**CS/ENGRI 172, Fall 2003: Computation, Information, and Intelligence**
**11/21/03: Machine Translation**

### Machine Translation Paradigms

Machine translation (MT) is the automatic translation from a *source language* (SL) to a *target language* (TL), while preserving the meaning of the source text within the target text. There are three styles of approach to this problem: direct replacement, syntactic transfer, and interlingua. They vary in their flexibility and the amount of computational resources required.

### Statistical Machine Translation and the IBM Candide System

Assume word-for-word translations, with no insertions or deletions of words permitted between source and target sentences.[1] We compute translation probabilities using an auxiliary source of information: *alignments* in sentence pairs made up of mutual translations.

The algorithm description uses the following notation:

$\mathbf{tr}(s \to t)$ the *transition weight* (or probability) that source-language word $s$ should be translated as target-language word $t$.

$p^{(1)}, p^{(2)}, \ldots, p^{(N)}$ the source/target-language sentence pairs in the training corpus

$p^{(i)} = (s_1^{(i)} s_2^{(i)} \ldots s_{l_i}^{(i)}; t_1^{(i)} t_2^{(i)} \ldots t_{l_i}^{(i)})$ the $i$th sentence pair, where $s_1^{(i)} \ldots s_{l_i}^{(i)}$ is an $l_i$-word source-language sentence and $t_1^{(i)} \ldots t_{l_i}^{(i)}$ is its $l_i$-word translation.[2]

$(1 \leftrightarrow j_1; 2 \leftrightarrow j_2; \ldots; l_i \leftrightarrow j_{l_i})$ an *alignment*, which lists for each of the $l_i$ words in the source-language sentence which word of the target-language sentence it is aligned to.

$A_1^{(i)}, A_2^{(i)}, \ldots, A_{m_i}^{(i)}$ a set of $m_i$ possible alignments associated with sentence pair $p_i$, where $m_i = (l_i)(l_i - 1)(l_i - 2) \ldots (2)(1)$.

**Contains**$(s \leftrightarrow t)$ is the set of alignments $A$ in which source-language word $s$ is aligned with target-language word $t$.[3]

**freq**$(s \leftrightarrow t; A)$ is the number of times we have the word $s$ aligned to $t$ in alignment $A$.

Using the variable $A$ to stand for an alignment drawn from an arbitrary sentence pair, we say that every alignment $A$ has an *alignment weight* (or probability) $\mathbf{awt}(A)$.

---

[1] This is clearly a simplification of full machine translation.

[2] Note that the $s_j^{(i)}$s and $t_j^{(i)}$s do not have to be distinct. The subscript $i$ reflects that different sentence pairs may have different lengths, though we assume that corresponding source and target language sentences have the same number of words.

[3] Notice that Contains$(s \leftrightarrow t)$ can include alignments from different sentence pairs.

**An Iterative Learning Algorithm for MT**

1. Initialization: For every sentence pair $p_i$, set $\text{awt}(A_1^{(i)}) = \cdots = \text{awt}(A_{m_i}^{(i)}) = 1/(m_i)$.

2. Repeat the following steps in order until no more changes occur:

3. Update translation weights: For every source/target word pair $(s, t)$, change $\text{tr}(s \rightarrow t)$ to:

$$\sum_{A \text{ in Contains}(s \leftrightarrow t)} \text{freq}(s \leftrightarrow t, A)\text{awt}(A)$$

4. Psuedo-normalize translation weights: Change each weight $\text{tr}(s \rightarrow t)$ to

$$\frac{\text{tr}(s \rightarrow t)}{\sum_{t'} \text{tr}(s \rightarrow t')}$$

   where $t'$ ranges over all target language words.

5. Update alignment weights: For every $A_k^{(i)} = (1 \leftrightarrow j_1; 2 \leftrightarrow j_2; \ldots; l_i \leftrightarrow j_{l_i})$, change $\text{awt}(A_k^{(i)})$ to $\text{tr}(s_1 \rightarrow t_{j_1})\text{tr}(s_2 \rightarrow t_{j_2})\cdots\text{tr}(s_{l_i} \rightarrow t_{j_{l_i}})$.

6. Psuedo-normalize alignment weights: For every alignment $A_k^{(i)}$, change $\text{awt}(A_k^{(i)})$ to

$$\frac{\text{awt}(A_k^{(i)})}{\sum_{q=1}^{m_i} \text{awt}(A_q^{(i)})}$$

**Example**

Suppose we have two sentence pairs: $p_1 = (\textit{chat bleu}, \textit{blue cat})$ and $p_2 = (\textit{chat}; \textit{cat})$. This yields three alignments:

$$\begin{aligned}
A_1^{(1)} &= (1 \leftrightarrow 1; 2 \leftrightarrow 2) \quad (\text{so “}\textit{chat}\text{” aligned to “}\textit{blue}\text{”}) \\
A_2^{(1)} &= (1 \leftrightarrow 2; 2 \leftrightarrow 1) \quad (\text{so “}\textit{chat}\text{” aligned to “}\textit{cat}\text{”}) \\
A_1^{(2)} &= (1 \leftrightarrow 1) \quad (\text{only one possible choice})
\end{aligned}$$

The algorithm will then compute the following translation and alignment weights. After convergence, the translation weights indicate our learned word-for-word translations.

| | $\text{awt}(A_1^{(1)})$ | $\text{awt}(A_2^{(1)})$ | $\text{awt}(A_1^{(2)})$ | $\text{tr}(\textit{chat} \rightarrow \textit{blue})$ | $\text{tr}(\textit{chat} \rightarrow \textit{cat})$ | $\text{tr}(\textit{bleu} \rightarrow \textit{blue})$ | $\text{tr}(\textit{bleu} \rightarrow \textit{cat})$ |
|---|---|---|---|---|---|---|---|
| a. Init | 1/2 | 1/2 | 1 | — | — | — | — |
| b. Update-tr | 1/2 | 1/2 | 1 | 1/2 | 3/2 | 1/2 | 1/2 |
| c. Pnorm-tr | 1/2 | 1/2 | 1 | 1/4 | 3/4 | 1/2 | 1/2 |
| d. Update-awt | 1/8 | 3/8 | 3/4 | 1/4 | 3/4 | 1/2 | 1/2 |
| e. Pnorm-awt | 1/4 | 3/4 | 1 | 1/4 | 3/4 | 1/2 | 1/2 |
| f. Update-tr | 1/4 | 3/4 | 1 | 1/4 | 7/4 | 3/4 | 1/4 |
| g. Pnorm-tr | 1/4 | 3/4 | 1 | 1/8 | 7/8 | 3/4 | 1/4 |