Algorithm Writeup: IBM Model 1 + Diagonal Bias + Symmetrization

Joseph Gonzaludo, David Benjamin

9/18/2025

1 Mathematical Description of the Algorithm

1.1 IBM Model 1

IBM Model 1 defines the conditional probability of a foreign sentence $f = (f_1, \ldots, f_{l_f})$ given an English sentence $e = (e_0, e_1, \ldots, e_{l_e})$ (with $e_0 = \texttt{NULL}$) as

$$P(f \mid e) = \prod_{i=1}^{l_f} \sum_{j=0}^{l_e} t(f_i \mid e_j),$$

where $t(f \mid e)$ is the word translation probability.

Training proceeds via Expectation–Maximization (EM):

- **Initialization:** Set $t(f \mid e)$ uniformly for all (f, e) pairs.
- **E-step:** For each sentence pair (f, e) and each foreign word position i, compute posterior alignment probabilities:

  $$\delta_{i,j} = \frac{t(f_i \mid e_j) \cdot \mathrm{bias}(i, j, l_f, l_e)}{\sum_{j'=0}^{l_e} t(f_i \mid e_{j'}) \cdot \mathrm{bias}(i, j', l_f, l_e)},$$

  where $\mathrm{bias}(i, j, l_f, l_e)$ is the diagonal bias weight (described below).
- **M-step:** Update translation probabilities using expected counts:

  $$t(f \mid e) = \frac{\sum_{(f,e)} \sum_{i: f_i = f} \delta_{i,j(e)}}{\sum_{(f,e)} \sum_{i=1}^{l_f} \delta_{i,j(e)} + \epsilon},$$

  where $j(e)$ denotes the position of English word e in the sentence, and ϵ is a small constant to prevent division by zero.

At inference time, the Viterbi alignment is chosen by

$$a_i = \arg\max_{0 \le j \le l_e} t(f_i \mid e_j) \cdot \mathrm{bias}(i, j, l_f, l_e).$$

1.2 Diagonal Bias Extension

To encourage alignments near the diagonal, we introduce a multiplicative bias term:

$$\mathrm{bias}(i, j, l_f, l_e) = \begin{cases} 1 & \text{if } \lambda \le 0 \\ \exp\Big(-\lambda \left| \frac{i}{\max(1, l_f - 1)} - \frac{j}{\max(1, l_e - 1)} \right|\Big) & \text{otherwise} \end{cases}$$

where $\lambda \ge 0$ controls the strength of the bias. When $\lambda = 0$, no bias is applied (weight $= 1$). The normalization uses $\max(1, l - 1)$ to handle single-word sentences by setting their position to 0. This term penalizes alignments that are far from the diagonal in normalized sentence position space, reflecting the linguistic tendency for word order preservation.

1.3 Numerical Stability

To handle cases where all alignment scores are near zero (below threshold ϵ), the algorithm falls back to a uniform distribution over all possible English alignments for that foreign word.

1.4 Symmetrization

IBM Model 1 is asymmetric: $P(f \mid e)$ is not equivalent to $P(e \mid f)$. To improve robustness, we train two models by running the algorithm twice:

$$A_{f \leftarrow e} \quad \text{(normal training)} \quad \text{and} \quad A_{e \leftarrow f} \quad \text{(reverse training)},$$

where reverse training swaps the roles of foreign and English sentences. The two resulting alignment sets are combined using the *grow-diag* algorithm:

1. Convert reverse alignments to forward coordinates: $(j, i) \to (i, j)$.
2. Compute intersection $I = A_{f \leftarrow e} \cap A_{e \leftarrow f}$ and union $U = A_{f \leftarrow e} \cup A_{e \leftarrow f}$.
3. Initialize alignment set $A = I$.
4. Repeat until no changes occur:

- For each alignment $(i, j) \in A$, check its 4-connected neighbors: $(i \pm 1, j)$ and $(i, j \pm 1)$.
- Add any neighbor $(i', j') \in U$ that is not already in A.

This procedure balances **precision** (from the intersection) with **recall** (from union-based growth), yielding more accurate alignments.