ORDER-FREE SPOKEN TERM DETECTION

Lidia Mangu, George Saon, Michael Picheny and Brian Kingsbury

IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA

ABSTRACT

In this paper, we propose Time-Marked Word (TMW) lists as a replacement for the lattices and Confusion Networks (CNs) widely used as indexing vehicles for Spoken Term Detection (STD). In a TMW list, candidates are simply tagged with posterior probabilities and time information and stored as a large list of words: the additional ordering present in a lattice or CN is discarded. TMW lists compactly summarize a large ASR search space. Representing a large search space is critical for STD metrics such as ATWV that heavily penalize misses of rare keywords. Comparisons on the OpenKWS 2014 Tamil limited language pack task [1] show that the new TMW-based indexing results in better performance while being faster and having a smaller footprint.

Index Terms— keyword search, spoken term detection, keyword spotting, audio indexing, confusion networks

1. INTRODUCTION

Finding a target term in an audio corpus is one of the fundamental problems in automatic speech processing. Given the vast amount of existing spoken information, there is an increasing need for small indices and fast search. Typically, state-of-the-art spoken term detection (STD) systems search for terms in an index built from the output of an automatic speech recognition (ASR) system. The ASR output representation is the 1-best hypothesis, and using it for indexing results in good STD performance if the ASR system has low word error rate. However, most state-of-the-art STD systems, which often have to deal with degraded inputs, benefit from using a richer ASR output representation. Lattices and confusion networks (CNs) [2] are two commonly used representations of multiple hypotheses from an ASR system, and are frequently used for building STD indices [3, 4, 5, 6, 7, 8, 9, 10, 11]. The drawback of the lattice approach is the large disk space needed to store the index. Confusion networks are much smaller, but the CN computation can be prohibitive for large lattices. We propose a replacement for lattices and confusion networks, - time-marked word (TMW) lists. TMWs are a set of words with start and end times and posterior scores. Unlike lattices and CNs, which explicitly represent word ordering in their topologies, TMWs lack such structure, encoding word ordering implicitly in the time marks. The relationship between lattices or CNs and TMWs is analogous to the relationship between a sentence and its bag-of-words representation. To accommodate the lack of explicit word-order information in the TMW lists, we propose a new Weighted Finite State Transducer (WFST) architecture for STD.

The organization of this paper is as follows: Section 2 describes the TMW lists and compares them with the standard representations, lattices and CNs. An overview of the task, metric, and ASR system used for indexing is given in Section 3. Section 4 and Section 5 describe the indexing and search in the proposed architecture. Section 6 shows our experiments and results and we conclude in Section 7.

2. ASR OUTPUT REPRESENTATIONS

Most speech recognition systems produce lattices or confusion networks to be used for STD indexing. Lattices are partially ordered networks of word hypotheses, with links in the networks carrying word identity, time information, language model (LM) and acoustic model (AM) scores. Posterior probabilities for the links in a lattice can be computed from the LM and AM scores using the Forward-Backward algorithm. Confusion networks have a linear structure, representing the competing word hypotheses and their posterior probabilities in consecutive time intervals (*confusion bins*). CNs are produced from lattices through a 2-step process: (1) *Intra-word* clustering, in which the lattice arcs which have the same word label, start and end time are merged and their posteriors summed up, and (2) *Inter-word* clustering, in which all the lattice arcs are clustered until the partial order becomes a total order, leading to the linear structure. CNs are orders of magnitude smaller than lattices, but they take extra time to compute. The *inter-word* clustering step accounts for 99% of the computation time. To avoid this time-consuming step, we propose the time-marked word (TMW) list, which is the output of the *Intra-word* clustering step: an enumeration of word labels, start and end times, and posterior probabilities, (w, s, e, p). Silence, hesitations and other filler words are not written into this list. A lattice is computed in memory, but only the TMW list is produced on disk. If we want to reduce the size of the TMW lists further, we can relax the exact time match constraint to allow for arcs with large overlap to merge as well. In this paper we report results for exact match only.

3. DATA AND ASR SYSTEM DESCRIPTION

We conducted our experiments in the context of the IARPA Babel program [12], which focuses on spoken term detection for low-resource languages. The STD task is defined by NIST in the OpenKWS14 Evaluation Plan [1]. We chose the limited language pack track (LP) of the program, in which only 20 hours of audio, (10 hours of transcribed data) is used for building ASR models and lexicons, making it more interesting for out-of-vocabulary (OOV) keyword search. In this paper, we focus on the Tamil language, which was the OpenKWS 2014 evaluation task. The limited language pack includes a 20-hour development set (DEV). For these experiments we used two keyword sets: IBM-1, containing 1721 in-vocabulary (IV) queries and 654 OOV queries, and IBM-2, containing 1978 IV queries and 617 OOV queries, generated by IBM [13] and supplied to all OpenKWS participants.

The metric used for the Babel program is Term-Weighted Value (TWV), which was first used in the NIST 2006 STD Evaluation [14]. We report keyword search performance in terms of maximum Term-Weighted Value (MTWV) which is the best TWV for all values of the decision threshold. We also report Optimal TWV (OTWV) which gives an upper-bound of the performance under perfect keyword-specific thresholding and Supremum TWV (STWV) which gives an upper bound of the performance assuming perfect detection scores and thresholding [1].

The acoustic model used in these experiments is a collection of three DNNs which differ in the number of output states (1000, 2000, 3000). The DNNs take 9 consecutive frames as input where each frame is a concatenation of a 40-dimensional FMLLR vector [15] and a 7-dimensional fundamental frequency variation (FFV) vector [16, 17]. Each DNN has 5 hidden layers with 1024 sigmoid units. During decoding, the output scores of the DNNs are combined at the frame level with equal weights. The training of the nets comprises (1) layer-wise discriminative pre-training using the cross-entropy criterion, (2) stochastic gradient training using back-propagation and the cross-entropy criterion, and (3) sequence discriminative training using stochastic gradient and the state-level minimum Bayes risk criterion [18]. The dictionary has 14.1K words and 21.3K pronunciations. The language model (LM) is a trigram LM with modified Kneser-Ney smoothing trained only on the acoustic transcripts.

The lattices, CNs and TMW lists are produced using a dynamic decoder [19]. The word error rates for the 1-best hypotheses from the lattices and confusion networks are 73.9% and 73.1%, respectively. For simplicity, we present results for this acoustic model only, which was the IBM model with the best ATWV performance in the OpenKWS14 evaluation. Similar improvements are obtained for other acoustic models.

4. INDEXING

In this section we describe the order-free method we propose for indexing TMW lists. An index containing all the information needed for keyword search (audio file identity, start time, end time, and word label) is constructed from a TMW list using the following steps.

1. For each utterance, i, we create a word loop WFST, which has S_i as the start node, E_i as the end node, and arcs from S_i to E_i for each item (w, s, e, p) in the TWM list. These arcs have w as the input label, (s,e) as the output label and $-log(p)$ as the cost. E_i is connected to S_i by a zero-cost epsilon arc, thus creating a word loop.

2. The final single index is obtained by creating a new start node, S, that is connected to each S_i by zero-cost arcs with input label epsilon and output label i (or audio file id), and a new end node, E, that is connected to each E_i by zero-cost epsilon-arcs.

Figure 1 shows the TMW-based index. The set of keywords that can be retrieved by this index is larger than the one that can be retrieved by a lattice index due to the full connectivity. A multi-word keyword might not be found in a lattice index if there is no path connecting the word components in the lattice. This can be a problem especially for large keywords. In the case of a CN-based index, which is already a much more connected structure, the TMW-based index allows for new sequences of words which might be missed in a CN due to an *Inter-word* alignment error.

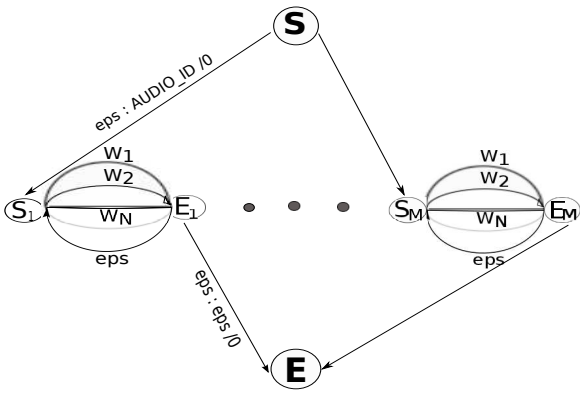

Fig. 1. TWM-based index.

Note that even though our intention is to have the ASR system output TMW lists instead of lattices and CNs, in case those alternate outputs exist, they can be easily converted to TMW lists and be indexed in a similar fashion. We will refer to those as Lattice-TMW and CN-TMW indexing in Section 6.

5. SEARCH

Each query is converted into a word automaton to search the index described in Section 4. In-vocabulary (IV) query automata are directly composed with the word index transducer. For OOV search, either (1) queries are converted to IV queries (proxies) using a phone confusability (P2P) transducer [20, 21, 22] and then composed with the word index, or (2) the index is converted to phone level by replacing all words with their pronunciations and is then searched via composition with phone automata. A phone automaton is generated by (1) converting an OOV word automaton to a phone automaton P using the lexicon, (2) composing P with P2P, and (3) extracting N-best paths. Both methods produce identical results, with the better choice depending on memory and computational constraints, as well as on the size of the vocabulary. The advantage of the proxy method comes from a smaller index size and faster search. But, for large vocabulary sizes, the conversion of OOV queries to IV proxies is computationally and memory intensive, in which case the phonetic method is preferred. Note that for many tasks the IV search can also benefit from expansion using a P2P transducer, in which case the indexing and search pipeline for all the queries will be the same, and only the degree of phonetic expansion (N-best) will differ (less expansion for IV queries).

Regardless of the type of composition, word-based or phone-based, the result of the composition, after projecting on the output label, is a list of hits for each query and the corresponding score. A hit contains the audio file id, as well as a sequence of start and end time pairs (s_i, e_i) corresponding to the word components of a multi-word query "audio file id" (s_1, e_1) (s_2, e_2) ... (s_n, e_n). In contrast to the previous lattice and CN-based WFST approaches in which the start and end time pairs were ordered due to the structure of the index, when employing TMW lists we use one more step: all the hits containing consecutive time pairs that are not *ordered* are eliminated. Two time pairs (s_i, e_i) and (s_{i+1}, e_{i+1}) are ordered if $s_i < s_{i+1}$ and $s_{i+1} - e_i < thresh$, where $thresh$ is empirically determined. In other words, the start times have to be sorted in time, and the putative locations of the word components should not be far from each other. Note that $s_{i+1} - e_i$ could be negative if the two time pairs overlap. The final posting list consists of the surviving hits, which have start time s_1 and end time e_n. In case there are two overlapping hits for a keyword, we keep only the hit with the maximum score. For each keyword, the scores below a threshold are normalized as in [20], while high scores are kept intact.

6. EXPERIMENTS AND RESULTS

The OpenFST Library [23] is used for both indexing and search. There are many methods [24, 25, 26] for creating the phone confusability transducer. For the OpenKWS evaluation we used a simple method that compares the Viterbi alignment of the training data transcripts to the decoded output to accumulate state-level confusions which are then converted to phone-level confusions.

System	MTWV	OTWV	STWV
Lattice STD	0.1503	0.2723	0.4625
CN STD	0.1518	0.2810	0.4912
TMW STD	0.1549	0.2883	0.5116

Table 1. Comparison of STD performance.

System	Index Size	Time to produce
Lattice	21G	82 hours
CN	110M	124 hours
TMW	295M	80 hours

Table 2. Comparison of size and computational times.

As a baseline for the TMW based STD we use the state-of-the-art lattice and CN WFST STD architectures we successfully deployed in both the DARPA RATS and IARPA Babel evaluations [4, 24, 25, 27, 20]. In the lattice architecture a word index built from lattices [4] is used for IV search and a phone index is used for OOV search, after the OOV queries are expanded using the P2P transducer. In the CN approach, a word index built from CNs [20] is used for both IV and OOV search. All queries are mapped to IV proxies after expansion with the P2P transducer. THe same confusability transducer is used for all approaches, and the same degree of expansion for IV (N-best=2000) and OOV queries (N-best=20000). Table 1 and Table 2 show the performance, index size and computational time for TMW lists, CNs, and lattices produced by the acoustic model described in Section 3. It can be seen that TMW STD has the best MTWV, OTWV, and STWV, requires the least amount of time for index generation, and produces a smaller index than lattice STD. While CN STD has an even smaller index size, if we increase decoding beams for CN STD to match the TMW STD index size, the CN STD performance is still worse (MTWV=0.1525) and the time to produce the CN STD index increased by 20%.

We also investigate the difference between order-free indexing and structured indexing for a given ASR output type. Order-free indexing based on lattices (lattice-TMW) is simply a matter of converting lattices to TMW lists and then applying TMW indexing and search. This is identical to TMW STD, except that the lattices have been written to disk. For order-free indexing based on CNs (CN-TMW), we create TMW lists by extracting the words with time information and their posterior probabilities from CNs, and then apply TMW indexing and search. The comparison between lattice STD and lattice-TMW STD is made in Table 1, while the comparison between CN STD and CN-TMW STD is made in Table 3. Even if we use CNs as an intermediate representation, order-free indexing improves STD performance.

System	MTWV	OTWV	STWV
CN STD	0.1518	0.2810	0.4912
CN-TMW STD	0.1525	0.2993	0.5001

Table 3. Comparison of CN indexing methods

System	MTWV	Index Size
Lattice STD	0.1553	149G
TMW STD	0.1602	1G

Table 4. Comparison for larger ASR decoding beams

The STD results above are obtained using the same ASR decoding parameters, namely the ones used in the evaluation. For the ATWV metric it is very important that rare words are not missed; therefore, better performance can be achieved if the index is rich enough to contain instances of those words, even if the scores are low. If the only hit for a word has a very low score, after normalization this score becomes 1, and will survive any thresholding. Given that TMW lists are much smaller than lattices and faster to produce than CNs, we can afford to increase the decoding beams and thus prune fewer hypotheses. As seen in Table 4, with an index that is 150 times smaller, we obtained better performance.

In all the above experiments, indexing is based on word ASR decoding; however, our best evaluation system used three indexes: (1) word-based, (2) word-based but with no language model scores, and (3) morph-based. For each query we search in the three indexes simultaneously and merge the results. Fig 2 shows the architecture of an index that can be used for this parallel search. The labels T_1,T_2,T_3 identify the sub-index that produces a given hit in the resulting posting list. These identifiers are needed due to the different merging strategies used in case of overlapping hits: for hits coming from the same sub-index we keep only the maximum scoring one, while for hits coming from different sub-indexes we add up the scores. As seen in Table 5, parallel indexing and search results in 40% relative improvement in ATWV, and this improvement holds when TMW STD is used instead of CN STD. We are comparing only against CN STD because this was the system that was submitted in the OpenKWS evaluation. TMW STD is especially beneficial for parallel indexing and search. Given the complex structure of the parallel index, it is important to have small sub-indexes which can also be produced quickly.

7. CONCLUSION AND FUTURE WORK

We propose time-marked word (TMW) lists as a replacement for lattices and CNs, as input for STD indexing. TMW lists are much smaller than lattices, and faster to compute than CNs. To accomodate the lack of explicit word-order information in the TMW lists, we propose a new word-loop FST architecture for STD. The burden of insuring that the words in a multi-word query are correctly ordered in an STD hit is transferred from the indexing step to the search step. While previously the index encoded this information, causing the index to be large (lattices) or slower to produce (CNs), the current approach simply imposes an efficient time order test during search. We also show that the proposed STD architecture can be applied to lattices and CNs by converting them to TMW lists. In this work we create TMW lists after creating a lattice in memory, which allows us to compute word posterior probabilities. As future work, we are investigating faster methods for computing reliable scores for the time-marked words to be used for STD.

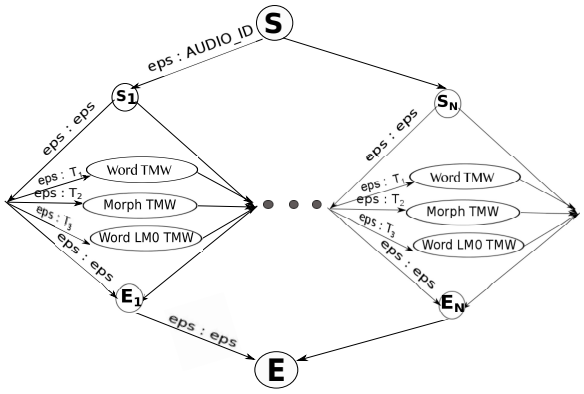

Fig. 2. Parallel index.

System	MTWV
CN STD	0.2194
TMW STD	0.2210

Table 5. Comparison of parallel STD architecture for CN STD and TMW STD

Acknowledgement

This work is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Laboratory (DoD/ARL) contract number W911NF-12-C-0012. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government. 2/ This effort uses the IARPA Babel Program language collection IARPA-babel204b-v1.1b Tamil limited language pack.

8. REFERENCES

[1] NIST, "The spoken term detection (STD) 2014 evaluation plan.," in *http://www.nist.gov/itl/iad/mig/openkws2014.cfm*, 2014.

[2] 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*, vol. 14, no. 4, pp. 373–400, 2000.

[3] C. Chelba, T.J. Hazen, and M. Saraçlar, "Retrieval and browsing of spoken content," *IEEE Signal Processing Magazine*, vol. 25, no. 3, pp. 39–49, 2008.

[4] D. Can and M. Saraclar, "Lattice indexing for spoken term detection," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 8, pp. 2338–2347, 2011.

[5] M. Saraclar and R. Sproat, "Lattice-based search for spoken utterance retrieval," in *Proc. HLT-NAACL*, 2004, pp. 129–136.

[6] C. Parada, A. Sethy, and B. Ramabhadran, "Query-by-example spoken term detection for OOV terms," in *Proc. ASRU*, 2009, pp. 404–409.

[7] P. Yu and F. Seide, "A hybrid word/phoneme-based approach for improved vocabulary-independent search in spontaneous speech," in *Proc. Interspeech*, 2004.

[8] T.J. Hazen T. Hori, I.L. Hetherington and J.R. Glass, "Open-vocabulary spoken utterance retrieval using confusion networks," in *Proc. ICASSP*, 2007.

[9] V. Turunen and M. Kurimo, "Indexing confusion networks for morph based spoken document retrieval," in *Proc. SIGIR*, 2007.

[10] K. Vertanen, "Combining open vocabulary recognition and word confusion networks," in *Proc. ICASSP*, 2008.

[11] S. Nakagawa, K. Iwami, Y. Fujii, and K. Yamamoto, "A robust/fast spoken term detection method based on a syllable n-gram index with a distance metric," 2013, vol. 55, pp. 470–485.

[12] "IARPA broad agency announcement IARPA-BAA-11-02," 2011.

[13] J.Cui, J.Mamou, B.Kingsbury, and B.Ramabhadran, "Automatic keyword selection for keyword search development and tuning," in *Proc. ICASSP*, 2014.

[14] NIST, "The spoken term detection (STD) 2006 evaluation plan.," in *http://www.nist.gov/speech/tests/std/*, 2006.

[15] H. Soltau, G. Saon, and B. Kingsbury, "The IBM Attila speech recognition toolkit," in *Proc. SLT*, 2010, pp. 97–102.

[16] K. Laskowski, M. Heldner, and J. Edlund, "The fundamental frequency variation spectrum," in *Proc. 21st Swedish Phonetics Conference (Fonetik 2008)*, 2008.

[17] F. Metze, Z. A. W. Sheikh, A. Waibel, J. Gehring, K. Kilgour, Q. B. Nguyen, and V. H. Nguyen, "Models of tone for tonal and non-tonal languages," in *Proc. ASRU*, 2013.

[18] B. Kingsbury, "Lattice-based optimization of sequence classification criteria for neural- network acoustic modeling," in *Proc. of ICASSP*, 2009.

[19] H. Soltau and G. Saon, "Dynamic network decoding revisited," in *Proc. ASRU*, 2009.

[20] L.Mangu, B.Kingsbury, H. Soltau, H.-K. Kuo, and M. Picheny, "Efficient spoken term detection using confusion networks," in *Proc. ICASSP*, 2014.

[21] M.Saraclar, A.Sethy, B.Ramabhadran, L.Mangu, J.Cui, X.Cui, B.Kingsbury, and J.Mamou, "An empirical study of confusion modeling in keyword search for low resource languages," in *Proc. of ASRU*, 2013.

[22] G.Chen, O.Yilmaz, J.Trmal, D.Povey, and S.Khundapur, "Using proxies for oov keywords in the keyword search task," in *Proc. of ASRU*, 2013.

[23] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri, "OpenFST: A general and efficient weighted finite-state transducer library," in *Proc. CIAA*, 2007, pp. 11–23.

[24] L. Mangu, H. Soltau, H.-K. Kuo, B. Kingsbury, and G. Saon, "Exploiting diversity for spoken term detection," in *Proc. ICASSP*, 2013.

[25] B. Kingsbury, J. Cui, X. Cui, M. Gales, K. Knill, J. Mamou, L. Mangu, D. Nolden, M. Picheny, B. Ramabhadran, R. Schluter, A. Sethy, and P. Woodland, "A high-performance cantonese keyword search system," in *Proc. ICASSP*, 2013.

[26] U. Chaudhari and M. Picheny, "Matching criteria for vocabulary-independent search," in *IEEE Transactions on Audio Speech and Language Processing*, 2012, vol. 20, pp. 1633–1642.

[27] L. Mangu, H. Soltau, H.-K. Kuo, and G. Saon, "The IBM keyword search system for the DARPA RATS program," in *Proc. ASRU*, 2013.