CIS 530 Fall 2014 Final Project

Instructor: Ani Nenkova
TA: Kai Hong

Released: November 24, 2014
Due: 11:59PM December 17, 2014.
No late days allowed!

1 Overview

For the final project, you will develop a system for identifying information-dense news texts, which report important factual information in direct, succinct manner. Such classifier can be very helpful in summarization and question answering, where the prediction from the classifier can be used as an additional indicator of importance. Specifically in single document summarization, the beginning of the article is either a great summary that conveys the key facts reported in the article or rather uninformative side-story. For example, an accurate detector of information density would indicate the need for better summary snippets in systems like Google news.

For the project you will develop a supervised classifier for the information-dense/non-informative distinction. The focus is on feature engineering. The challenge is to come up with features that distinguish between the two types of text.

The teams that achieve the highest accuracy of prediction will be awarded extra credit, so unlike in the assignments here the goal is to develop the best possible system for the task. The labels for the test data will not be released but you will be able to submit your predictions and get an overall accuracy. You will be allowed to do four preliminary submissions. The fifth submission will be final one that you will have to describe and evaluate in your final report. You can use the submissions before the final one as baseline/reference systems. We will also have a live leaderboard where you can see how other teams are doing.

The two baselines will be a lexical representation with dictionary defined without analysis of the training data and non-lexicalized syntactic production rules.

2 Data

The labeled data for the project comes from the business section of the New York Times (NYT). The labels for the opening paragraphs (*leads*) are derived semi-automatically. The NYT release for research purposes[1] includes editor written summaries for many of the articles. To obtain labels, we computed the word overlap between the human summary and the lead. Leads with his overlap are considered information-dense and leads with low overlap with the human summary are considered uninformative.

There are $2,282$ labeled leads in the training set. They all come from the business section of the NYT. Information-dense leads are labeled $+1$ and uninformative leads are labeled -1. For each lead we also provide a label between 0 an 1, which is equal to the fraction of lead tokens that are also in the human abstract

[1] `https://catalog.ldc.upenn.edu/LDC2008T19`

in the NYT. If you decide you want to use this information to improve the prediction accuracy, you are welcome to do so. However the final evaluation will be in terms of binary accuracy.

There are 250 leads in the test set. These were manually labeled by multiple annotators. We will not release the actual labels on the test data. You will submit your predictions in a specified format and get an overall accuracy. You will be able to see your aggregate performance on the test data only four times, so you need to carefully think with settings of the systems you would like to compare.

Note that the lead files are plain text files, so you will need to do tokenization, normalization and sentence splitting. As in all other applications the way you choose to preprocess your data is going to affect the results. Make the most informed choices you can and describe how you chose to pre-process the data in your final submission report. For your own system, use the same pre-processing as the one you developed for the baselines.

All project-related data is in `/home1/c/cis530/project`. The subdirectories include:

- train_data: Plain text files with the leads in the training data. The name of the file is used as identifier in the train_labels file.
- train_labels: Each line contains three strings delimited by space: "LeadId Label Overlap", where *LeadId* corresponds to a file name in the train_data, *Label* is the information-dense label for that lead and *Overlap* is the real-value overlap between the lead and the human abstract. Here is an actual line from the file:

  ```
  2005_03_19_1658101.txt 1 0.987830765674
  ```

- test_data: The directory contains 250 text files for which you need to generate predictions using your model.

3 Evaluation

To evaluate your system, you need to submit your predictions for each test lead. The predictions file should contain one line for each test lead, consisting of a lead id and lead predicted label, delimited by space, as shown below:

```
2007_03_27_1836070.txt 1
 ...
2002_01_23_1361630.txt 1
```

The test set was randomly drawn and manually annotated, so the size of the classes there are not equal as in the training set and reflect the natural expected distribution in the newspaper. The majority class (information-dense) accounts for 58% of the test leads.

We will use accuracy—the number of correct predictions divided by the number of samples—to evaluate your prediction. The accuracy of your submission will be automatically updated on the class leaderboard. Updates will happen automatically every 30 minutes.

4 Project guidelines

- Projects can be done individually or in teams of two.
- You cannot manually label the data in the test set in order to get better performance. This will be considered cheating.

- In your project submission, describe all tools and resources, along with any relevant parameters, that you use. Also include a list of variables in your code that point to the resource location and give tool parameters so these can be changed locally to make it possible to run your code.
- In your project report you should report the accuracy of the system submitted last to the leaderboard. You should submit the code that generated the final leaderboard results.
- The best five systems among all groups get extra credit for the final project. First place gets 25%, second gets 20%, third gets 15%, forth gets 10% and fifth gets 5%.

5 Your own system

The expectation is that you will develop a supervised system to distinguish information dense texts. You can reuse code from earlier class assignments as you implement features for your classifier. First think about what features you may want to introduce and why you may expect them to be useful for this task.

We do not release a dedicated development set and you have a limited number of submissions for your predictions. Should you need to adjust parameters, you can use a portion of the training data as a development set, or use 10 fold cross validation on the training data to decide on any parameters and system settings.

Your final submission should include the code for the final result you submitted and which was recorded at the leaderboard.

Most preprocessing functions can be performed with Stanford coreNLP but you are welcome to use any other tools that you may consider useful. You may find this project resource page helpful:
`http://www.cis.upenn.edu/~cis530/project.html`

6 Report requirement

For the project report, please use the templates provided at the top of the project resource page
`http://www.cis.upenn.edu/~cis530/project.html`
The project report should be no more than 3 pages long. It should include the following sections:

1. The general idea/method for your system: what features you have implemented and why you expect them to be helpful for the task.
2. What resources or tools you have used and how are they included in your implementations.
3. Performance: the accuracy of your system and its variants on the test set. The accuracy of the final system should be that for the last submission for the leaderboard. You need to record for yourself any preliminary results that you have submitted before the final submission. You can also comment on other teams' performance and where you stand with respect to that. However you will not know details of the approach for these other systems and will not be able to draw any meaningful conclusions. For this reason, keep these comparisons as short as possible. It is fine to omit them altogether.
4. Discussion and analysis. Discuss your results and any interesting comparisons and conclusions that you can draw from the results of the five variants of your system that you have submitted to the leaderboard.

7 Submitting your work

7.1 Register your team

Before getting results on the leaderboard, you need to submit your team name. **Every team member has to submit their individual registration. The deadline for registration will be 11:59PM, Dec. 5**. You simply need to submit a text file with a team name. The team name can be the combined names of the members or a made up name—the choice is yours. What you need to submit is a plain text file called `team.txt` containing a single line which gives the team name.

```
% turnin -c cis530 -p proj_groups team.txt
```

7.2 Submit to the leaderboard

You are allowed to submit four preliminary test predictions to the leaderboard. The fifth submission will be your final one—the one you need to submit code for and describe in your final report.

To get the prediction accuracy, you should submit a single file `test.txt` to the leaderboard.

```
% turnin -c cis530 -p leaderboard test.txt
```

The format of the file `test.txt` is described in Section **Evaluation**. The accuracy according to the manual gold-standard will appear at the leaderboard after the next scheduled upadte. The leaderboard will be automatically updated **every 30 minutes**, so you may not see your score immediately. The leaderboard will be hosted here:
`http://www.cis.upenn.edu/~cis530/leaderboard.htm`.

The final ranking will reflect on your last submission before the deadline.

7.3 Submit your code and write-up

Please submit the code for your system and a short readme describing the submitted files:

```
% turnin -c cis530 -p project project_yourpennkey.zip
```

Only one group member needs to make a submission. Below is the submission check list:

- Code for your system
- Final project report
- A readme file with description of the files if you submit more than one file with code