Weighted Finite-State Transducers

Important operations and practical work

Rachid Riad

February 5, 2018

LSCP/NPI, ENS

Table of contents

Motivations

Motivations

Several Components are finite-state

1. Language models (e.g. *n-grams*)
2. Lexicons
3. Phonological rules
4. HMM tri-phones

Motivations

Several Components are finite-state

1. Language models (e.g. *n-grams*)
2. Lexicons
3. Phonological rules
4. HMM tri-phones

Use same representation and algorithms for all

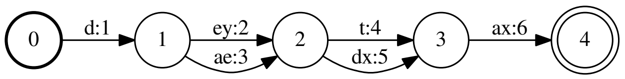
1. Consistency
2. Powerful algorithms available at all levels
3. Flexibility to combine or factor in unforeseen ways

[Mohri, 1997]

Transducers and Acceptors

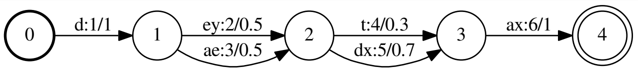
Finite State Acceptors

- **Finite State Automaton**
- Example I: **Pronunciation model acceptor:**
- 'data' accepted pronunciations



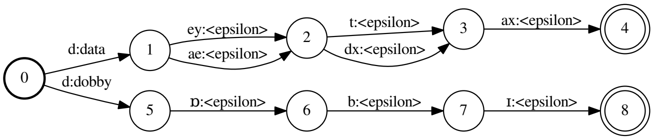
Finite State Acceptors

- Finite automaton with weights: **Weighted Finite State Automaton**
- Example II: **Pronunciation model acceptor with weights**
- 'data' weighted accepted pronunciations



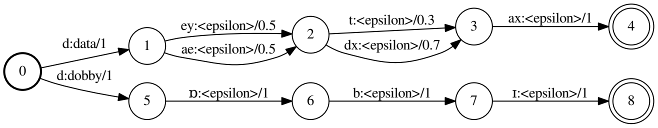
Finite State Transducers

- Finite State Transducer
- Example III: **Pronunciation lexicon dictionary**
- Dictionary of pronunciations transformation from sequence of phones to words in {data, dobby}



Weighted Finite State Transducers

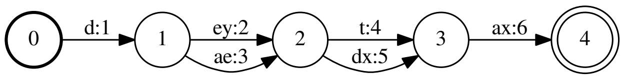
- Weighted Finite State Transducer
- Example IV: **Pronunciation lexicon dictionary with weights**
- Weighted Dictionary of pronunciations transformation from sequence of phones to words in {data, dobby}



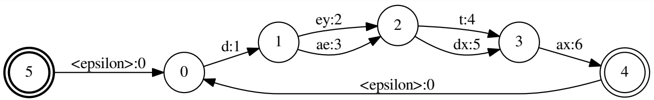
Basic operations on WFSTs

Closure

A :

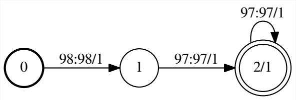

0
d:1
1
ey:2
ae:3
2
t:4
dx:5
3
ax:6
4

A*:

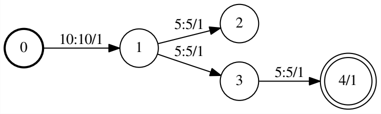

5
<epsilon>:0
0
d:1
1
ey:2
ae:3
2
t:4
dx:5
3
ax:6
4
<epsilon>:0

Union

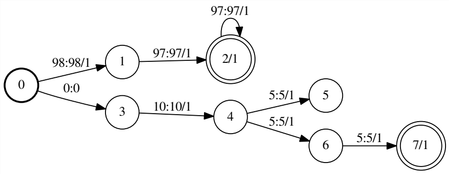
'Parallel combination' of A and B:



(a)



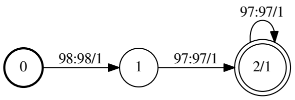
(b)



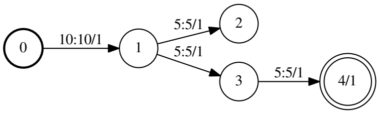
(c)

Concatenation

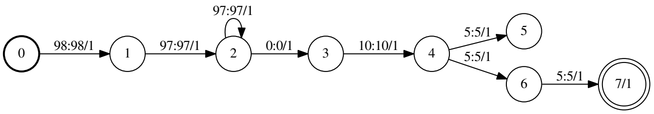
'Serial combination' of A and B:



(d)



(e)



(f)

Composition

Motivations to decompose the problem

1. Hard to build everything from scratch
2. Combine in unexpected ways and compact representation
3. Simplification of construction and modelling

Composition

Motivations to decompose the problem

1. Hard to build everything from scratch
2. Combine in unexpected ways and compact representation
3. Simplification of construction and modelling

Several WFSTs building blocks

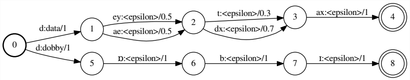
1. HMM tri-phones from acoustic modelling
2. Lexicon pronunciation dictionary from linguists
3. Use extra dataset to build the Language models

Dive into the algorithm details:
https://www.youtube.com/watch?v=DyY69sX7RGk

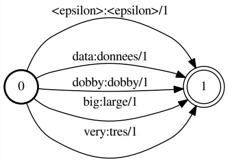
Composition

- Example V: **Composition of English Pronunciation and English to French**
- One WFST going from **English Phones** to **English Words**
- One WFST going from **English words** to **French Words**
- Composition outputs an WFST going from **English phones** to **French Words**

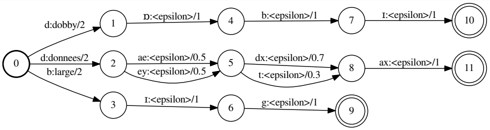
Composition



(g)



(h)



(i)

Practical work

Summary

- Virtual Machine to avoid any installation issues
- Download it and try it **as soon as possible**!
- Additional Jupyter notebook for the figures of this presentation
- OpenFST and Kaldi libraries
- 3 Exercises from Basic commands to Mispell detector

Questions?

Semirings I

WFSTs and their algorithms rely on algebraic structures called **semirings**.

Definition

A **semiring** is an algebraic structure $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$:

1. *Associative and Commutative* operation $\oplus$, with the identity $\overline{0}$
2. *Associative* operation $\otimes$, with the identity $\overline{1}$
3. $\oplus$ distributes over $\otimes$:
   $\forall a, b, c \in \mathbb{K}$
   $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$
   $c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b)$
4. $\overline{0}$ is an annihilator for $\otimes$: $\forall a \in \mathbb{K}, a \otimes \overline{0} = \overline{0} \otimes a = \overline{0}$

Implications for optimisation, search, and combinations algorithms

Semirings II

Seminring	Set	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Boolean	0,1	$\vee$	$\wedge$	0	1
Probability	$\mathbb{R}_+ \cup \{+\infty\}$	$+$	$\times$	0	1
Log	$\mathbb{R}_+ \cup \{-\infty, +\infty\}$	$\oplus_{log}$	$+$	∞	0
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0

Table 1: Example of Semirings: $x \oplus_{log} y = -log(e^{-x} + e^{-y})$

[Mohri et al., 2002]

Finite State Automaton and Finite State Transducer

Definition

Definition A WFST is defined as an 8-tuple, $T = (\Sigma, \Delta, Q, I, F, E, \lambda, \rho)$. Here Σ represents the finite alphabet of input symbols, Δ represents the finite output alphabet, Q represents the finite set of states, $I \subseteq Q$ the set of initial states, $F \subseteq Q$ the set of final states, $E \subseteq Q \times (\Delta \cup \{\epsilon\}) \times (\Sigma \cup \{\epsilon\}) \times \mathbb{K} \times Q$ a finite set of state-to-state transitions, $\lambda : I \rightarrow K$ the initial weight function, and $: F \rightarrow \mathbb{K}$ the final weight function mapping F to $\mathbb{K}$.

- A WFSA is simply a WFST where the output labels have been omitted
- Similarly, FSAs and FSTs lack weights on the arcs or states

References i

Mohri, M. (1997).
Finite-state transducers in language and speech processing.
Computational linguistics, 23(2):269–311.

Mohri, M., Pereira, F., and Riley, M. (2002).
Weighted finite-state transducers in speech recognition.
Computer Speech & Language, 16(1):69–88.