ITCS 4111/5111 Introduction to Natural Language Processing

Assignment 6 **Due: December 1st, 2017 11:59 pm**

Problem 1 (40 points):

The first task is to implement the direct MT system, where you will need to collect your own data.

Language

First you will need to choose a language. You will be translating **from** this other language ('F') **to** English ('E'). You do not need to be fluent in language F, but you should know it well enough to be able to roughly assess the quality of the translation.

Data

To build your system, you will need a small working corpus on which you can test it. Your first job is to create that corpus. It should be 15 sentences. **Don't write the sentences yourself**; take real sentences from a source in your chosen language, such as a newspaper, a novel, a web site, etc. You can have sentences from different sources. All the sources should be included in your write-up.

Dev-test split

From this point on, leave out about 5 sentence pairs (F-E) for testing, and work only with the rest of the pairs when developing your system. The set of 5 sentences is your test set; the set of 10 sentences is your dev set, which you use both as a source of insights into the translation problem, and to produce evaluations during the development process. At the end, you will come back to the test set to see if the system you developed generalizes well.

Don't look at the test set while you're developing! The reason for this is that, the more you know about your test set, the more this allows you to tailor your system to perform well on that set, and that's not a fair evaluation. It's a violation of the honor code to look at the test set after you built your dictionary (which should be your very first step in building the MT system).

Dictionary

We're assuming a closed vocabulary system, so you will have access to a dictionary for all the words in the working corpus. (A complex real-world system would work with an open-vocabulary assumption, and deal with new words on the fly.)

Create a bilingual E-F (English / Language-F) dictionary for each word in your working corpus. It's difficult to get a good downloadable dictionary, so do this using a web-based or print English-X dictionary (Here's a web-based dictionary for several languages. Google Translate often works quite well for word to word translation, too.)

Don't try to work directly with the entire dictionary. Rather, create a little dictionary file that has just the words in your working corpus and the corresponding translation for English. Note that, if you have more than one translation for some word, you can put all the translations in the dictionary, and you will need a heuristic to choose the correct one in context when you are translating a sentence. (That could just be using the most frequent translation, but you can use information about the source sentence, a language model, etc.) **Handpicking the correct translations in advance is not allowed.**

Translation system

Now write code (Python, as usual) to implement the following "**Direct MT**" system:

- Use your bilingual dictionary to translate each word from Language F into English.
- Obtain any annotation you want on your sentences -- word tokenization, lemmatization, POS-tagging, parsing, word sense disambiguation, anything you need. Do not write your own tools for this; you can look for a toolkit in your favorite language, and we can help with that too. (Note that no specific type of annotation is required, and you will not be graded on the type of annotation you choose. *It is using these tools intelligently that matters*.)
- Now write code for 6-10 **pre- or post-processing strategies** to improve the baseline translations from the direct method. Good strategies will be ones that generalize well and produce significant improvements on the translation, making it look more like real English. Leverage your knowledge of the languages, and try to spot patterns in your dev set.

Note that we would ask you to take the following **iterative approach** to think of and report the strategies you propose. For each strategy, you should:

- Observe the translations of your current system (with 0 or more strategies you already implemented) on the dev set, and identify problems
- Come up with **one strategy** that would alleviate the problem you identified
- Implement that strategy and evaluate its performance

Problem 2 (40 points):

IBM Model 1

In this part of the homework, will need to implement the IBM Model 1 algorithm, and train it on the Europarl corpus that is provided. This corpus contains sentences extracted from the proceedings of the European Parliament, and are translated to various European languages. In this task, we will ask you to choose either Spainish or French (not both), and then train a statistical MT system to translate Spanish (es) or French (fr) to English (en). No prior knowledge of either Spanish nor French should be necessary to finish the MT system, but you might find that helpful when doing error analysis.

We have selected a subset of the corpus for this assignment, which contains ~10,000 Spanish-English or French-English sentence pairs. The development set and test set you will be using are subsets of an actual dev set used in academic research, which contains news articles in the three languages stated above.

Once you are finished with your statistical MT system, use the dev set to evaluate your system's performance in terms of BLEU scores. We have implemented BLEU score computations for you so you wouldn't need to worry about that. To evaluate your translation BLEU scores, use the `bleu_score.py` script we provided in the archive file. To evaluate your MT system, dump your translations to a file, and run from the command line (*note the ordering of the two files*):

```
python bleu_score.py <the-reference-file> <your-output-file>
```

For example, your command might be

```
python bleu_score.py dev/newstest2012.en my-awesome-dev-trans.en
```

Improve the System

After you have your MT system running, you will probably see that the translation quality is not as good as you expected. In fact, the translation quality might not be much better than a direct MT system (think about why). Next, your job is to come up with at least 1 strategy to improve your statistical MT system, implement it, and fine-tune it on the dev set. A few ideas for this improvement might be (but not limited):

- incorporating a language model
- rearranging translations based on part-of-speech
- phrase extraction

Again, do not look at the test set before actually testing the performance of your system!

Evaluation and Analysis (Both Systems)

Testing

When you're finished with your translation system run it on the test set. Your code will need to run the baseline system (direct MT with dictionary and IBM Model 1), then produce whatever annotations you need for your improvement strategies, then execute those strategies.

Note that for some previous assignments, we usually take care of this and run your code on a test set that you haven't had access to. The logistics for this assignment is different: you will run the system on the test set yourself. We count on your honesty for this step: **do not use the test set while you are developing, and do not go back to developing after you evaluate your system on the test set.**

Error Analysis

After you're all done and have produced your translations for the test set, there will inevitably still be errors. (In fact, you probably still have errors in the dev set as well -- MT is hard!) Now is the time to think deeply about those errors and how to make the system better.

Make sure you leave plenty of time for this: error analysis is one of the most important stages in the development of complex systems! Where are things going wrong: maybe there's ambiguity in the source sentences? Or perhaps idiomatic meanings? How is the fluency of the output? You should not only identify the errors in your output, but also think carefully about what aspects of language make the problem difficult; what simplifying assumptions in your approach fail to tackle those aspects of language; what information would be needed to avoid the errors; and what might be some ways of getting that information. Machine translation is an open area of research, so of course you are likely to run into errors that are very difficult to avoid. What's important in this assignment is that you understand why those errors happen.

Problem 3 (20 points):**

Google it!

Next, run the sentences in your test set through Google Translate and discuss any errors that Google makes. Where does Google do better? Are there places where your implementation does better? Why?

For direct MT problem, you should compare all 5 sentences in your test set. For statistical MT problem, picking 5 sentences at random from your test set would suffice.

Problem 4 (40 points) (EXTRA CREDIT PROBLEM – ONLY ALLOWED FOR THOSE WHO SUBMITTED PRIOR ASSIGNMENTS LATE AND WANT TO MAKE UP FOR THE 0 POINTS ASSIGNED FOR THOSE, DO NOT ATTEMPT THIS IF YOU DID NOT RECEIVE A 0 IN ANY OF THE PRIOR ASSIGNMENTS):

In this task, you will work on the other language you did not choose in Problem 2. That is, If you chose Spanish for Problem 2, then repeat Problem 2 and 3 for French (and vice versa).

What to submit

Code and write-up. Be precise and concise in your write-up.

Note: ITCS 5111 students will be graded out of a 100 points total for all problems in this assignment. ITCS 4111 students will be graded out of 80 points total for all problems in this assignment, except for those marked with **.