Automatic Speech Recognition (CS753)
Lecture 4: WFST algorithms contd. + WFSTs in ASR

Instructor: Preethi Jyothi
August 3, 2017
Quiz-1 Postmortem

- Common Mistakes:
  - Missing insertion/deletion in E.fst
  - Forgot to mark final states/self-loops
  - Output vocabulary for T.fst has to be complete words, “bad”, “bead”, etc. rather than letters
Project Proposal

• Start brainstorming!

• In case of doubt, discuss potential ideas with me during my office hours (Thur, 5:00 pm to 6:30 pm)

• Once decided, you will have to fill out a form specifying:
  • Title of the project
  • Names/roll numbers of all project members
  • A 300-400 word abstract of the proposed project

• Due by 11:59 pm on Aug 14th
Composition: Recap

- If $T_1$ transduces $x$ to $z$, and $T_2$ transduces $z$ to $y$, then $T_1 \circ T_2$ transduces $x$ to $y$

- Note: output alphabet of $T_1 \subseteq$ input alphabet of $T_2$

- E.g. If $T_1$ removes punctuation symbols from a string, and $T_2$ changes uppercase letters to lowercase letters, then $T_1 \circ T_2$ brings about both changes
Determinization: Recap

- A (W)FST is **deterministic** if:
  - Unique start state
  - No two transitions from a state share the same input label
  - No epsilon input labels

- Not all WFSAs can be determinized
Determinization: Weighted FSA

Some *Weighted*-FSAs are not determinizable! [M97]

Weight of string $ab^n c = n$ and weight of $ab^n d = 2n$

After seeing $ab^n$ an FSA can’t remember $n$

Determinization: Recap

- A (W)FST is **deterministic** if:
  - Unique start state
  - No two transitions from a state share the same input label
  - No epsilon input labels

- Not all WFSAs can be determinized

- Guaranteed to yield a deterministic WFSA under some technical conditions characterising the automata (e.g. twins property)
Minimization

Minimization: find an equivalent deterministic FSA with the least number of states (and transitions)

Unweighted FSAs have a unique minimal FSA [Aho74]

Obtained by identifying and merging equivalent states

Minimization: Weighted FSA

Two states are equivalent only if for every input string, the outcome — weight assigned to the string, if accepted — starting from the two states are the same.

Redistribute weights before identifying equivalent states.
Minimization: Weighted FSA

Reweighting OK as long as resulting WFSA is equivalent

Can reweight using a “potential function” on states

“Weight pushing”: Reweighting using a potential function that optimally moves weights towards the start state
Minimization: Weighted FSA

After weight-pushing, can simply apply unweighted FSA minimization (treating label/weight as label)

Guaranteed to yield a minimal WFSA (under some technical conditions required for weight-pushing)
Toolkits to work with finite-state machines

- AT&T FSM Library (no longer supported)
  http://www3.cs.stonybrook.edu/~algorith/implement/fsm/implement.shtml

- RWTH FSA Toolkit
  https://www-i6.informatik.rwth-aachen.de/~kanthak/fsa.html

- Carmel
  https://www.isi.edu/licensed-sw/carmel/

- MIT FST Toolkit
  http://people.csail.mit.edu/ilh/fst/

- OpenFST Toolkit (actively supported)
  http://www.openfst.org/twiki/bin/view/FST/WebHome
Brief Introduction to OpenFst
Quick Intro to OpenFst (www.openfst.org)

Input alphabet (in.txt)

Output alphabet (out.txt)

"0" label is reserved for epsilon

A.txt

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>&lt;eps&gt;</td>
<td>an</td>
<td>a</td>
<td>an</td>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>&lt;eps&gt;</td>
<td>n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>a</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Quick Intro to OpenFst (www.openfst.org)
Compiling & Printing FSTs

The text FSTs need to be “compiled” into binary objects before further use with OpenFst utilities

- Command used to compile:

  ```
  fstcompile --isymbols=in.txt --osymbols=out.txt A.txt
  A.fst
  ```

- Get back the text FST using a print command with the binary file:

  ```
  fstprint --isymbols=in.txt --osymbols=out.txt A.fst A.txt
  ```
Drawing FSTs

Small FSTs can be visualized easily using the draw tool:

```bash
fstdraw --isymbols=in.txt --osymbols=out.txt A.fst | dot -Tpdf > A.pdf
```
FSTs can get very large!
WFSTs applied to ASR
WFST-based ASR System

- Acoustic Models
  - Acoustic Indices
  - Triphones

- Context Transducer
  - Monophones

- Pronunciation Model
  - Words

- Language Model
  - Word Sequence
WFST-based ASR System

Acoustic Models → Context Transducer → Pronunciation Model → Language Model

Acoustic Indices → Triphones

H

\[ \text{One 3-state HMM for each triphone} \]

FST Union + Closure

Resulting FST

H

\[ \text{a/a\_b} \quad f_1:ɛ \quad f_3:ɛ \quad f_4:ɛ \quad f_5:ɛ \]

\[ \text{f_0:a:a\_b} \quad f_2:ɛ \quad f_4:ɛ \quad f_6:ɛ \]

\[ \text{b/a\_b} \]

\[ \text{x/y\_z} \]
WFST-based ASR System

C⁻¹: Arc labels: “monophone : phone / left-context_right-context”
WFST-based ASR System

Figure reproduced from “Weighted Finite State Transducers in Speech Recognition”, Mohri et al., 2002
WFST-based ASR System

Acoustic Indices \rightarrow \text{Acoustic Models} \rightarrow \text{Triphones} \rightarrow \text{Context Transducer} \rightarrow \text{Monophones} \rightarrow \text{Pronunciation Model} \rightarrow \text{Language Model} \rightarrow \text{Words} \rightarrow \text{Word Sequence}

WFST-based ASR System

0 \rightarrow \text{the} \rightarrow \text{birds/0.404} \rightarrow \text{animals/1.789} \rightarrow \text{were/0.693} \rightarrow \text{are/0.693} \rightarrow \text{walking} \rightarrow G

0 \rightarrow \text{boy/1.789}