CS 378 Lecture 18

Today
- Finish IBM Model 1
- Phrase-based MT
- Syntactic MT (briefly)
- Neural MT and seq2seq models

Recap IBM Model 1

\[ \overline{s} = \text{Je fais un bureau NULL} \]
\[ \overline{a} = [a_1 = 1, a_2 = 2, a_3 = 2, a_4 = 3, a_5 = 4] \]
\[ \overline{t} = \text{I am making a desk} \]

\[ P(\overline{t}, \overline{a} \mid \overline{s}) = \prod_{i=1}^{n} P(a_i) P(\text{til} s_{a_i}) \]

\[ P(a_i) = \text{uniform dist over } \{1, 2, \ldots, m + 1\} \]

\[ P(\text{til} s_{a_i}) \]

\[ P(\text{I} \mid \text{Je}) = 0.8 \]
Announcements

- Custom FP proposals returned w/ connectors
- Optimal lecture Tuesday
- A4
- A5

Inference in Model 1

What we care about: \( P(\bar{a} | \bar{T}, \bar{s}) \)

\[
\begin{align*}
\argmax_a P(\bar{a} | \bar{T}, \bar{s}) &= \frac{1}{m+1} \\
\frac{1}{m+1} P(a; \bar{T}; \bar{s}) &= \prod_{i=1}^{n} P(a_i | s_i) P(s_i) \\
\end{align*}
\]

\( P(\bar{a} | \bar{T}, \bar{s}) \) proportional to \( \prod_{i=1}^{n} P(t_i | s_i) \)

\( P(a_i | \bar{T}, \bar{s}) \) proportional to \( P(t_i | s_i) \)
Example

\[
P(a, I, s) \propto \begin{cases} P(I|Je) & a_1 = 1 \\ P(I|NULL) & a_1 = 2 \end{cases}
\]

\[
P(a_1|I, s) = \begin{cases} \frac{2}{3} & a_1 = 1 \\ \frac{1}{3} & a_1 = 2 \end{cases}
\]
Ex 2: \( J \) and NULL

I like

\[
P(a_1 | \overline{s_j}_t) = \begin{cases} 
0.8 & J \text{ true} \\
0 & \text{aime = } 0 \\
0.4 & \text{NULL} \end{cases} \]

\[
P(a_2 | \overline{s_j}_t) = \begin{cases} 
0.1 & J \text{ true} \\
1.0 & \text{aime = } 0 \\
0.3 & \text{NULL} \end{cases} \]

More complex models: HMM alignment model

\[
P(a_i | a_{i-1})
\]

IBM Models 2-4

Learning: Unsupervised learning: EM (Expectation Maximization)
\[
\begin{align*}
\text{maximize} & \quad \prod_{i=1}^{D} \log \sum_{\bar{a}} P(\bar{a}, \bar{t}^{(i)} | \bar{s}^{(i)}) \\
\end{align*}
\]
Seq 2 seq models

Key idea: encode source sent w/ RNN, "decode" target w/ another RNN

Encoder

I ← \( P(t_1 | \tilde{s}) = \text{softmax}(\text{wh}_i) \)

\( \text{hidden}_i \)

am going

\( \text{decoder} \)

representation of the French sentence

\[ P(t_1 | \tilde{s}) = P(t_1 | \tilde{s}) P(t_2 | \tilde{s}, t_1) \ldots \]
Feed output $T_i$ into cell input for $T_{i+1}$

**Why?**  Model capture word-word dependencies more easily

**Training**  
Given sequence pairs $(s, F)$

$$
\text{loss} = \sum \log P(t_i^* | s_j, t_{i-1}^*)
$$

Like in language modeling, assume everything up until now matches our reference/gold
Problem: Long-range dependencies

What we want is a way to look back at the input more easily.

Solution: attention