CMPT 413/825 Assignment 4

Turash, Luiz, Jin, Justin

November 23, 2017

1 Brief Description of the Algorithm

The algorithm implemented in assignment 4 for the translation decoding task is based on the decoding algorithm for phrase-based models in the paper by Michael Collins called "Phrase-Based Translation Models."

Briefly, the algorithm uses phrase-based lexicons which are a set of lexical tuples of the form (f, e, g) where f is a sequence of one or more words of the foreign word (in our case, the language of interest is French), e is the sequence of one or words of English, and g is the scoring which is a real number.

We can then learn a phrase-based lexicon by training IBM Model 2, by using the expectation-maximization algorithm based on the alignment task from assignment 3. From this, we derive an alignment matrix for each French-English sentence pair.

After finding all consistent lexical tuples of sequence of French words and sequence of English words, we calculate the score g,

$$g(e, f) = log\frac{count(e, f)}{count(e)}$$

where f is the sequence of French words, and e is the sequence of English words.

A phrase is defined as a 3-tuple (s, t, e) where $1 \leq s \leq t \leq n$, and s and t denote the start and end word index, respectively, of the French sentence $x_1, x_2, ..., x_n$ of length n. e is defined as the translation of the ordered French word sequence $x_s, ..., x_t$ to English.

We found the set of all derivations, denoted $\Upsilon(x)$, as the set of all valid derivations for an input sentence x, which in our case is in French. It is the set of sequences of phrases, say $p_1 p_2 \cdots p_n$ where all p_i's are lexical tuples defined above, each word in x is translated once, and the distance between lexical tuples are from each other is limited by the distortion parameter η.

After finding the set of all valid derivations $\Upsilon(x)$, we score each derivation in the set$\Upsilon(x)$ for all sentences x by taking the arg max,

$$\arg\max_{y \in \Upsilon(x)} f(y)$$

where

$$f(y) = h(e(y)) + \sum_{k=1}^{L} g(p_k) + \sum_{k=1}^{L-1} \eta \times |t(p_k) + 1 - s(p_{k+1})|$$

2 Pseudocode of the Algorithm

```
Data: French sentences
Result: English translation using the model
for f in French do
    Initialize hypothesis as binary tuple;
    Initialize stacks, with first entry as hypothesis;
    for i, s in enumerate(stacks) do
        for h in sort(s.itervalues() by -h.logprob do
            for j from i+1 to length(f)+1 do
                if f[i:j] in tm then
                    for phrase in tm[f[i:j]] do
                        s = update_stacks(j, h, phrase, f[i:j], j == len(f),
                         stacks, -1)
                    end
                end
            end
            for j in range i+2 to length(f) do
                if  (j-5) < 4 then
                    break
                end
                newF = list(f[:]);
                swap newF[i] and newF[j];
                tnewF = tuple(newF);
                if tNewF[i:j] in tm then
                    for phrase in tm[tNewF[i:j]] do
                        stacks = update_stacks(j, h, phrase, tNewF[i:j], j
                         == len(f), stacks, i-j)
                    end
                end
            end
            for j in range i+3 to length(f) do
                if (j-5) > 4 then
                    break
                end
                newF = list(f[:]);
                swap newF[i], newF[j];
                perms = permutation(newF[i:i+3]);
                for perm in enumerate(perms) do
                    tNewF = tuple(list(perm) + newF[i+3:length(newF]);
                    if tNewF[i:j] in tm then
                        for phrase in tm[tNewF[i:j]] do
                            stacks = update_stacks(j, h, phrase, tNewF[i:j],
                             j == len(f), stacks, i-j)
                        end
                    end
                end
            end
        end
    end
    winner = max(stacks[-1].itervalues(), f(h) = h.logprob);
    w = improve(winner);
end
```