Isabel Fan (Pennkey: isafan)
February 20, 2015

CIS 526 Homework 2 Report

Motivation

I updated the default simple stack decoder that was provided to us by including reordering and adding future cost estimation. I also restricted the stack size to contain at most 3000 elements (s = 3000) and there to only be 15 translations per phrase (k = 15), providing the final leaderboard model score/log probability of -1263.84. I choose to iterate on the existing code because I was not able to get my attempts at finite state transducer or bidirectional decoding, and I realized that increasing the stack size seemed to improve on the decoding quality significantly. Reordering was needed because the base code was monotone translation, and offering reordering (some window of it) has provided improvements according to the textbook/Koehn. Future cost was added because it allowed for a much better basis for pruning decisions without becoming too computationally expensive (i.e., computing the exact expected cost of translating the rest of a sentence).

Algorithm Description

As mentioned, I iterated on the simple stack decoder provided in the homework kit. The below mathematical formula was upheld.

$$\begin{aligned}\mathbf{e}^* &= \arg\max_{\mathbf{e}} \sum_{\mathbf{a}} p_{TM}(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) \times p_{LM}(\mathbf{e}) \\ &\approx \arg\max_{\mathbf{e}} \max_{\mathbf{a}} p_{TM}(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) \times p_{LM}(\mathbf{e}).\end{aligned}$$

I included reordering first through the inclusion of a nested loop of all possible input spans for the current French sentence. I also included reordering through the use of a bitmap vector that represented the coverage vector of translated words. I updated the most inner-nested loop of the program by only computing the updated log probability if the range was an uncovered/untranslated interval within the French sentence. With the reordering I had to include the coverage vector as a field within the hypothesis tuple, as well as the input span.

I included the creation of a future cost table, where an entry was added for every possible input span for a given French sentence. When initializing the future cost table, I updated the future cost of a given input span if the phrase existed in the translation model. As the program iterates through every French sentence, I only updated the future cost if the current input span was uncovered/untranslated within the French sentence. The updated cost influences the stack ordering because I had updated the sorted stack within the code from ordering not being based on the log probability but rather the computed future cost of log probability plus the future cost for a given input span.

Results

Inclusion of the reordering and future cost had the most significant improvement in the model score. After that, I experimented with different values of stack size and translations per phrase. My observation was that the stack size seemed to have a greater influence on the model than the translations per phrase, so I tended to alter the stack size value a lot. Ultimately I tested stack sizes of 3000 and 6000, and the former returned a better model score so I used stack size = 3000 in the final result. I believe that perhaps s = 6000 did not

work at an off-basis but that if I increase the stack size further, the model would continue to improve. This is because I had also tested with a stack size of 160, for example, and it performed worse than using a stack size of 150 while keeping all other parameters the same. Then when I increased the stack size further to 200, the performance improved. If I had time, I would have still experimented with larger stack size (perhaps 12,000) because I believe it would improve, however it would take a very long time given that a stack size of 6000 took roughly two hours.

After a translations per phrase value of 20 performed worse than 15, I stuck with k = 15.

Description	Stack Size (-s)	Translations per phrase (-k)	Model Score
Simple stack decoder (default)	-	-	-1439.87
Default with reordering + future cost	100	-	-1344.83
Default with reordering + future cost	100	5	-1292.37
Default with reordering + future cost	100	10	-1290.22
Default with reordering + future cost	100	15	-1289.52
Default with reordering + future cost	100	20	-1290.32
Default with reordering + future cost	150	15	-1281.60
Default with reordering + future cost	155	15	-1282.01
Default with reordering + future cost	160	15	-1284.30
Default with reordering + future cost	200	15	-1282.05
Fixed/refactored code (I realized my indices were incorrect)	150	15	-1279.69
Default with reordering + future cost	1500	15	-1275.46
Default with reordering + future cost (final)	3000	15	-1263.84
Default with reordering + future cost	6000	15	-1264.32