Homework #5: CMPT-413

Anoop Sarkar
http://www.cs.sfu.ca/~anoop

(1) **Human Translation**

NASA's latest mission to Mars has found some strange tablets. One tablet seems to be a kind of Rosetta stone which has translations from a language we will call MARTIAN-A (sentences 1a to 12a below) to another language we will call MARTIAN-B (sentences 1b to 12b below). The ASCII transcription of the alien script on the Rosetta tablet is given below:

```
1a. ok'sifar zvau hu .

1b. at'sifar somuds geyu .

2a. ok'anko ok'sifar myi pell hu .

2b. at'anko at'sifar ashi erder geyu .

3a. oprashyo hu qebb yuzvo oxloyzo .

3b. diza geyu isvat iwla pown .

4a. ok'sifar myi rig bzayr zu .

4b. at'sifar keerat ashi parq up .

5a. yux druh qebb stovokor .

5b. diza viodaws pai shun .

6a. ked hu qebb zu stovokor .

6b. dimbe geyu keerat pai shun .

7a. ked druh zvau ked hu qebb pnah .

7b. dimbe viodaws somuds dimbe geyu iwla woq .

8a. ked bzayr myi pell eoq .

8b. gakh up ashi erder kvig .

9a. yux eoq qebb zada ok'nefos .

9b. diza kvig pai goli at'nefos .

10a. ked amn eoq kin oxloyzo hom .

10b. dimbe kvig baz iluh ejuo pown .

11a. ked eoq tazih yuzvo kin dabal'ok .

11b. dimbe kvig isvat iluh dabal'at .

12a. ked mina eoq qebb yuzvo amn .

12b. dimbe kvig zeg isvat iwla baz .
```

We would like to create a translation from the source language which we will take to be MARTIAN-B and produce output in the target language which will be MARTIAN-A. Due to severe budget cutbacks at NASA, decryption of these tablets has fallen to people like you. In this question, you should try to solve this task by hand to get some insight into the process of translation.

a. Use the above translations to produce a translation dictionary. For each word in MARTIAN-A provide an equivalent word in MARTIAN-B. If a word in MARTIAN-A has no equivalent in MARTIAN-B then put the entry "(none)" in the dictionary.

b. Using your translation dictionary, provide a word for word translation for the following MARTIAN-B sentences on a new tablet which was found near the Rosetta tablet.

```
13b. gakh up ashi woq pown goli at'nefos .

14b. diza kvig zeg isvat iluh ejuo .

15b. dimbe geyu pai shun hunslob at'anko .
```

The MARTIAN-A sentences you produce will probably appear to be in a different word order from the MARTIAN-A sentences you observed on the Rosetta tablet. Some words might be unseen and so seemingly untranslatable. In those cases insert the word `?` for the unseen word.

c. The word for word translation can be improved with additional knowledge about MARTIAN-A word order. Luckily another tablet containing some MARTIAN-A sentences (untranslated) was found on the dusty plains of Mars. Use these MARTIAN-A sentences in order to find the most plausible word order for the MARTIAN-A sentences translated from MARTIAN-B sentences in (1b).

```
ok'anko myi oxloyzo druh .
yux mina eoq esky oxloyzo pnah .
ok'anko yolk stovokor koos oprashyo pnah zada ok'nefos yun zu kin hom .
ked hom qebb koos ok'anko .
ok'sifar zvau hu .
ok'anko ok'sifar
myi pell hu .
oprashyo hu qebb yuzvo oxloyzo .
ok'sifar myi rig bzayr zu .
yux druh qebb stovokor .
ked hu qebb zu stovokor .
ked bzayr myi pell eoq .
ked druh zvau ked hu qebb pnah .
yux eoq qebb zada ok'nefos .
ked amn eoq kin oxloyzo hom .
ked eoq tazih yuzvo kin dabal'ok .
ked mina eoq qebb yuzvo amn .
```

Using this additional MARTIAN-A text you can even find a translation for words that are missing from the translation dictionary (although this might be hard to implement in a program, cases that were previously translated as `?` can be translated by manual inspection of the above MARTIAN-A text).

(2) † **Word Alignment for Machine Translation**

Aligning words is a key task in machine translation. We start with a large corpus of aligned sentences called a parallel corpus. For example, we might have the following sentence pair from the Canadian Hansards, which are the published proceedings of the Canadian parliament:

le droit de permis passe donc de $ 25 à $ 500 .
we see the licence fee going up from $ 25 to $ 500 .

The task in this question is to find the word alignments automatically. For example, given the above sentence pair, your program should output the following alignment that uses the word indices from French to English:

```
0-2 1-4 3-3 4-5 4-6 5-7 7-8 8-9 9-10 10-11 11-12 12-13
```

This corresponds to an alignment of the words as shown in the following table:

0 le	2 the
1 droit	4 fee
2 de	
3 permis	3 license
4 passe	5 going
4 passe	6 up
5 donc	7 from
6 de	
7 $	8 $
8 25	9 25
9 à	10 to
10 10 $	11 $
11 500	12 500
12 .	13 .

The file `align_stub.py` contains a complete but very simple alignment algorithm. For every word, it computes the set of sentences that the word appears in. Intuititvely, word pairs that appear in similar sets of sentences are likely to be translations. Our aligner first computes the similarity of these sets with Dice's coefficient. Given the co-occurence count $C(e, f)$ of words e and f in the parallel corpus, Dice's coefficient for each pair of words e_i, f_j is:

$$\delta(i, j) = \frac{2 \times C(e_i, f_j)}{C(e_i) \cdot C(f_j)}$$

The provided code will align any word pair e_i, f_j with a coefficient $\delta(i, j)$ over 0.5. You can experiment with different thresholds using `-t value`.

Just like `align_stub.py` your code should implement at least two command line arguments:

```
python align.py [options]

options are:

-p DIR/PREFIX   prefix for parallel data, Defaults: DIR=../ PREFIX=hansards
-n NUMBER       number of training examples to use, Default: n = sys.maxint
```

For instance, I should be able to run your submission as follows:

```
cd answer
python align.py -p ../hansards -n 1000
```

where the files `../hansards.e` and `../hansards.f` are assumed with the same number of lines each (one sentence per line).

The following pseudo-code provides an algorithm that can learn a translation probability distribution $t(e|f)$ from a set of previously translated sentences. $t(e|f)$ is the probability of translating a given word f in the source language as the word e in the target language.

initialize $t(e|f)$ uniformly
do
    set $c(e|f) = 0$ for all words e, f
    set total$(f) = 0$ for all f
    for all sentence pairs $(\mathbf{e}, \mathbf{f})$ in the given translations
        for all word types e in $\mathbf{e}$
            n_e = count of e in $\mathbf{e}$
            total $= 0$
            for all word types f in $\mathbf{f}$
                total += $t(e|f) \cdot n_e$
            for all word types f in $\mathbf{f}$
                n_f = count of f in $\mathbf{f}$
                rhs = $t(e|f) \cdot n_e \cdot n_f$ / total
                $c(e|f)$ += rhs
                total(f) += rhs
    for each f, e
        $t(e|f) = c(e|f)$ / total(f)
until convergence (or set to fixed number of iterations)

By initializing uniformly, we are stating that each target word e is equally likely to be a translation for given word f. See more about convergence below.

The psuedo-code given above is a very simple statistical word alignment model that is called (for historical reasons) IBM Model 1.

This pseudo-code learns values for probability $t(e|f)$. It is based on a model of sentence translation that is based only on word to word translation probabilities. We define the translation of a source sentence $\mathbf{f}_s = (f_1, \ldots, f_{l_f})$ to a target sentence $\mathbf{e}_s = (e_1, \ldots, e_{l_e})$, with an alignment of each e_j to to some f_i according to an alignment function $a : j \to i$.

$$\Pr(\mathbf{e}_s, a|\mathbf{f}_s) = \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

Assume we have a three word French sentence ($l_f = 3$) sentence-aligned to a three word English sentence ($l_e = 3$). If the alignment function is defined as $a : \{0 \to 0, 1 \to 1, 2 \to 2\}$ we can derive the probability of this sentence pair alignment to be

$$\Pr(\mathbf{e}_s, a|\mathbf{f}_s) = t(e_0|f_0) \cdot t(e_1|f_1) \cdot t(e_2|f_2)$$

In this simplistic model, we allow any alignment function that maps any word in the source sentence to any word in the target sentence (no matter how far apart they are). However, the alignments are not provided to us, so:

$$\begin{aligned}
\Pr(\mathbf{e}_s|\mathbf{f}_s, t) &= \sum_a \Pr(\mathbf{e}_s, a|\mathbf{f}_s, t) \\
&= \sum_{a(0)=1}^{l_f} \cdots \sum_{a(l_e)=1}^{l_f} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \quad \text{(this computes all possible alignments)} \\
&= \prod_{j=1}^{l_e} \sum_{i=1}^{l_f} t(e_j|f_i) \quad \text{(converts } l_f^{l_e} \text{ terms into } l_f \cdot l_e \text{ terms)}
\end{aligned}$$

The pseudo-code provided implements the EM algorithm which learns the parameters $t(\cdot \mid \cdot)$ that maximize the log-likelihood of the training data:

$$L(t) = \operatorname*{argmax}_t \sum_s \log \Pr(\mathbf{e}_s|\mathbf{f}_s, t)$$

Check for convergence by checking if the value of $L(t)$ does not change much from the previous iteration (difference from previous iteration is less than 10^{-4}, for example).

The best alignment to a target sentence in this model is obtained by simply finding the best translation for each word in the source sentence. For each word f_j in the source sentence the best alignment is given by:

$$a_j = \operatorname*{argmax}_i t(e_i|f_j)$$

Produce the output alignment for each sentence pair in the parallel corpus in the format j-a_j providing an alignment a_j for each word j in the French sentence.

An IBM Model 1 implementation should give you the following alignment error rate (AER) without the `-n 1000` option so that it uses the entire training data.

```
Precision = 0.574202
Recall = 0.736686
AER = 0.373938
```

You may get a slightly different number based on implementation details. Once you have implemented IBM Model 1 you should experiment further with more sophisticated alignment models. Here are some ideas:

- Do a random initialization instead of uniform initialization and do random restarts to find better values of $L(t)$.
- Implement a model that can align some French words to NULL to delete them.
- Train a French-English model and a English-French model and combine their predictions.
- Implement an HMM alignment model.
- Implement a model that is aware of morphology so that stems share alignment probability across word forms.
- Implement a model that prefers to align words close to the diagonal.