11-763 Homework 1
Due: 9/29/2015

1 Introduction

The purpose of this homework assignment is to gain familiarity with a structured prediction task - specifically sequence labelling - and to develop a codebase that you can use in the future. You will implement a named entity recognition (NER) system and evaluate it on the CoNLL 2003 shared task.

2 Task

Your task is to implement a BIO tagger for four kinds of entities: people, locations, organizations, and names of other miscellaneous entities. We are providing the weights file for a fully trained tagger, and you will need to write code for the Viterbi algorithm, feature functions, and support code. Given the input sentence x, the job of the tagger is to find the highest scoring tag sequence $\hat{y}$ under the linear model:

$$\hat{y} = \arg\max_{y} \vec{w} \cdot \vec{f}(x, y)$$

You do not need to write code for training, but you still get the joy of producing a fully functioning tagger!

3 Data format

The data is in four columns separated by a space. Each token in the sentence has been put on a separate line, and the sentences are separated by a blank line. The first column is the token, and second is the part-of-speech tag, the third is a syntactic chuck tag (which we will not use), and the fourth is the named entity (NE) tag. The NE tags use a modified BIO tagging scheme, where 'O' denotes outside, 'I' denotes inside or begin, and 'B' is only used for the start of a new NE in the case of two identically tagged NEs in a row. Here is an example sentence:

```
France NNP I-NP I-LOC
and CC I-NP O
Britain NNP I-NP I-LOC
backed VBD I-VP O
Franz NNP I-NP I-PER
```

```
Fischler NNP I-NP I-PER
's POS B-NP O
proposal NN I-NP O
.  .  O O
```

4 Tagging

Your tagger should take as input a file in this format, ignore the last column (don't cheat!), and produce a new file with a fifth column containing the tags produced by your tagger. We will only be checking to see if your code produces the same tags as the reference tagger. We are providing two datafiles (with four columns): 'dev' which contains the output of the reference tagger on some data which you can use for debugging your code, and 'test' which contains human annotated tags on some other data. We will compare your tagger to the reference tagger on 'test'.

Do not worry about matching the output of the reference tagger exactly.

We have tried to describe the features as thoroughly as possible, but there may be implementation details that you do not have to duplicate.

You can check the performance of your tagger by running the CoNLL 2003 evaluation script. For reference, our tagger obtains an F1 measure of 79.8% on 'test'. Compared to the reference tagger (i.e. evaluating on 'dev'), your tagger should have an F1 measure of $> 95\%$.

5 Features

We will use the following real-valued features. All of these are conjoined with the current tag. (In our feature names `:` represents conjunction.) An example feature name and value is given for the word `France` in the above sentence. Remember that although we show features for the tag sequence in the example, your Viterbi decoder will need to consider all tags for each word.

1. Current word w_i. Example:
   `Wi=France:Ti=I-LOC 1.0`
   We always put the ouput label (`Ti=LABEL`) at the end of feature names.

2. Lowercased word o_i. Example:
   `Oi=france:Ti=I-LOC 1.0`

3. Current POS tag p_i. Example:
   `Pi=NNP:Ti=I-LOC 1.0`

4. Shape of current word s_i. Just replace all letters with `a` or `A` depending on capitalization, and replace digits with `d`. Example:
   `Si=Aaaaaa:Ti=I-LOC 1.0`

5. The features 1-4 for the previous word, and the next word. Examples:
   `Wi-1=<START>:Ti=I-LOC 1.0`
   `Pi-1=<START>:Ti=I-LOC 1.0`
   `Wi+1=and:Ti=I-LOC 1.0`

   Note: `<START>` and `<STOP>` symbols are added to the beginning and end of the sentence, and features in 5 referencing them have values `<START>` and `<STOP>`. When we are at the `<STOP>` symbol, we do not include the features for $i + 1$.

6. The features 1-4 conjoined with the features in 5. Examples:
   `Wi=France:Wi+1=and:Ti=I-LOC 1.0`
   `Oi=france:Si-1=<START>:Ti=I-LOC 1.0`
   In the feature name, we put the features 1-4 before the features in 5.

7. Previous tag (t_{i-1}), as well as features 1-5 conjoined with the previous tag. Examples:
   `Ti-1=<START>:Ti=I-LOC 1.0`
   `Wi+1=and:Ti-1=<START>:Ti=I-LOC 1.0`
   The previous tag comes just before the current tag in the feature name.

8. Length k prefix for the current word, for $k = 1, 2, 3, 4$. Examples:
   `PREi=Fr:Ti=I-LOC 1.0`
   `PREi=Fra:Ti=I-LOC 1.0`
   Our convention is that if the current word is shorter than k for a prefix of length k, it is not duplicated for that prefix. So the word `to` fires the features `PREi=t:Ti=LABEL` and `PREi=to:Ti=LABEL` just once.

9. Is the current word in the gazetteer for the current tag? Example:
   `GAZi=True:Ti=I-LOC 1.0`
   Our convention is that if `Galerie Intermezzo` is in the gazetteer for the tag `LOC`, then this feature fires for both the unigrams `Galerie` and `Intermezzo`.

10. Does the current word start with a capital letter? Example:
    `CAPi=True:Ti=I-LOC 1.0`

11. Position of the current word (indexed starting from 1). Example: `POSi=1:Ti=I-LOC 1.0`

6 Weights file

Each line in the weights file lists the feature name, a space, and the weight for that feature. Features that do not appear in the weights file have a weight of zero.

7 Codebase

We suggest you make your codebase fairly modular. Although you do not need to do it this way, our tagger is broken up into the following parts:

- Decoder (implements Viterbi algorithm)
- Feature Functions (implements the features listed above)
- Load Data (for reading and writing the CoNLL data format)
- Feature Vectors (for loading, saving, and math operations on feature and weight vectors)
- Main program (parses command line arguments)

You may use any programming language for this assignment.

8 Turn-in

Run your tagger on 'test', and email the output and a .zip or .tgz file of your code with the subject 'assignment_one - andrew_id' to cmu-instructors-11763-fa2015@googlegroups.com by 11:59pm on the due date. For questions, please email Kartik at kartikgo@cs.cmu.edu .