Word Alignment

Ziqian Luo
Carnegie Mellon University
ziqianlu@andrew.cmu.edu

Abstract

The project implements three statistical machine translation models for generating best word alignment from given English and French sentence pairs. The first model is simply a heuristic model, just make an estimation by pointwise mutual information. The second one is IBM model 1, and the third one is a Hidden Markov Model(Stephan Vogel, 1996). The last two models are trained by EM algorithm. By applying intersect technique on HMM, trained on 10000 sentences, the best AER is 0.1976 and the best BLEU is 19.34.

1 Implementation details

Word alignment is an important supporting task for modern statistical machine translation. It is the natural language processing task of identifying translation relationships among the words in a bitext, resulting in a bipartite graph between the two sides of the bitext, with an arc between two words if and only if they are translations of one another. Three models are implemented to explore that. The heuristic model is in Section 1.1, and details about the IBM Model and HMM model is in Section 1.2, Section 1.3 respectively.

1.1 Heuristic aligner

The Heuristic aligner matches up words on the basis of some statistical measure of association,using simple statistics taken directly from the training corpora. In this model, I simply pair each French word f with the English word e for which the ratio C(f,e) / (C(e)*C(f)) is greatest, where C(e) and C(f) denotes total number of occurrence in the corpus for each single English word and each single French word, and C(f*e) denotes total number of occurrence in the corpus for each possible pair of English word and French word.

1.2 IBM Model 1

The IBM model is trained by EM algorithm. For the Expectation Step we need to compute and maximize P(a|e,f), where e is an English sentence and f is an foreign sentence. In this project, it is a French sentence. Applying the chain rule we have:

$$p(a|\mathbf{e},\mathbf{f}) = \frac{p(\mathbf{e},a|\mathbf{f})}{p(\mathbf{e}|\mathbf{f})}$$

For the denominator P(e|f):

$$\begin{aligned} p(\mathbf{e}|\mathbf{f}) &= \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \\ &= \frac{\epsilon}{(l_f+1)^{l_e}} \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \\ &= \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i) \end{aligned}$$

For the numerator P(e,a|f)

$$p(\mathbf{e},a|\mathbf{f}) = \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

For the Maximization Step, we have to collect counts:

$$c(e|f;\mathbf{e},\mathbf{f}) = \frac{t(e|f)}{\sum_{i=0}^{l_f} t(e|f_i)} \sum_{j=1}^{l_e} \delta(e,e_j) \sum_{i=0}^{l_f} \delta(f,f_i)$$

After collecting these counts over a corpus, we can estimate the model:

$$t(e|f;\mathbf{e},\mathbf{f}) = \frac{\sum_{(\mathbf{e},\mathbf{f})} c(e|f;\mathbf{e},\mathbf{f}))}{\sum_{e} \sum_{(\mathbf{e},\mathbf{f})} c(e|f;\mathbf{e},\mathbf{f}))}$$

The pseudocode is shown in Figure 1:

Input: set of sentence pairs (**e**, **f**)
Output: translation prob. $t(e|f)$
initialize $t(e|f)$ uniformly
while not converged **do**
  // initialize
  count$(e|f)$ = 0 **for all** e, f
  total(f) = 0 **for all** f
  for all sentence pairs (**e**,**f**) **do**
    // compute normalization
    for all words e in **e** **do**
      s-total(e) = 0
      for all words f in **f** **do**
        s-total(e) += $t(e|f)$
      end for
    end for
    // collect counts
    for all words e in **e** **do**
      for all words f in **f** **do**
        count$(e|f)$ += $\frac{t(e|f)}{\text{s-total}(e)}$
        total(f) += $\frac{t(e|f)}{\text{s-total}(e)}$
      end for
    end for
  end for
  // estimate probabilities
  for all foreign words f **do**
    for all English words e **do**
      $t(e|f) = \frac{\text{count}(e|f)}{\text{total}(f)}$
    end for
  end for
end while

Figure 1: Pseudocode for IBM MODEL 1

1.3 Hidden Markove Model

The Hidden Markove Model is also trained by EM algorithm.

For the E step, the expected count in each emission matrix cell and transition matrix cell is:

$$d_{f,e}^{(t)}(\theta) = \sum_{i=1}^{I}\sum_{j=1}^{J} \mathbb{1}[f_j = f]\mathbb{1}[a_j = i]\mathbb{1}[e_i = e] \times \\ \times p(a_j = i|\mathbf{f}, \mathbf{e}, \theta^{(t)}, \psi^{(t)})$$

$$d_{k}^{(t)}(\psi) = \sum_{i=1}^{I}\sum_{i'=1}^{I}\sum_{j=1}^{J} \mathbb{1}[a_j = i]\mathbb{1}[|i - i'| = k]\mathbb{1}[a_{j-1} = i'] \times \\ \times p(a_j = i, a_{j-1} = i'|\mathbf{f}, \mathbf{e}, \theta^{(t)}, \psi^{(t)})$$

where

$$p(a_j = i|\mathbf{f}, \mathbf{e}, \theta^{(t)}, \psi^{(t)}) = \frac{\alpha_j^i \beta_j^i}{Z}$$

$$p(a_j = i, a_{j-1} = i'|\mathbf{f}, \mathbf{e}, \theta^{(t)}, \psi^{(t)}) = \frac{\alpha_{j-1}^{i'} \cdot \beta_j^i \cdot \psi_{|i-i'|}^{(t)} \cdot \theta_{f_j, e_i}}{Z}$$

We can compute α and β table by a DP step:

$$\alpha_j^i = \sum_k \alpha_{j-1}^k \psi_{|k-i|} \theta_{e_i, f_j}$$

$$\alpha_0^i = \psi_{|0-i|} \theta_{e_i, f_0}$$

$$\beta_j^i = \sum_k \beta_{j+1}^k \psi_{|k-i|} \theta_{e_k, f_{j+1}}$$

$$\beta_{J-1}^i = 1$$

For the M step, we can update parameters:

$$\theta_{f,e}^{(t+1)} = \frac{d_{f,e}^{(t)}(\theta^{(t)})}{\sum_{\tilde{f}} d_{\tilde{f},e}^{(t)}(\theta^{(t)})}$$

$$\psi_k^{(t+1)} = \frac{d_k(\psi^{(t)})}{\sum_l d_l(\psi^{(t)})}$$

The θ table is emission matrix and ψ table is transition matrix in traditional HMM. In French to English task, θ table has a row size of total English vocabulary size and has a column size of total French vocabulary size, ψ table length is the maximum length of an English sentence.

For backtracking, I used Viterbi algorithm. I used W table to store the max value of each state at each time step and P table to store corresponding back pointers. Here, W and P table is the same size as α and β table. In the French to English task, it has a row size of single English sentence size and has a column size of single French sentence size.

2 Performance

The models are only trained on 10000 sentence pairs.

All the aligners in IBM Model 1 and HMM are intersected aligners of one French to English aligner and one English to French Aligner, which means I record the alignments that both of the two single aligners agree in the intersected aligners.

For the IBM Model 1, the best AER result is 0.2996 on epoch 15 and BLEU is 13.05 on epoch 26

For the HMM, the best AER is 0.1976 on epoch 6 and the best BLEU is 19.34 epoch 10

3 Experiments

3.1 Heuristic Model

The results is shown in Table 1 for 10000, 50000 and 100000 training sentence pairs.

Sentence	AER	BLEU
10000	0.6196	11.95
50000	0.4431	18.08
100000	0.4888	18.87

Table 1: Results on Heuristic Model

The performance is getting higher with more training sentences.

3.2 IBM Model 1

For IBM Model 1 trained on 10000 sentences, the results in shown in table 2.

Epoch	AER	BLEU
10	0.3005	12.27
15	**0.2996**	12.28
25	0.3012	12.98
26	0.3004	**13.05**

Table 2: Results on IBM Model 1

The BLEU might get higher if trained on more epoch but I just stopped when it reaches 13.

3.3 HMM

For Hidden Markov Model trained on 10000 sentences, the results in shown in table 3.

Epoch	AER	BLEU	Training Time
5	0.2009	18.75	4min42s
6	**0.1976**	18.96	5min23s
7	0.1972	19.01	6min35s
8	0.2040	19.34	7min01s
9	0.2049	19.26	7min35s
10	0.2053	**19.34**	8min23s

Table 3: Results on HMM

It also seems to be that BLEU might get higher if trained on more epoch, however, the Alignment Error Rate will increase so I stopped at epoch 10.

4 Discussion

4.1 Future Work

In Hidden Markov Model, the initialization of ψ and θ table might influence the final result. There might exist some improvement on performance if I can use the IBM Model 1 to find a better initialization in Hidden Markov Model and use Heuristic Model to find a better initialization in IBM model 1.

References

Christoph Tillmann Stephan Vogel, Hermann Ney. 1996. Less grammar, more features. *The 16th International Conference on Computational Linguistics*, 2(96):836.