CSCI 544 Applied NLP: Homework 3

Due Date: October 28, 2015 (10:59 AM PST)

Bitbucket project name = csci544-hw3

Overview

The goal of this assignment is to get some experience with sequence labeling. Specifically, you will be assigning dialogue acts to sequences of utterances in conversations from a corpus. In sequence labeling it is often beneficial to optimize the tags assigned to the sequence as a whole rather than treating each tag decision separately. With this in mind, you will be using a machine learning technique, conditional random fields, which is designed for sequence labeling. You will be using the toolkit, CRFsuite.

The raw data for each utterance in the conversation consists of the speaker name, the tokens and their part of speech tags. Given a labeled set of data, you will first create a baseline set of features as specified below, and measure the accuracy of the CRFsuite model created using those features. You will then experiment with additional features in an attempt to improve performance. The best set of features you develop will be called the advanced feature set. You will measure the accuracy of the CRFsuite model created using those features. The last task is assigning dialogue act tags to a set of unlabeled data. You will do this with two models. The first model uses the baseline feature set and is trained on all the labeled data. The second model uses the advanced feature set and is trained on all the labeled data.

Data

The Switchboard (SWBD) corpus was collected from volunteers and consists of two person telephone conversations about predetermined topics such as child care. SWBD DAMSL refers to a set of dialogue act annotations made to this data. This (lengthy) annotation manual defines what these dialogue acts mean. In particular, see section 1c (The 42 Clustered SWBD-DAMSL Labels). Note, the statistics in this manual use a different training set than our experiments here but give you a rough idea of how frequently each dialogue act appears. We recommend that you skim the annotation manual to get an understanding of what the tags mean and help you think of good features.

If you have a question about the corpus and can't find an answer in the annotation manual, please post on Piazza or ask a TA or instructor. **DO NOT use resources from the web other than the annotation manual.**

We have divided the corpus into labeled (train) and unlabeled (test) data sets. Download csci544_hw3_data.tgz from Blackboard. Like assignment 2, you will use gunzip and tar to extract data from the file.

In both the labeled and unlabeled sets, individual conversations are stored as individual CSV files. These CSV files have four columns and each row represents a single utterance in the conversation. The order of the utterances is the same order in which they were spoken. The columns are:

- act_tag - the dialogue act associated with this utterance. This is blank for the unlabeled data.
- speaker - the speaker of the utterance (A or B).
- pos - a whitespace-separated list where each item is a token, "/", and a part of speech tag (e.g., "What/WP are/VBP your/PRP$ favorite/JJ programs/NNS ?/."). When the utterance has no words (e.g., the transcriber is describing some kind of noise), the pos column may be blank, consist solely of "./.", have a pos but no token, or have an invented token such as MUMBLEx. You can view the text column to see the original transcription.
- text - The transcript of the utterance with some cleanup but mostly unprocessed and untokenized. This column may or may not be a useful source of features when the utterance solely consists of some kind of noise.

You will also notice a Python file **hw3_corpus_tool.py** included with the data. The TAs have written this code to help you read the data. You are free to use this code or not. You are free to use this code in whatever way you want but if you use it then you must include it with your submission to create a standalone application. The file includes functions to read in these CSV files individually. Read the documentation in the file for more information or use Python's help() function.

CRFsuite

You can find binaries for Linux 64bit on the CRFsuite page. As discussed in the tutorial, each item in the sequence is represented by one line in the data. Each item begins with a label followed by features separated by tab characters. An empty line indicates the end of a sequence (e.g., conversation). The features are binary. The presence of a feature on a line indicates that it is true for this item. Absence indicates that the feature would be false. Here is a training example using features for whether a particular token is present or not in an utterance with the dialogue act "sv".

```
sv TOKEN_i TOKEN_certainly TOKEN_do TOKEN_.
```

For testing, you don't have the labels so you should make up labels (e.g., UNK) so that the format will be the same. Do not use the characters ":" or "\" in your feature names.

What to do

Your main goal is produce a set of dialogue act tags for the unlabeled data. You will use your labeled data ("training data set") to pick the best features for this task. In assignment 2, you simply split the labeled data by randomly putting 25% of the examples in the development set and using the rest to train your classifier. In this case, you would include entire conversations in either the training or development sets. In this assignment, it is up to you how you use your labeled data to evaluate different features. You could split the conversations and use a

certain percentage for development, or you could use k-fold cross-validation.

You should try a set of features that we'll call baseline. In the baseline feature set, for each utterance you include:

- a feature for whether or not the speaker has changed in comparision with the previous utterance.
- a feature marking the first utterance of the dialogue.
- a feature for every token in the utterance (see the description of CRFsuite for an example).
- a feature for every part of speech tag in the utterance (e.g., POS_PRP POS_RB POS_VBP POS_.).

You'll need to create a Python program that converts a single CSV file (either labeled or not) into the baseline features in the data format for CRFsuite. To create a data file for training, you'll need to run this code on every file in the training set and combine the results. REMEMBER that each sequence must be separated by an empty line. Thus, there should be a blank line between the data from one file and the data from another file. To create a data file for development and testing, you'll need to follow the same procedure.

```
create_baseline_features.py CSV_file_name

the features in CRFsuite data format should be sent to STDOUT
```

You should try at least one other set of features that we'll call advanced. The advanced feature set should include more information than the baseline feature set. The idea is that you want to improve performance. As discussed in the grading section, part of your grade depends on developing a feature set better than the baseline. You'll need to create a Python program that converts a single CSV file (either labeled or not) into the advanced features in the data format for CRFsuite.

```
create_advanced_features.py CSV_file_name

the features in CRFsuite data format should be send to STDOUT
```

What to turn in

Remember, we are downloading your entire Bitbucket repository. DO NOT put any of the input data in this repository. Instead, this is what should be in your repository. You need to use exactly the same names as found in the list below. Also, put everything in the top level directory. Do not create subdirectories.

- `report.txt` (use the assignment 3 report template)
- Any necessary support code including hw3_corpus_tool.py (if needed to run your code). Also include any code you used to run your experiments (e.g., to split the labeled data into training and development).
- For the baseline features
  - `create_baseline_features.py`
  - `swbdDAMSL.crfsuite.baseline.model`: the model file from CRFsuite generated from ALL the labeled data

    - `swbdDAMSL.crfsuite.baseline.out`: the labels generated using this model on the unlabeled data
- For the advanced features
    - `create_advanced_features.py`
    - `swbdDAMSL.crfsuite.advanced.model`: the model file from CRFsuite generated from ALL the labeled data
    - `swbdDAMSL.crfsuite.advanced.out`: the labels generated using this model on the unlabeled data

Grading

80% of the grade is based on your performance on tagging the unlabeled data using the baseline features. We will develop a specific grading rubric for assigning partial credit to cases where performance does not match the TAs' baseline model.

10% of the grade is based on your performance on tagging the unlabeled data using the advanced features. This 10% will be assigned by comparing your performance to that of your classmates and the model developed by the TAs.

10% of the grade is based on report.txt. In particular, we need to understand: how you calculated the accuracy of your baseline and advanced feature sets, what features are in your advanced feature set, and how you developed the advanced feature set.

Late Policy

We will immediately attempt to download assignments after the deadline. Any submissions after this deadline and before November 2, 2015 (10:59 AM PST) will be graded with a 30% penalty (i.e., best will be 70/100). After this second deadline, you will receive 0 points for this assignment.

If you wish to submit late (i.e., you want the TAs to grade what they downloaded after the SECOND deadline) then you must fill out **this form** before the second deadline.

Other Rules

- DO NOT look for any kind of help on the web outside of the Python documentation, CRFsuite webpages and SWBD DAMSL annotation manual.
- DO discuss issues on Piazza
- This is an individual assignment. DO NOT work in teams or collaborate with others. You must be the sole author of 100% of the code and writing that you turn in except for the code in hw3_corpus_tool.py.
- DO NOT use code you find online or anywhere else (except for hw3_corpus_tool.py).
- DO NOT turn in material you did not create except for the code in hw3_corpus_tool.py.
- Failing to follow these rules will result in a grade of zero. All cases of cheating or academic dishonesty will be dealt with according to

University policy.