Keyword Search Based on WFST Indexing

Wang Jian

Feb. 2, 2016

Keyword Search

- also known as Spoken Term Detection
- Traditional Approaches
  - LVCSR-based
    - LVCSR followed by a text searching
  - Acoustic KWS
    - Viterbi search on a network consists of keywords and garbage models
  - Phonetic Search
    - search on a lattice of phonemes

WFST-based Indexing

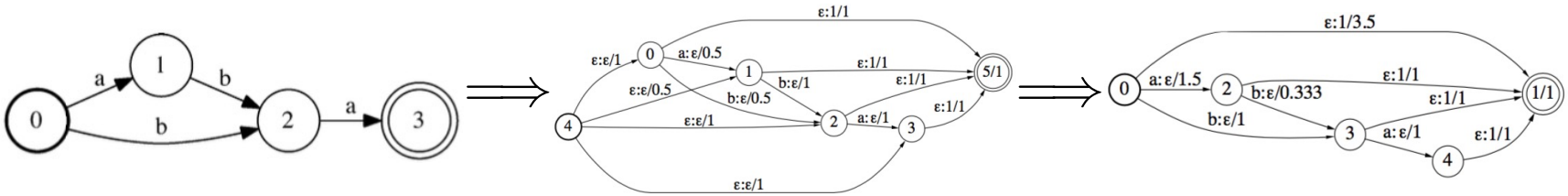
- Indexation WFST(T)
  - every path of indexation WFST represents
    - input: keyword
    - output: all utterances contain the keyword (with timing information)
- Search Method
  - convert keyword to a WFST(T)
  - compose two WFSTs $R = X \circ T$
  - paths on R produce the result utterance set

WFST-based Indexing

- Indexation WFST(T)
  - every path of indexation WFST represents
    - input: keyword
    - output: all utterances contain the keyword (with timing information)
- Search Method
  - convert keyword to a WFST(T)
  - compose two WFSTs $R = X \circ T$
  - paths on R produce the result utterance set

WFST-based Indexing

- Factor WFST
  - Given two strings u and v, v is a *factor* (substring) of u, if $u = xvy$ for some x and y
  - The *factor WFST* $F(u)$ of a string u is the minimal deterministic WFST recognizing exactly the set of factors of u



Utterance Level Keyword Search

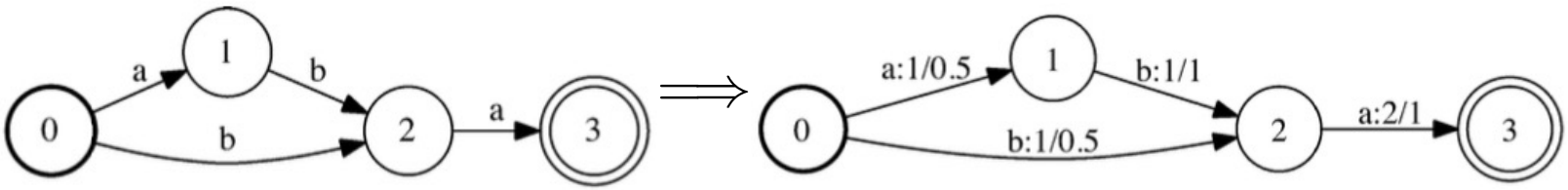
- Indexing
  1. run a LVCSR system and output a lattice for every utterance
  2. convert every lattice to a WFST(A) with word/phoneme as input/output and probability as weight
  3. construct $F(A)$ for every A
  4. take the union of all $F(A)$s
- Searching
  1. convert query/keyword to a WFST(X)
  2. compose the two WFST $R = \Pi_2(X \circ T)$
  3. extract the most likely paths of R

Utterance Level Keyword Search

- Indexing
  1. run a LVCSR system and output a lattice for every utterance
  2. convert every lattice to a WFST(A) with word/phoneme as input/output and probability as weight
  3. construct $F(A)$ for every A
  4. take the union of all $F(A)$s
- Searching
  1. convert query/keyword to a WFST(X)
  2. compose the two WFST $R = \Pi_2(X \circ T)$
  3. extract the most likely paths of R

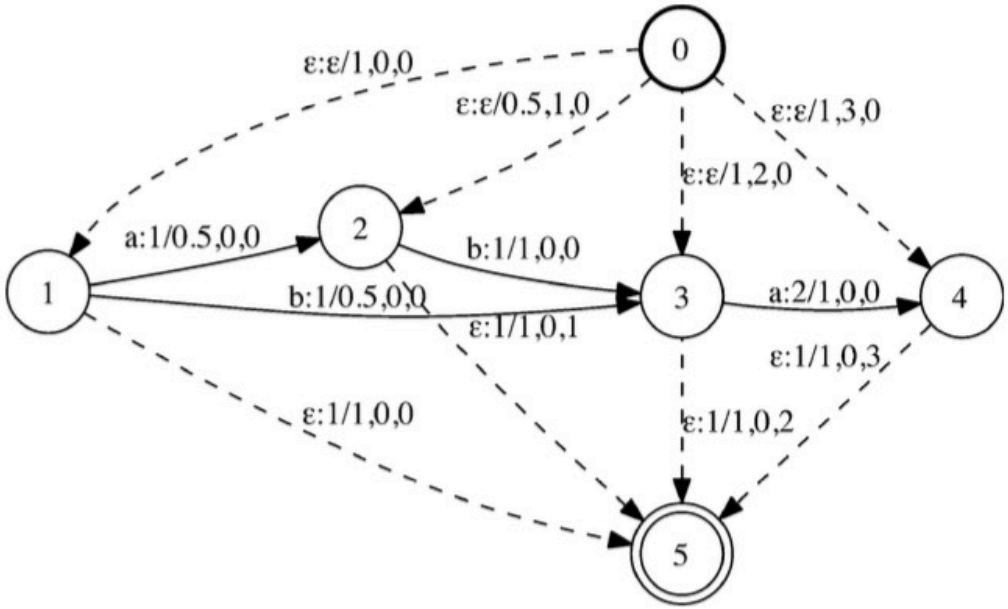
Timed Keyword Search

- Cluster the arcs with the same input label and overlapping time-spans.



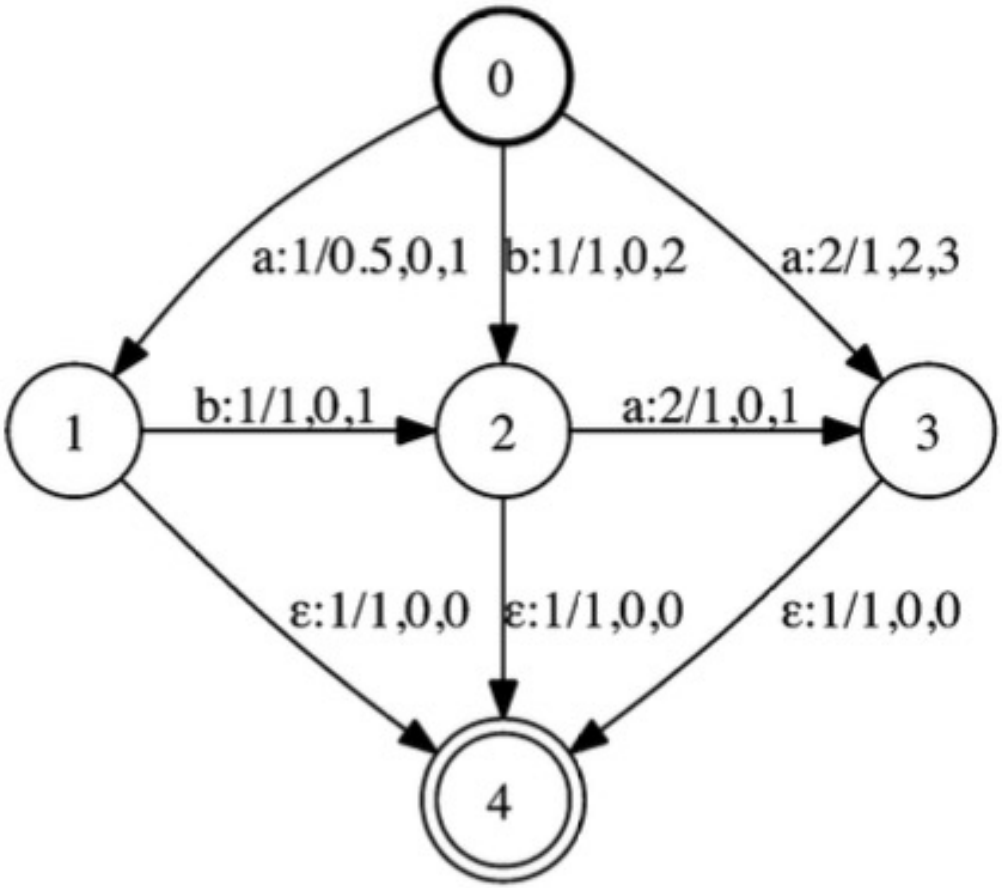
Timed Keyword Search

- generate factor with timing informations



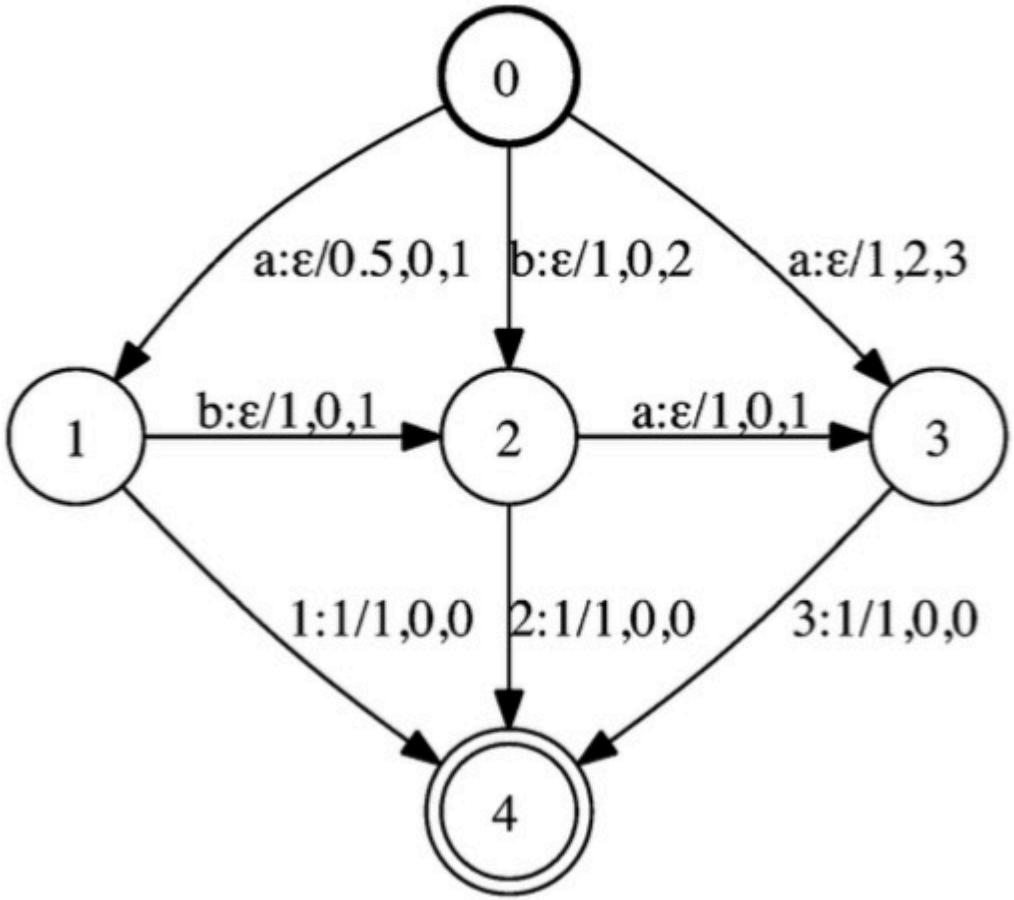
Timed Keyword Search

- merge arc with same input-output pair(overlapped labels)



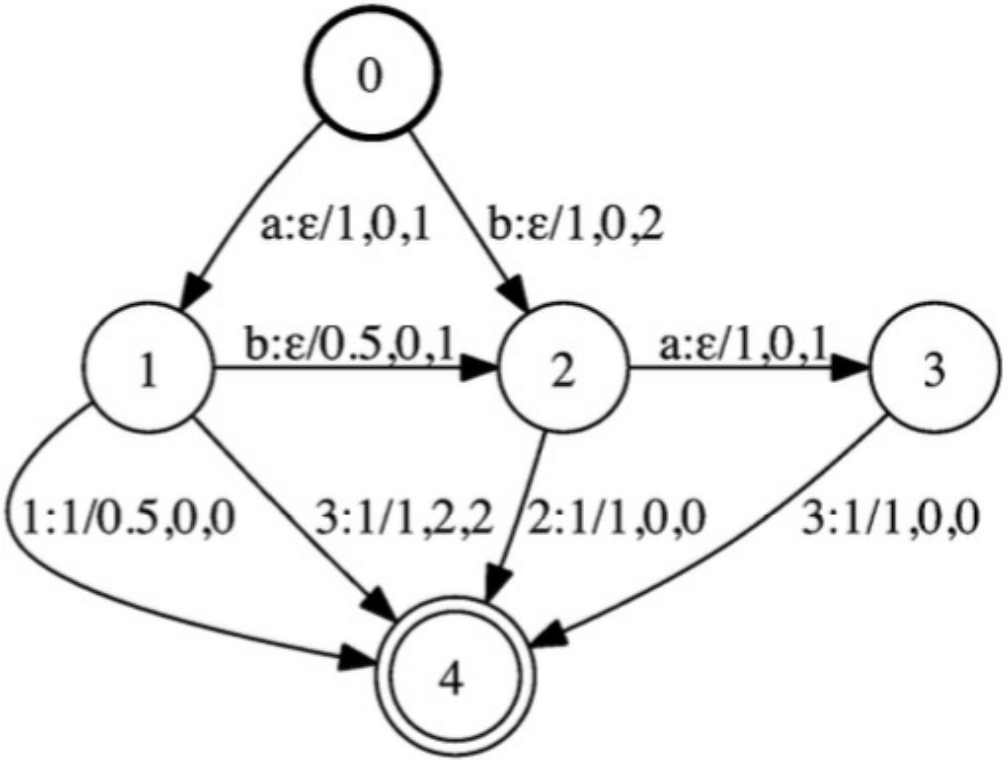
Timed Keyword Search

- factor disambiguation



Timed Keyword Search

- factor optimization



Setup

- **Scripts** Kaldi `egs/babel/s5c/`
- **dataset** 57904 utters, 41 hours
- **model** SAT fmllr gmm model (tri5a) with same dataset

Results

lmwt	ATWV	OTWV	STWV	MTWV/THRESH	Recall
8	0.5750	0.6318	0.9618	0.6318/0.260	0.9627
9	0.5767	0.6353	0.9618	0.6353/0.305	0.9627
10	0.5818	0.6352	0.9618	0.6291/0.252	0.9627
11	0.5750	0.6410	0.9618	0.6275/0.291	0.9627
12	0.5803	0.6352	0.9618	0.6237/0.299	0.9627

References

- utterance level search
  - Allauzen, Cyril, Mehryar Mohri, and Murat Saraclar. General Indexation of Weighted Automata: Application to Spoken Utterance Retrieval. In Proceedings of the Workshop on Interdisciplinary Approaches to Speech Indexing and Retrieval at HLT-NAACL 2004, 3340.
- timed search
  - Can, Doan, and Murat Saraclar. Lattice Indexing for Spoken Term Detection. Audio, Speech, and Language Processing, IEEE Transactions on 19, no. 8 (2011): 233847.