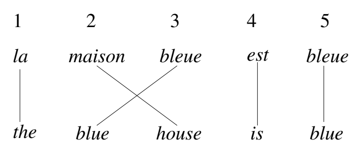
Computation, Information, and Intelligence (ENGRI/CS/INFO/COGST 172), Spring 2007
4/16/07: **Lecture 34 aid — Learning approaches to machine translation**

Topics: One simple approach to the problem of learning a translation dictionary; this approach illustrates a general method for learning when hidden structure is involved, and echoes an approach we've considered in a different context earlier in the course.

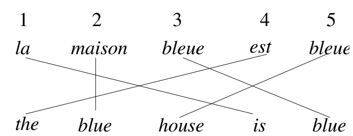
I. Example translation pair

Un program a été mis en application
vs.
And a program has been implemented

II. Example alignments Here is a graphical depiction of two out of the 120 possible alignments[1] for the sentence pair "*la maison bleue est bleue* vs. *the blue house is blue*", where the French sentence is the source.



[A1]



[A2]

Formally, we would denote [A1] by $(1 \leftrightarrow 1, 2 \leftrightarrow 3, 3 \leftrightarrow 2, 4 \leftrightarrow 4, 5 \leftrightarrow 5)$.

III. Notation

- For a sentence pair p, let $\mathrm{Aligns}(p)$ be the set of all possible alignments of the two sentences in p, and let $\mathrm{NumAligns}(p)$ be the size of this set.
- Let $\mathrm{freq}(s \leftrightarrow t, A)$ be the number of times we have the source word s "matched" to the target word t in alignment A. In our example above, we have $\mathrm{freq}(bleue \leftrightarrow blue, [A1]) = 2$.

---

[1] There are only 120 because we only consider "one-to-one and onto" alignments.

IV. An iterative learning algorithm for MT Inspired by IBM's Candide system from the 80s and 90s.

1. *Initialization:* For every sentence pair p, for every alignment A of p, set $\mathrm{Awt}^{(0)}(A) = 1/(\mathrm{NumAligns}(p))$.

Let i be increasing from 1 on, until the translation weights "converge":

2. *Compute temporary translation weights*:
For every source/target word pair (s, t), set $\mathrm{TempTr}(s \to t)$ to $\sum_A \mathrm{freq}(s \leftrightarrow t, A)\mathrm{Awt}^{(i-1)}(A)$.
3. *Get the translation weights by sum-normalizing the temporary ones*:
For each source word s, compute $\mathrm{norm}_s = \sum_{t'} \mathrm{TempTr}(s \to t')$;
then, set each $\mathrm{Tr}^{(i)}(s \to t)$ to $\mathrm{TempTr}(s \to t)/\mathrm{norm}_s$.

4. *Compute temporary alignment weights*: For every alignment $A = (1 \leftrightarrow a(1); 2 \leftrightarrow a(2); \cdots; \ell \leftrightarrow a(\ell))$,
set $\mathrm{TempAwt}(A)$ to $\mathrm{Tr}^{(i)}(s_1 \to t_{a(1)}) \times \mathrm{Tr}^{(i)}(s_2 \to t_{a(2)}) \cdots \times \mathrm{Tr}^{(i)}(s_\ell \to t_{a(\ell)})$
5. *Get the alignment weights by sum-normalizing the temporary ones*: For each pair p,
compute $\mathrm{norm}_p = \sum_{A' \in \mathrm{Aligns}(p)} \mathrm{TempAwt}(A')$;
then, for every A in $\mathrm{Aligns}(p)$, set $\mathrm{Awt}^{(i)}(A)$ to $\mathrm{TempAwt}(A)/\mathrm{norm}_p$.

V. Example[2] partial execution

Suppose we have two sentence pairs, p_1 = "*chat bleu* vs. *blue cat*" and p_2 = "*chat* vs. *cat*". This yields three alignments:

$$A_1 = (1 \leftrightarrow 1; 2 \leftrightarrow 2) \quad \text{(so } \textit{chat} \text{ aligned to } \textit{blue} \text{ in } p_1\text{)}$$
$$A_1' = (1 \leftrightarrow 2; 2 \leftrightarrow 1) \quad \text{(so } \textit{chat} \text{ aligned to } \textit{cat} \text{ in } p_1\text{)}$$
$$A_2 = (1 \leftrightarrow 1) \quad \text{(only one possible choice)}$$

		Alignment weights			Translation weights			
		A_1	A_1'	A_2	*chat* → *blue*	*chat* → *cat*	*bleu* → *blue*	*bleu* → *cat*
a.	Init	1/2	1/2	1	–	–	–	–
b.	TempTr's	“	“	“	1/2	3/2	1/2	1/2
					(insert normalization computations here)			
c.	Tr's	“	“	“	1/4	3/4	1/2	1/2
d.	TempAwt's	1/8	3/8	3/4	“	“	“	“
					(insert normalization computations here)			
e.	Awt's	1/4	3/4	1	“	“	“	“
f.	TempTr's	“	“	“	1/4	7/4	3/4	1/4
					(insert normalization computations here)			
g.	Tr's	“	“	“	1/8	7/8	3/4	1/4

[2]Adapted from Sections 26 ("Chicken and egg") and 27 ("Now for the Magic") of Kevin Knight's (1999) *A Statistical MT Tutorial Workbook* (http://www.isi.edu/natural-language/mt/wkbk.rtf). The tutorial also discusses more advanced models, and is often fairly amusing to boot.