WFSTs for ASR

Peter Bell

Automatic Speech Recognition – ASR Lecture 10
13 February 2020
Weighted Finite State Transducers

- Weighted finite state automaton that transduces an input sequence to an output sequence (Mohri et al 2008)
- States connected by transitions. Each transition has
  - input label
  - output label
  - weight

- Weights use the log semi-ring or tropical semi-ring – with operations that correspond to multiplication and addition of probabilities

- There is a single start state. Any state can optionally be a final state (with a weight)

- Used by Kaldi
Weighted Finite State Acceptors

![Diagram of Weighted Finite State Acceptors](image_url)
Weighted Finite State Transducers

Acceptors

Transducers
Weighted Finite State Transducers

**Acceptor**

```
0 1 2 3 4
```

- 0 to 1 with d/1
- 1 to 2 with ey/0.5, ae/0.5
- 2 to 3 with t/0.3, dx/0.7
- 3 to 4 with ax/1

**Transducer**

```
0 1 2 3 4
```

- 0 to 1 with d: data/1, d: dew/1
- 1 to 2 with ey: ϵ/0.5, ae: ϵ/0.5
- 2 to 3 with t: ϵ/0.3, dx: ϵ/0.7
- 3 to 4 with ax: ϵ/1

- 5 to 6 with uw: ϵ/1
Composition  Combine transducers $T_1$ and $T_2$ into a single transducer acting as if the output of $T_1$ was passed into $T_2$.

Determinisation  Ensure that each state has no more than a single output transition for a given input label

Minimisation  Transforms a transducer to an equivalent transducer with the fewest possible states and transitions

Weight pushing  Push the weights towards the front of the path
The HMM as a WFST

\[
\text{ax}_1 <\text{eps}> \quad \rightarrow \quad \text{ax}_2 <\text{eps}> \quad \rightarrow \quad \text{ax}_3 <\text{eps}> \quad \rightarrow \quad S_E
\]

\[
0 \quad \rightarrow \quad 1 \quad \rightarrow \quad 2 \quad \rightarrow \quad 3
\]
Applying WFSTs to speech recognition

Represent the following components as WFSTs

<table>
<thead>
<tr>
<th>transducer</th>
<th>input sequence</th>
<th>output sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G$ word-level grammar</td>
<td>words</td>
<td>words</td>
</tr>
<tr>
<td>$L$ pronunciation lexicon</td>
<td>phones</td>
<td>words</td>
</tr>
<tr>
<td>$C$ context-dependency</td>
<td>CD phones</td>
<td>phones</td>
</tr>
<tr>
<td>$H$ HMM</td>
<td>HMM states</td>
<td>CD phones</td>
</tr>
</tbody>
</table>

Composing $L$ and $G$ results in a transducer $L \circ G$ that maps a phone sequence to a word sequence.

$H \circ C \circ L \circ G$ results in a transducer that maps from HMM states to a word sequence.
Grammar - unigram

the:the/8.1333
piper:piper/7.4401
pickled:pickled/7.4401
picked:picked/7.4401
peter:peter/7.4401
peppers:peppers/7.4401
peck:peck/7.4401
of:of/7.4401
a:a/8.1333
0/0.0047087
Grammar - bigram

ASR Lecture 10  WFSTs for ASR
Weight pushing

0

peter:peter/0.5
the:the/1

1

piper:piper/4
picked:picked/10
peck:peck/1

2

3
Weight-pushed version

\[
\begin{align*}
0 & \xrightarrow{\text{peter:peter/3.2173}} 1 \\
& \xrightarrow{\text{the:the/0.71973}} 3 \\
1 & \xrightarrow{\text{piper:piper/0.64242}} 2 \\
& \xrightarrow{\text{picked:picked/6.6424}} 2 \\
3 & \xrightarrow{\text{peck:peck/0.63995}} 2
\end{align*}
\]
Lexicon, $L$

```
0
  \_ 1 \_ 9 \_ 16
|        |        |
| p:peck | eh:<eps> | k:<eps> |
  \_ 10 \_ 17 \_ 22 \_ 27
|        |        |        |
| p:peppers | eh:<eps> | p:<eps> | er:<eps> | z:<eps> |
  \_ 11 \_ 18 \_ 23
|        |        |        |
| p:peter | iy:<eps> | t:<eps> | er:<eps> |
  \_ 12 \_ 19 \_ 24
|        |        |        |
| p:picked | ih:<eps> | k:<eps> | t:<eps> |
  \_ 13 \_ 20 \_ 25 \_ 28
|        |        |        | l:<eps> |
|        |        |        | d:<eps> | 29
|        |        |        |
| p:piper | ih:<eps> | k:<eps> | ah:<eps> | l:<eps> |
  \_ 14 \_ 21 \_ 26
|        |        |        | er:<eps> | d:<eps> |
  \_ 15
| iy:<eps> |
|        |        |
| dh:the | ay:<eps> | p:<eps> | er:<eps> |
  \_ 6 \_ 14 \_ 21 \_ 26
|        |        |        |        |
| dh:the | ah:<eps> | iy:<eps> |
  \_ 7 \_ 8 \_ 15
|        |        |
```
Determinization – $det(L)$
Minimization – \( \min(\det(L)) \)
Composition: $L \circ G$
$\min(\det(L \circ G))$
Context-dependency: biphones

![Diagram showing context-dependency in biphones](image)
Context-dependency: triphones
C ⋄ L ⋄ G – biphones

ASR Lecture 10  WFSTs for ASR
HMM transducer, $H$

- We can also use a version that outputs context-dependent phones
- $H$ can be used to encode state-tying
Decoding using WFSTs

- Combining the transducers gives an overall HMM structure for the ASR system – but minimisation and determination operations on the WFSTs means it is much smaller than naively combining the HMMs
- But it is important in which order the algorithms are combined otherwise the transducers may “blow-up”
- Standard approach is to determinize and minimize after each composition
- In Kaldi, ignoring one or two details

\[ HCLG = \min(\det(H \circ \min(\det(C \circ \min(\det(L \circ G)))))\)\]