Translation with Weighted Finite-State Devices
Some Models of Translation

- IBM Models 1-5
- Hidden Markov Model
- Phrase-Based Models

Q: What do all of these things have in common?

A: They all define *weighted regular languages* over a set of output sentences.
Desiderata

• We need efficient algorithms and data structures to:
  • Encode all of the strings in the language.
  • Assign probabilities to all of those strings.
    • Via products such as $p(e)p(f | e)$.
  • Find the string with the highest probability.
  • Compute expectations over substrings.
  • Compute mappings between strings.
Practical Implementation

- Build each step as an individual transducer.
- Compose at runtime (pruning at each step).
- Assign probabilities to all of those strings.
  - Via products such as $p(e)p(f|e)$.
- Find the string with the highest probability.
- Compute expectations over substrings.
- Compute mappings between strings.
Regular Languages

$$\mathcal{L}_1 = \{ a a a, a b a, a a b, a b b \}$$

$$\mathcal{L}_2 = a^* = \{ a, aa, aaaa, \ldots \}$$

$$\mathcal{L}_3 = \{ "\text{the north wind howls}" \}$$

finite-state automata
Regular Languages

\{ \epsilon \} \text{ is regular}

\{ a \} \text{ is regular}

\mathcal{L}_1 \cup \mathcal{L}_2 \text{ is regular if } \mathcal{L}_1 \text{ and } \mathcal{L}_2 \text{ are regular}

\mathcal{L}_1 \cdot \mathcal{L}_2 \text{ is regular}

\mathcal{L}_1^* \text{ is regular}
We want a function:

\[ f : \mathcal{L} \rightarrow \mathbb{R}^+ \]

such that:

\[ f(w) \in [0, 1] \]

\[ \sum_{w} f(w) \in [0, 1] \]
Probabilistic Regular Languages

We want a function:

\[ f : \mathcal{L} \rightarrow \mathbb{R}^+ \]
Probabilistic Regular Languages

We want a function:

\[ f : \mathcal{L} \rightarrow \mathbb{R}^+ \]
Finite-State Transducers

We want a binary relation:

\[ r \subseteq \mathcal{L}_1 \times \mathcal{L}_2 \]
Finite-State Transducers

We want a binary relation:

\[ r \subseteq \mathcal{L}_1 \times \mathcal{L}_2 \]
We want a binary relation:

\[ r \subseteq \mathcal{L}_1 \times \mathcal{L}_2 \]

\[ r = \{ \text{a a a, b b b, a b a, b a b, a a b, b b a, a b b, b a a} \} \]
Weighted Finite-State Transducers

We have a binary relation:

\[ r \subseteq \mathcal{L}_1 \times \mathcal{L}_2 \]

\[ r = \{ a \ a \ a, \ b \ b \ b \ a a b, \ b b a \ a b b, \ b a a \} \]
**Weighted Finite-State Transducers**

We have a binary relation:

$$r \subseteq \mathcal{L}_1 \times \mathcal{L}_2$$

We want a function:

$$f : r \rightarrow \mathbb{R}^+$$

$$r = \{ a\ a\ a,\ b\ b\ b,\ a\ b\ a,\ b\ a\ b,\ a\ a\ b,\ b\ b\ a,\ a\ b\ b,\ b\ a\ a \}$$
Weighted Finite-State Transducers

We have a binary relation:

\[ r \subseteq \mathcal{L}_1 \times \mathcal{L}_2 \]

We have a function:

\[ f : r \to \mathbb{R}^+ \]

\[
\begin{align*}
\langle a \ a \ a, \ b \ b \ b \rangle & \to \frac{1}{9} \\
\langle a \ b \ a, \ b \ a \ b \rangle & \to \frac{2}{9} \\
\langle a \ a \ b, \ b \ b \ a \rangle & \to \frac{2}{9} \\
\langle a \ b \ b, \ b \ a \ a \rangle & \to \frac{4}{9}
\end{align*}
\]
**Weighted Finite-State Transducers**

We have a binary relation:

\[ r \subseteq \mathcal{L}_1 \times \mathcal{L}_2 \]

We have a function: \( f : r \rightarrow \mathbb{R}^+ \)

(Single string, multiple elements of \( r \))
Algorithms don’t Change...

- Shortest path (e.g. Dijkstra, A*): most probable *pair*
- Determinization (not all can be determinized)
- But not w.r.t. to pairs, not single string!
- Lazy composition (e.g. intersection): \( p(e)p(f|e) \)
Composing Transducers
Composing Transducers
Composing Transducers

composed with:

```
  a:b
  a:b
  a:b
  a:b
  a:b
  b:a
  b:a
  b:a
```

```
  a
  a
  a
```
Composing Transducers

composed with:

yields:
Composing Transducers

computes a function:
\[ f : \mathcal{L}_1 \rightarrow \mathcal{P}(\mathcal{L}_2) \]

composed with:

yields:
Composing Transducers

A transducer composed with:

\[ f : \mathcal{L}_1 \rightarrow \mathcal{P}(\mathcal{L}_2) \]

with weights:

\[ f : \mathcal{L}_1 \rightarrow \mathcal{P}(\mathcal{L}_2 \times \mathbb{R}^+) \]

yields:

\[ \text{b} \rightarrow \text{b} \rightarrow \text{b} \rightarrow \text{b} \]
IBM Model 4
Although north wind howls, but sky still very clear.

Although north wind howls, but sky still very clear.
Although north wind howls, but sky still very clear.

虽然北风呼啸，但天空依然十分清澈。

虽然
IBM Model 4

Although north wind howls, but sky still very clear.

$p_f(1|虽然)$
Although north wind howls, but sky still very clear.

IBM Model 4

Although north wind howls, but sky still very clear.

虽然北风呼啸，但天空依然十分清澈。

虽然北风呼啸，天空天空依然清澈。
Although north wind howls, but sky still very clear.

IBM Model 4
Although north wind howls, but sky still very clear.

虽然北风呼啸，但天空依然十分清澈。
IBM Model 4

Although north wind howls, but sky still very clear.

However
IBM Model 4

Although north wind howls, but sky still very clear.

Although north wind howls, but sky still very clear.

However

$p_t(\text{However} | \text{虽然})$
Although north wind howls, but sky still very clear.

However north wind strong, the sky remained clear. under the
IBM Model 4

Although north wind howls, but sky still very clear.

However north wind strong, the sky remained clear. under the
Although north wind howls, but sky still very clear.

Although north wind howls, but sky still very clear.

虽然北风呼啸，但天空依然十分清澈。

However north wind strong, the sky remained clear. under the
Although north wind howls, but sky still very clear.  

However north wind strong, the sky remained clear. under the

$p_d(0|\text{However})$
Although north wind howls, but sky still very clear.

However north wind strong, the sky remained clear. under the
Although north wind howls, but sky still very clear.

However, north wind strong, the sky remained clear. under the

\[ p_d(8 | north) \]
Although north wind howls, but sky still very clear.

However north wind strong, the sky remained clear. under the
Although north wind howls, but sky still very clear.

However, the sky remained clear under the strong north wind.
Although north wind howls, but sky still very clear.

However, north wind strong, the sky remained clear under the strong north wind.

$p(\text{English, alignment}|\text{Chinese}) = \prod_{p_f} \prod_{p_t} \prod_{p_d}$
Although north wind howls, but sky still very clear.

IBM Model 4

Although north wind howls, but sky still very clear.

虽然北风呼啸，但天空依然十分清澈。
Although north wind howls, but sky still very clear.

虽然北风呼啸，但天空依然十分清澈。
Although north wind howls, but sky still very clear.

虽然北风呼啸，但天空依然十分清澈。
Although north wind howls, but sky still very clear.

雖然 北 風 呼嘯，但 天空 依然 十分 清澈。
Although north wind howls, but sky still very clear.

Although north wind howls, but sky still very clear.

虽然北风呼啸，但天空依然十分清澈。
Although north wind howls, but sky still very clear.

虽然 北 风 呼啸，但 天空 依然 十分 清澈。
Although north wind howls, but sky still very clear.

虽然北风呼啸，但天空依然十分清澈。
Although north wind howls, but sky still very clear.

Although north wind howls, but sky still very clear.

虽然北风呼啸，但天空依然十分清澈。
IBM Model 4

Although north wind howls, but sky still very clear.

Although north wind howls, but sky still very clear.

With weights, defines a bigram LM!
IBM Model 4

Although north wind howls, but sky still very clear.

Although north wind howls, but sky still very clear.

fertility transducer
IBM Model 4

However north wind strong, the sky remained clear. Under the
IBM Model 4

substitution transducer

However, north wind strong, the sky remained clear. Under the

虽然 北风 呼啸，天空 天空 依然 清澈。 ε ε

虽然是：However 虽然：Although
IBM Model 4

However, north wind strong, the sky remained clear. Under the
IBM Model 4

This is not a transducer, but we can build an FSA that accepts all permutations.

However north wind strong, the sky remained clear. under the
IBM Model 4

This is not a transducer, but we can build an FSA that accepts all permutations.

To translate: run the entire cascade in reverse!

However north wind strong, the sky remained clear. under the
IBM Model 4

This is not a transducer, but we can build an FSA that accepts all permutations.

To translate: run the entire cascade in reverse!

However, north wind strong, the sky remained clear. Under the

___  ___  ___  ___  ___  ___  ___  ___  ___  ___  ___  ___  ___  ___  ___

Looks like stack decoding algorithm Matt showed you, but doesn’t know as much about graph topology.
Phrase-based Models

Although north wind howls, but sky still very clear.

虽然 北 风 呼啸, 但 天空 依然 十分 清澈。
Phrase-based Models

Although north wind howls, but sky still very clear.
Although north wind howls, but sky still very clear.
Although north wind howls, but sky still very clear.
Although north wind howls, but sky still very clear.
Although north wind howls, but sky still very clear.

Phrase-based Models

However the strong north wind, the sky remained clear under

However
Although north wind howls, but sky still very clear.

However, the strong north wind, the sky remained clear under it.
Although north wind howls, but sky still very clear.

However, the sky remained clear under the strong north wind.

\[
p(\text{English, alignment}|\text{Chinese}) = p(\text{segmentation}) \cdot p(\text{translations}) \cdot p(\text{reorderings})
\]
Phrase-based Models

Although north wind howls, but sky still very clear.

虽然北风呼啸，但天空依然十分清澈。
Phrase-based Models

Although north wind howls, but sky still very clear.
Although north wind howls, but sky still very clear.

Although north wind howls, but sky still very clear.

即使北风呼啸，但天空依然十分清澈。
Phrase-based Models

Although north wind howls, but sky still very clear.

As with IBM Models, construct a cascade of transducers.
Phrase-based Models

Although north wind howls, but sky still very clear.

As with IBM Models, construct a cascade of transducers. Determinize, remove epsilons, minimize at each step.
Phrase-based Models

Although north wind howls, but sky still very clear.

As with IBM Models, construct a cascade of transducers. Determinize, remove epsilons, minimize at each step. See Kumar et al. (2004) for a complete implementation.
A Note on Weights

In general, on arbitrary semirings: $\langle A, \oplus, \otimes, 1, 0 \rangle$
A Note on Weights

Where does $\frac{1}{9}$ come from?

In general, on arbitrary semirings: $\langle A, \oplus, \otimes, 1, 0 \rangle$
A Note on Weights

Where does $\frac{1}{9}$ come from? A: $1 \times \frac{1}{3} \times \frac{1}{3}$

In general, on arbitrary semirings: $\langle A, \oplus, \otimes, 1, 0 \rangle$
A Note on Weights

Where does $\frac{1}{9}$ come from?

In the log domain: $\log(1) + \log(\frac{1}{3}) + \log(\frac{1}{3})$

In general, on arbitrary semirings: $\langle A, \oplus, \otimes, 1, 0 \rangle$
Statistical Machine Translation

- formal language theory
- decision theory
- linguistics
- machine learning
- algorithms
Next Week:

Non-finite-state models of translation!