CS 288: Machine Translation

Anting Shen
23566738
antingshen@berkeley.edu

1 Introduction

2 Heuristic Aligner

As a heuristic aligner, I chose to score alignments using ratio of counts. Each French word is aligned to the highest scoring English word. Training size is set at 10,000. At first, I tried scoring $S = c(e, f)/(c(e) \cdot c(f))$, but this performed worse than the baseline aligner. Tweaking the formula to $S = c(e, f)/(c(e) + c(f))$, AER improved to 58.3and BLEU improved to 12.421. Scoring using normalized counts (count divided by total) drops performance to 58.5% AER and 12.208 BLEU, so this change was reverted.

Building this aligner with 10,000 sentences takes 1 second. Increasing training size to 100,000 decreases AER to 46.1%, increases time to 23 seconds, and BLEU to 17.270. Further increasing size to 200,000 decreases AER to 44%, and increases time to 2 minutes 52 seconds. Phrase table took too long to build with this data size so BLEU was not computed.

3 Model 1 Aligner

My first model 1 implementation uses 100 iterations of soft-EM. It uses a counter to store probabilities, and a new counter is created each iteration and is filled using probabilities calculated from values from the previous iteration. Null is handled by appending a null word to the beginning of each English sentence and ignoring null alignments in the output. It is not intersected, trains in 2m31s using 803M memory, and achieves AER of 38.3% and BLEU of 15.795 on the size 10,000 set.

Using an intersected model instead, where the model only predicts alignments that are found by aligners in both directions, precision increased from 0.56 to 0.84, while recall decreased from 0.73 to 0.63. This was expected as the intersected model predicts a sparser alignment. Interestingly, AER and BLEU both decreased to 27.4% and 13.2, respectively. This shows that improvements in AER do not always increase BLEU, perhaps due to the translation system not working as well using sparser alignments. The intersected model also doubles training time to 5m5s to train both models in both directions.

4 HMM Aligner

Then I experimented with aligning using HMM's, then running EM to learn the probabilities of both transitions and emissions. Transitions are simplified as a distance, assuming that probability of jumping a distance is independent of position. All transitions larger than a max of 10 are binned to 10, and a transition matrix is made and normalized for each sentence. $P(f|e)$ is initialized to uniform, and transitions are initialized to a guess of transitions, with most of the weights on a jump distance of 1. For learning weights, probabilities and partial alignments are normalized per french word, giving sentences weight proportional to their length.

4.1 Nulls

At first I used a single null with a hardcoded probability of jumping to and from null of 0.2, but this did allowed paths in the HMM to jump large distances through null without penalty, so a null was used for each position, with a fixed probability of jumping to the null of the corresponding position, and learned transitions out of null. This actually performed badly, with AER of around 45%. Error analysis showed that words at the beginning of sentences had high null alpha values because they have paid fewer null penalties, giving them higher alignments to null and causing errors. Adding null penalty to alphas of the initial state, tuning null probability, and penalizing null alignments all didn't fix the errors. Unfortunately, the final solution I chose was to train a model 1 aligner, then borrow its probabilities for null and fix the null

probability in the HMM to 0 during training.

Since my HMM was intersected, it had very high precision, but somewhat lower recall. As a fix, during prediction, the alpha-beta values for nulls are summed and then discounted by a factor of 0.3 to reduce the number of nulls, as intersection introduces a fair number of nulls already, so the HMM should err on the side of outputting alignments instead of nulls.

4.2 Underflow

The HMM ran into underflow problems during training with longer sentences, since each timestep alpha values would get smaller and eventually exceed that representable by a double. To deal with this, I took the simple approach of not using long sentences for training. This was fine as there were more than enough short sentences that I could not use them all anyway, and there was not a need to learn transitions for very large jumps since they are small enough to be approximated by the last buckets on the transitions. Additionally, the portions trained by model 1 do use all training sentences.

4.3 Results

All this worked fairly well, and achieves an AER of 18.17%, recall of 0.71, and precision of 0.93, which is significantly better than model 1 alone. However, not learning transitions and emissions of null values did harm its AER compared to the reference implementation. Doubling the number of iterations saw a small increase of 0.05% in AER, but also doubles training time so was reverted. BLEU scores weren't able to be accurately calculated, as the testing harness would not let the aligner skip long sentences to build the phrase table so their alignments were inaccurate due to underflow.