EN.600.468 Statistical Machine Translation
Final Project:

Reordering Assignment Assembly

Presented To
Phillip Koehn

By
Sarika Halarnakar - shalarn1
Sindhuula Selvaraju - sselvar4
Johns Hopkins University
10 May 2016

Executive Summary

Introduction

For our final project we have created a machine translation homework assignment on reordering. Languages differ in many ways based on word order alone. Word order defines relationships between words. Some classifications can be made by naming the typical order of subject, verb, and object, but there are also many other differences in word orders such as Adjective/Noun placement, Noun/Relative Clause placement, and in some languages the word order is not fixed. In machine translation, it is important to preserve word order because the meaning of a sentence can change depending on word order. For example, the difference between "I had my house cleaned" and "I had cleaned my house" is an important distinction to make as the word ordering changes the meaning of the sentence.

We have assembled a basic implementation that provides baseline metrics. Students will need to improve these metrics in their own implementations. Similar to our previous homework assignments, we will provide some implementation suggestions to help improvement.

Project Outline

We have provided a basic grammar ruleset that students will have to add to in order to build a fully comprehensive ruleset and improve their results. We have implemented our project using python.

Relevant Papers

1. Pre-ordering of phrase-based machine translation input in translation workflow
2. A Word Reordering Model for Improved Machine Translation
3. Automatically Learning Source-side Reordering Rules for Large Scale Machine Translation

Experimental Design

Provided Files: Non-SVO Spanish sentences sourced from the Europarl corpus, their POS tags, and their English equivalents from the Europarl corpus.

Input: Spanish sentences in non-SVO order sourced from the Europarl corpus.

Output: Correctly ordered Spanish sentences.

User Scoring Rubric: We have built an evaluation program, similar to the one provided to us for our other homeworks. The user will use this program with their output to check if their accuracy is better than that of the program given by us.

Evaluation Measures: Our evaluate script calculates the accuracy of the reordering model by comparing the generated word ordering with the correct word ordering.

Table of Contents

Contents of the Repository

- **final_submission:**
  - dictionary.py: This program will form the dictionary to map English words to their Spanish translations.
  - evaluate.py*: This is the main evaluation function
  - extract.py**: This program was used to extract sentences from the data files
  - reorder.py: This is the baseline program
  - translate.py*: This program attempts to literally translate words in the English sentence to Spanish
  - **data**
    - dev.en*: English dataset for development data
    - dev.es: Spanish dataset for development data
    - dev.pos.txt: POS tags for the dev.es sentences in txt format
    - en-es-enwiktionary.txt: The dictionary file from which the dictionary is formed
    - postags.xlsx: Excel workbook with sheets containing POS tags for train,test and dev datasets
    - test.en*: English dataset for test data
    - test.es: Spanish dataset for test data
    - test.pos.txt: POS tags for the test.es sentences in txt format
    - train.en: English dataset for training data
    - train.es: Spanish dataset for training data
    - train.pos.txt: POS tags for the train.es sentences in txt format
  - MaltParserInput**: Can be used to provide input for extract.py
  - sample.xlsx**: Can be used to provide input for extract.py
- **extras:** This folder contains all the intermediate work done to create the assignment. These files are not required for the homework

** : These files need not be provided in the homework

* : These files should be hidden in the homework

Project Layout

Goal of Assignment

For our final project, we created a reordering assignment for future Machine Translation students. In machine translation, it is important to preserve word order because the meaning of a sentence can change depending on word order. Languages differ in many ways based on word order alone. Word order defines relationships between words. Some classifications can be made by naming the typical order of subject, verb, and object, but there are also many other differences in word orders such as Adjective/Noun placement, Noun/Relative Clause placement, and in some languages the word order is not fixed.

For example, compare the difference between

"I had my house cleaned"

and

"I had cleaned my house"

There is an important distinction to make between the two sentences as the word ordering changes the meaning of the sentence.

Reordering is the process of taking input in Spanish:

Se formó un Comité de Selección.

And finding its best Spanish reordering with the correct English word order:

Un Comité de Selección se formó .

Which would make translating to the English ordered sentence easier:

A Selection Committee was formed.

The goal of our assignment is to reorder Spanish sentences to match the word order of its English equivalent

.

Our Approach

To reorder, we need a model of reordered Spanish sentences conditioned on the original Spanish sentence. In our project, we have provided a basic rule-based reordering system that simply reorders the subject, object, and verb clauses from the original Spanish sentence into SVO order.

Our model assumes that the Spanish sentence can be evenly split into subject, verb, and object clauses. This method parses input sentences, groups the words into their respective clauses, and reorders the clauses using a set of hand- crafted rules to get SVO-like sentences.

The students' challenge is to improve our system to account for specific word by word reorderings such as noun/adjective placements, relative clauses, adpositions etc. for the Spanish dev dataset. They will need to generate the most probable Spanish reordered sentence with the most probable English word order.

However, rule-based reordering is language specific, so a linguist has to find the best ruleset for every language pair. Though there are successful rulesets for many language pairs, if we could completely reorder the words in input sentences by preprocessing to match the word order of the target language, we would be able to greatly reduce the computational cost of machine translation systems. To get full credit, the student **must** additionally experiment with another reordering algorithm.

Any permutation of phrases is a valid translation, so we strongly suggest having a clear training algorithm to model the data. They can use reordering limits as described in the textbook (Chapter 6) and lecture slides. Some things they might try:

- Automatically learning source-side reordering rules
- Pre-ordering of phrase-based machine translation.
- Chunk-Based Verb Reordering.

Usage

Clone the repository:

```
git clone https://github.com/sindhuula/MT_2016/finalproject.git
```

Under the `reorder` directory, you now have simple reorderer. Test it out!

```
python reorder.py > data/train.output
```

This creates the file `output` with translations of `data/train.es`. You can compute the accuracy using `evaluate`.

```
python evaluate -t train < train.output
```

Evaluation

Our evaluate script calculates the accuracy of the reordering model by comparing the generated word ordering with the correct word ordering. Each sentence is evaluated out of 100 points. 30% of the accuracy each sentence is awarded for generating an SVO form sentence. We have provided the preliminary code that already passes this test, however students may feel free to improve our method. The remaining 70% is awarded on a word by word ordering comparison with the correct Spanish reordering. So for example, if the following sentence is the correct Spanish reordering:

El blanco perro ladra a me
S V O

And the original sentence is:

A me ladra el perro blanco
O V S

And the generated reordering is

El perro blanco ladra a me
S V O

Then the computed accuracy would be 30 for the correct SVO ordering and (4/6)*70 for the word by word ordering, totaling 76.67%.

The training and development sets given contain the original Spanish and English sentences (train.es/train.en, dev.es/dev.en). We have also provided a POS tag file for each Spanish dataset (Documentation).

Our Process

To create this assignment we first needed to assemble the data, which ended up being one of the most challenging aspects of our project. We first needed to find non-SVO Spanish sentences and their English SVO equivalents. We decided to run a dependency parser on selected sentences from the Spanish Europarl corpus. We kept a copy of these selected sentences and their English Europarl equivalents in MaltParserInputs to later reference for our data. We then ran our extract.py script, which generates our data files, on the output tags of the dependency parser. Our extraction method finds non-SVO sentences based on the dependency tags, then locates those Spanish sentences and its equivalent English sentence by sentence number and writes them to our data files.

To reorder, we ran the reorder.py script which reorders the output of our extract.py into SVO ordered Spanish sentences by grouping the words into their respective subject/verb/object clauses, and reordering the clauses using a set of hand-crafted rules.

Our evaluation method uses the output of the reorder.py script and compares each reordered sentence with the literal translation of the English sentence. We use translate.py to get the literal translation, dictionary.py to map english words to all it's Spanish translations. We tried many online methods(PyDictionary, Google Translate API) to do a word by word translation but ended up having to use an offline dictionary that does a word by word translation of the stemmed or lemmatized English word. The translation also tries to take care of bigrams/trigrams in english mapping to unigrams/bigrams/trigrams in spanish.

Complications

Due to the way we decided to generate data, we had to ensure that each line of the input Spanish files matched each sentence detected by the dependency tracker, and each line of the English file. This was timely and tedious process as the dependency parser would detect more sentences than the number of lines in the files, so we first had to change the Spanish file lines to match the sentences detected. Then, we would need to compare the Spanish file to the English file and match up the sentences, and even change some of the English sentences to provide a better translation for our evaluator. Due to the nature of this process, we limited our dependency parser input data to 1000 lines, which generally produced 100-200 non-SVO sentences. We realized that the dependency parser would not identify the root of the sentence as a verb or subject and since our extractor only strictly selects sentences with at least one

subject, one verb, and one object, we had to omit many sentences from the input. We also realized that the dependency parser only identified auxiliary verbs, so we had to modify the code to look at the POS tagging for verbs instead of the dependency tag. We would have prefered to use a more accurate parallel corpus. We attempted to manually generate our parallel data, but we did not think we would be able to produce a sufficient number of parallel sentences given the time constraints and our limited knowledge of the Spanish language, so we opted to use the Europarl corpus.

Also when using the translate.py program to find literal translations of the English sentences to know the exact word alignments we found that many online dictionaries do not have free apis or they have a daily quota so after attempting to use various online dictionaries - Google Translate, Word Reference, Merriam Webster to name a few, we decided to use an offline dictionary. However, one major set back is that the dictionary we were using was incomplete(though it has many duplicates) and a lot of new words had to be added. This process is still not fully complete due to this reason and may result in some words missing in the translated sentence. We would have prefered to use an online dictionary as it would have provided translations for conjugations and morphed words and with more access to resources for online dictionaries, we would have been able to provide a full literal translation in our evaluation method .

We also tried having other evaluation criteria like checking with the POS tags. But we were unable to do so because of several factors like NLTK pos_tag() is not very accurate, we did not have sufficient training data to model a new tagger and some approaches would’ve just over simplified the assignment.

Conclusion

Reordering is a tricky problem to solve because languages can have many different structures that are equivalent and syntactically correct. Using hand-written rules is tedious because these rules vary from language to language, and may not even hold true for all sentences in a particular language pair. Metrics for evaluation also prove to be difficult to identify because there are many ways to construct a correct sentence in different languages. One of the most challenging aspects is to be able to evaluate the outputs without actually giving away too much information on how to actually do the assignment. Given more time and resources we would like to extend the assignment further by adding our own POS tagger in order to increase the accuracy of evaluation.

References

1. European Parliament Proceedings Parallel Corpus 1996-2011
2. MaltParser Web Service
3. Pre-ordering of phrase-based machine translation input in translation workflow
4. A Word Reordering Model for Improved Machine Translation
5. Automatically Learning Source-side Reordering Rules for Large Scale Machine Translation
6. Reordering Lecture Notes