_{\text{end}}} w[\pi]$$

α_n and β_n can be conveniently computed using the well-known forward-backward recursion, e.g. [11].

The posterior lattice representation stores four fields with each edge: $S[a]$, $E[a]$, $I[a]$, and $P_{\text{arc}}[a]$, and two fields with each node: $t[n]$, and $P_{\text{node}}[a]$.

In our previous work [12] it was shown that in a word spotting task, ranking by the phrase posterior probability is theoretically optimal. With the posterior lattice representation, the phrase posterior of query string Q is computed as

$$P(*, t_s, Q, t_e, *|O) = \sum_{\substack{\pi=(a_1,\cdots,a_K): \\ t[S[\pi]]=t_s \wedge t[E[\pi]]=t_e \wedge I[\pi]=Q}} \frac{P_{\text{arc}}[a_1] \cdots P_{\text{arc}}[a_K]}{P_{\text{node}}[S[a_2]] \cdots P_{\text{node}}[S[a_K]]}. \quad (1)$$

The posterior lattice representation is lossless. It has several advantages as stated in [2]. In this paper, the posterior lattice representation is needed by the lattice conversion and system combination, which we will discuss later.

3. LATTICE GENERATION AND CONVERSION

In this section, we briefly describe how to generate lattices with word and sub-word units, and how to convert lattices to a different unit. A detailed description can be found in [10].

We use a same Large-Vocabulary Continuous Speech Recognition (LVCSR) decoder to generate lattices with different base units: word, character, tonal and toneless syllable. Trigram language models trained with different units are used for corresponding recognizers. Result lattices are represented as posterior lattices as described in section 2.

ered to flatten acoustic emission probabilities. This matters when sums of path probabilities are taken instead of just determining the best path.

[2] In [1], a similar concept is implemented by weight pushing.

Converting word lattices to sub-word lattices (character or syllable) involves an arc splitting process, as a word may be split to multiple characters (or syllables). The process is trivial with posterior lattices: replacing the word arc with a sequence of connected sub-word arcs, with all the sub-word arcs having the same arc posterior as the word arc, and equally splitting the time period. Converting character lattices to syllable lattices or tonal-syllable lattices to toneless-syllable lattices is a simple label replacing operation on lattices.

Sometimes words and characters have multiple pronunciations (polyphonies), thus the conversion to syllables is not unique. In this case, the best pronunciation info for each arc provided by the recognizer is used to determine the corresponding syllables.

4. SYSTEM COMBINATION IN A COMPACT LATTICE INDEX

[3] has shown that lattices generated with different recognizers contain complementary information and can be fused to achieve better accuracy over each subsystem. In this section, we will present a method to combine multiple lattices for better accuracy, while the combined system can still be represented by a single compact lattice.

4.1. Lattice Combination

Let $\mathrm{L}_1, \cdots, \mathrm{L}_n$ denote lattices generated with different subsystems, Q being the query. [10] proposed to combine multiple systems by a *posterior combination*:

$$P^{\text{COMB}}(*, t_s, Q, t_e, *|O) = \sum_{i=1}^{n} \gamma_i \cdot P(*, t_s, Q, t_e, *|\mathrm{L}_i).$$

An equivalent representation of the above posterior combination is the *lattice combination*, which combine all sub-lattices L_i into a single lattice by merging start nodes and end nodes of all L_i, and weighting arcs/nodes from L_i by γ_i. However, though in the form of a single lattice, the representation of the combined lattice is not really efficient as we still need to store all arcs/nodes from all sub-lattices. There are two-levels of redundant information in the combined lattice:

- **within-lattice:** In raw lattices generated by speech decoders, a single word (or sub-word) often has multiple lattice arcs with different acoustic or LM context conditions and slightly off time boundaries, this duplication was found to be unnecessary for an STD task;
- **cross-lattice:** Lattices with the same base unit but from different recognizers contain more or less similar recognition alternatives in the same time period, which becomes duplicated in the combined lattice.

To exploit these redundancies, we propose use of lattice compression method presented in [2]: Time-based Merging for Index (TMI), which we will explain below.

4.2. Time-based Merging for Index (TMI)

The basic idea of TMI is to cluster lattice nodes with similar times, and to approximate word hypotheses by arcs between node clusters rather than individual nodes. The clustering criterion is simple: two consecutive nodes can be clustered together *unless* that would create a loop (a hypothesis starting and ending in the same clustered node); and among the manifold of clusterings that satisfy this condition, the one leading to the smallest number of clusters is considered the optimum solution. It can be found using dynamic programming:

- sort nodes $n_1...n_N$ in ascending time
- for each node n_i, determine m_i: the maximum node that n_i can be grouped with without causing a loop
- set cluster counts $C_0 \leftarrow 1$; $C_i \leftarrow \infty \, \forall \, i > 0$
- set backpointers $B_i \leftarrow n_i \, \forall \, i > 0$
- for $i = 1...N$: // DP recursion
  - for $j = i...m_i$: if $C_{i-1} + 1 \leq C_j$:
    - $C_j \leftarrow C_{i-1} + 1$
    - $B_j \leftarrow i$ // cluster $\{n_i...n_j\}$
- $k \leftarrow N$; while $k \neq 0$: // trace back, merge nodes
  - create new node cluster $\{B_k...n_k\}$, relink arcs
  - $k \leftarrow B_k - 1$
- merge arcs that connect the same two clusters with same word by summing up their posteriors

We call this "Time-based Merging for Indexing," as it effectively clusters nodes with similar time points, although node times are only used for node sorting. The process keeps all phrases. It also introduces additional paths, but they are restricted to not cause insertions or deletions of full words, and our experiments show no loss of accuracy from false positives (these random combinations are unlikely to be valid phrases for which one could search).

When TMI is applied to the combined lattice in the previous section, it will not only merge nodes/arcs within sub-lattices, but also merges across sub-lattices, i.e. both within-lattice and cross-lattice redundancies are addressed.

5. EXPERIMENTAL RESULTS

5.1. Setup

We evaluate our method by a keyword-spotting task on a 4-hour long Chinese spontaneous dictation set. The phone set contains 187 phones, with 28 "initial" (the Consonant) phones, 157 tonal "final" (the Vowel) phones, and two silence phones. There are a total of 1,666 tonal syllable and 423 toneless syllables [13]. An acoustic model trained on 154-hour reading-style speech plus 148-hour spontaneous speech is used for all setups. 39-dimension MFCCs are used. A dictionary with 68,933 words is used for both the LVCSR recognizer and for the word breaker (which is used for query processing). Trigrams with different units are all trained from a text corpus containing about 2.1 billion characters.[3]

The baseline Word Error Rate (WER) of all recognizers is shown in Table 1. To compare among different units, Character Error Rate (CER), Syllable Error Rate (SER) and toneless SER are listed as well. As expected, the word-based system has the best error rates. Interestingly, the character-based system is even worse than the tonal-syllable-based one. This may be caused by polyphonies – characters with multiple pronunciations. Normally a specific pronunciation of a character corresponds to a specific meaning, which means the character language model may not be as efficient as the syllable language models.

Table 1. Accuracy of different recognizers (WER:word error rate, CER: character error rate, SER: syllable error rate, all in %)

recognizer unit	WER	CER	SER	Toneless SER
Word	48.43	36.98	35.38	30.81
Character	–	42.90	41.33	35.90
Syllable	–	–	39.08	33.64
Toneless Syllable	–	–	–	35.83

An automatic procedure as described in [14] was used to select queries. Example queries are 春节 (spring festival), 俄罗斯 (Russia), 埃菲尔铁塔 (Eiffel Tower). Results are reported in Figure of Merit (FOM), which is defined by National Institute of Science and Technology (NIST) as the detection/false-alarm curve averaged over the range of [0...10] false alarms per hour per keyword. Lattice recalls (recalls of all query matches, which are upper bounds of FOMs) are listed as well for the purpose of analysis.

5.2. Lattice Generation and Conversion

The first two lines of Table 2 compare the bestpath-only (S0) approach versus a lattice-based (S1) approach for a word-based system. With the high WER (48.4%), the bestpath has a recall at only 51.5%, while using lattices increases the recall to 71.2%, with the FOM improved to 69.2%.

Lines tagged with S2, S3, S4 list performances of lattices generated with characters, tonal syllables and toneless syllables.[4] Tonal-syllable-based lattices show the best FOM of 72.3% and recall of 76.0%, which indicates that tonal syllables are an efficient compromise between the language model strength and the vocabulary coverage.

Table 2 also lists results for lattice conversion as tagged as Sx.y. The results show that converting lattices from a higher-

[3] In [10], syllable trigrams were trained only on the dictionary. In this paper, they are retrained on the LM training corpus, which achieves better WERs and FOMs.

[4] The lattice recall is primarily a function of the beam setting in decoding. However, as different systems have different perplexities, it is difficult to fairly compare beam settings across systems. We tune each recognizer for the maximum recall, so the recall here could be understood as "the upper limit of recall with a realistic setup" for each system.

Table 2. Keyword spotting results. The index size is measured as lattice arcs (index entries) per spoken character. All numbers are in %.

id.	index	FOM	Rec	Size
S0	word-bestpath	50.7	51.5	0.7
S1	word	69.2	71.2	82.7
S1.1	=>character	71.1	73.2	99.9
S1.2	=>syllable	72.3	75.2	99.5
S1.3	=>toneless syl.	73.7	77.6	99.5
S2	character	67.6	70.4	756.3
S2.1	=>syllable	69.8	73.5	759.5
S2.2	=>toneless syl.	72.0	77.1	759.5
S3	syllable	72.3	76.0	110.4
S3.1	=>toneless syl.	73.6	78.4	110.4
S4	toneless syl.	68.8	72.9	217.1
combining S1.3+S2.2+S3.1+S4				
C1	lattice combination	78.6	84.6	1204.4
C2	+ within-lattice TMI	79.6	86.1	32.0
C3	+ cross-lattice TMI	79.2	86.4	19.3

level unit to a lower-level one always provides better FOMs and recalls. All the recognizers achieve the best performance when lattices are converted to toneless-syllables. After conversion, toneless-syllable lattices from the word-based recognizer (S1.3) show the best FOM of 73.7%.

5.3. System Combination

In the next experiment, we combine the converted toneless-syllable lattices from four recognizers. This is not only because toneless-syllable lattices provide the best performance from each recognizer, but also because they enable cross-lattice merging by using the same base unit. Equal weights are used for combination.

The first line (C1) shows the result of the direct lattice combination without TMI compression, which has a significantly better FOM of 78.6% as compared with the best subsystem 73.7%, with an index size of 1,204.4. The next line (C2) shows the effect of removing within-lattice redundancy, which is done by applying TMI to each sub-lattice before lattice combination. A 37 times size reduction (from 1204.4 to 32) is observed. Interestingly, TMI also slightly improves the accuracy, by introducing additional paths to raw lattices. In the last line (C3), TMI is applied to the merged lattice, thus both within-lattice and cross-lattice redundancies are addressed. A further size reduction from 32.0 to 19.3 is achieved.

6. CONCLUSION

We examined the Chinese Spoken Term Detection Task by a lattice-based approach. We compared lattices generated with different units: word, character, tonal and toneless syllables, and lattices converted from a higher-level unit to a lower-level one. Our experimental results have shown that the best performance is with converted toneless-syllable lattices from a word-based recognizer.

We then combined multiple systems by a lattice-level integration followed by a time-based arc/node merging. By fully exploiting the redundant information in the combined system, we resulted in a compact lattice index with 19.3 arcs per spoken character, with the FOM improved to 79.2% from the best subsystem at 73.7%.

7. ACKNOWLEDGMENTS

The authors wish to thank our colleague Dr. Yu Shi for sharing her setup for the Chinese spontaneous dictation set.

8. REFERENCES

[1] M. Saraclar, R. Sproat, Lattice-based Search for Spoken Utterance, *Proc. HLT'2004*, Boston, 2004

[2] Z. Y. Zhou, P. Yu, C. Chelba, F. Seide, Towards Spoken-document Retrieval for the Internet: Lattice Indexing For Large-Scale Web-search Architectures, *Proc. HLT'2006*, New York, 2006.

[3] P. Yu, Frank Seide, A Hybrid Word/Phoneme-based Approach for Improved Vocabulary-independent Search in Spontaneous Speech, *Proc. ICLSP'2004*, Korean, 2004.

[4] H. M. Wang, Experiments in syllable-based retrieval of broadcast news speech in Mandarin Chinese, Speech Communication 32 (2000), pp. 49-60.

[5] J. Shao, Q. W. Zhao, P. Y. Zhang, Z. J. Liu, Y. H. Yan, A Fast Fuzzy Keyword Spotting Alogrithm Based on Syllable Confusion Network, *Proc. Interspeech'2007*, Antwerp, 2007

[6] Y. C. Pan, H. L. Chang, B. Chen, L. S. Lee, Subword-based Position Specific Posterior Lattice(S-PSPL) for Indexing Speech Information, *Proc. Interspeech'2007*, Antwerp, 2007

[7] S. W. Lee, K. Tanaka, Y. Itoh, Combining Multiple Subword Representations for Open-vocabulary Spoken Document retrieval,*Proc. ICASSP'2005*, Philadelphia, 2005.

[8] B. Chen, H. M. Wang, L. S. Lee, Retrieval of Broadcast News Speech in Mandarin Chinese Collected in Taiwan Using Syllable-level Statistical Characters, *Proc. ICASSP'2000*, Istanbul, 2000.

[9] H. M. Wang, H. Meng, P. Schone, B. Chen, W. K. Lo, Multi-Scale Audio Indexing for Translingual Spoken Document Retrieval, *Proc. ICASSP'2001*, , 2001.

[10] S. Meng, P. Yu, F. Seide, J. Liu, A Study of Lattice-Based Spoken Term Detection for Chinese Spontaneous Speech, *to appear in Proc. ASRU'2007*, Kyoto, 2007.

[11] F. Wessel, R. Schluter, K. Macherey, and H. Ney, Confidence Measures for Large Vocabulary Continuous Speech Recognition, IEEE transaction on Speech and Audio Processing, Vol.9, No.3, Mar.2001.

[12] P. Yu, K. J. Chen, C. Y. Ma, F. Seide, Vocabulary-independent Indexing of Spontaneous Speech, IEEE transaction on Speech and Audio Processing, Vol.13, No.5, Special Issue on Data Mining of Speech, Audio and Dialog.

[13] C. Huang, Y. Shi, J. L. Zhou, M. Chu, T. Wang, E. Chang, Segmental Tonal Modeling for Phone Set Design in Mandarin LVCSR, *Proc. ICASSP'2004*, Montreal, 2004.

[14] F. Seide, P. Yu, *et al.*, Vocabulary-independent Search in Spontaneous Speech, *Proc. ICASSP'2004*, Montreal, 2004.