CS4740/5740 Introduction to NLP
Fall 2017
Question Answering

Part 1: due via CMS by Friday, 10/27 11:59pm
Part 2: due via CMS by Thursday, 11/09 11:59pm

1 Overview

Goal for this project: To gain a bit of experience in the design, implementation, and evaluation of question-answering (QA) systems. To gain experience in working with standard off-the-shelf NLP components.

As in previous assignments, this assignment is fairly open-ended. You are to apply your NLP knowledge to implement a QA system that will work on a real world challenge task: **S**tanford **Qu**estion **A**nswering **D**ataset (SQuAD).The input to the system is a passage with a couple of questions, and for each question, the output is a segment of text from the passage that answers it. No human intervention is allowed in deriving answers. Below is a sample [1]:

Passage: In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity. The main forms of precipitation include drizzle, rain, sleet, snow, graupel and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. Short, intense periods of rain in scattered locations are called showers.
Q1: What causes precipitation to fall?
A1: gravity
Q2: What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?
A2: graupel
Q3: Where do water droplets collide with ice crystals to form precipitation?
A3: within a cloud

2 Grouping

We recommend groups of 2-4 people.

3 Data

We provide you with the following files:

- **training.json** The training set.
- **development.json** The development set.
- **testing.json** The testing set.
- **sample.json** A sample prediction file for the development set.
- **evaluate.py** The evaluation script. To run it, execute

  ```
  python evaluate.py <path_to_data> <path_to_predictions>
  ```

  For example, you can evaluate the sample prediction file with the following command:

  ```
  python evaluate.py development.json sample.json
  ```

Please note that the training, development, and testing sets for the project are **different** from the official ones at Stanford, so you need to download the datasets from CMS.

4 Project Phases

4.1 Part 1

Your goal for Part 1 is to:

- Design and implement a baseline system for SQuAD.
- Run, evaluate and analyze your baseline on the training and development corpus.
- Work out a proposal for your final system.

4.2 Part 2

Your goal for Part 2 is to:

- Implement your final QA system based on your proposal (and our feedback) in Part 1.
- Run, evaluate and analyze your final system on the development and testing corpus.

5 What to Turn in

5.1 Part 1

- **Code of your baseline system**.
- **Answer file** produced by your baseline system for the questions from the development corpus.
- **Report** (maximum 2 pages).

5.2 Part 2

- **Code of your system**.
- **Answer files** produced by your system for the questions from the development corpus and the testing corpus.
- **Report** (maximum 8 pages).

Refer to the rubric section for detailed requirements.

6 Rubric (Tentative)

Guideline: Performance does NOT play an important factor in this assignment. We will mainly take into account your design, implementation and analysis of your system. We expect that **your system and report reflect a serious effort at trying to build a good system for this QA task**. Specifically, we expect your final systems to be **more sophisticated** than simple systems such as basic string matching or random answer selection.

6.1 Part 1 (25 pts)

- (8 pts) **Code and Output**
  - (4 pts) If you can show some progress towards building your baseline system.
  - (4 pts) If your baseline system can also generate the answer file in the correct format.
- (15 pts) **Report**
  - (6 pts) Report of your baseline system
    - (2 pts) A description of your baseline system (or your plan of building your baseline system if it has not been finished).
    - (2 pts) A short justification for the baseline system of your choice.

    - (2 pts) Evaluation results (exact match and F1 score) and analysis of the results, e.g. analysis of your system output on some portion of the development corpus.
  - (6 pts) A proposal for your final system. This could include a series of improvements over your baseline system or could be a completely different approach. This proposal is your opportunity to get feedback on your final system, so please make use of it.
  - (2 pt) Include a section describing how each member contributed to the project.
  - (1 pt) Write names and netIDs of every member in your group on the first page of your report.
- (2 pts) Form the group and make submissions correctly on CMS and Gradescope.

6.2 Part 2 (75 pts)

- (5 pts) **Format correctness**. If your system can generate the answer files in the correct format.
- (10 pts) A description of **your QA system and any baseline approaches** that you compare. Enough detail should be provided so that, in theory at least, we could re-implement it. The description should explain each component in your QA system, the steps that your system takes to answer a question, any additional online sources of information used by the system, etc. Make clear which components of the system you built yourself vs. downloaded from elsewhere vs. got from another student in the course.
- (5 pts) A **justification** of why you think that the particular approach will be effective.
- (5 pts) A detailed walk-through of what your system did to handle one question (any one) in the corpus.
- (10 pts) Evaluation results (exact match and F1 score) and analysis of your system's performance on the questions from the development and testing corpus, including a comparison with the baseline system(s). Your analysis should also answer the following questions: How well did the system work? What worked? What didn't work? Can you say anything about which component is strongest/weakest?
- (35 pts) **Design and Implementation Quality**.
  - **Poor (7 pts)** The design and implementation is fatally flawed. In particular, your final submission will be graded poor if it is as simple as the random guess / sliding window approach in [1].

  - **Fair (14 pts)** The design shows minimal effort beyond the simplest implementations such as the random guess / sliding window approach. The implementation does not show enough work for this project.
  - **Average (21 pts)** The design and implementation is reasonably acceptable, though it has some deficiencies. In particular, your work will be graded average if you can develop a system solely based on the type of question and named-entity recognition.
  - **Good (28 pts)** The design is good and the implementation shows an appropriate amount of work for this project.
  - **Excellent (35 pts)** The submission contains more ideas and/or work than most submissions for this project and goes the extra mile. In particular, your work will be graded excellent if it is as good as the logistic regression model in [1].

- (2 pts) Include a section describing how each member contributed to the project.
- (2 pts) Form the group and make submissions correctly on CMS and Gradescope.
- (1 pt) Write names and netIDs of every member in your group on the first page of your report.

7 Suggestions for How to Proceed

- Start simple! Select some really really dumb strategy to produce answers for each question just to make sure that you will have something to evaluate and to turn in. Only after you can do that should you proceed to something more sophisticated. The simple system can be your baseline against which you can compare and submit for Part 1. For example, the very first approach to try can be random guess / sliding window [1]. Another option to begin with is to focus on the type of question and develop a strategy specifically for that question type.
- Feel free to reuse your code from Project 1/2.
- It is fine to try a strategy very different from anything discussed in class. It is even fine if the system that you produce does terribly in terms of performance, however, you need to be able to argue (in your write-up) why the strategy that you investigated MIGHT have worked. As mentioned in the rubric, for your final system, we do expect **more sophisticated and NLP-ish solutions** than very simple solutions (though those simple solutions serve as good baseline systems).

References

[1] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text.