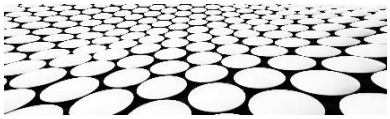
CSI5180 Topics in AI
Virtual Assistants



MODULE 3 – ASSIGNMENT

Building a paraphrase classifier

GOALS

Given the important role that paraphrases play in Virtual Assistant, at the stages of intent detection and fulfillment, especially in QA from unstructured text, the purpose of this assignment is to:

- further explore paraphrases and their variations
- get hands-on experience with a paraphrase dataset
- program a simple paraphrase classifier
- follow a scientific methodology

SUBMISSION DEADLINE

- Wednesday, March 9th, 11:30pm

There is an extra week to this module because of reading week.

SUBMISSION METHOD

- In Brightspace, the Module 3 checklist contains a link for your submission
- Submit a short report (5 pages max excluding your title page) in PDF format, describing your approach, your experiment, and your results

INSTRUCTIONS

EXPLORE THE DATASET

1. Download the dataset SemEval-PIT2015 from this site: https://github.com/cocoxu/SemEval-PIT2015. The dataset was used for an international competition at SemEval 2015, called *Paraphrase and Semantic Similarity in Twitter.*
2. Explore the format of the dataset. From the github site we see (as in figure below) that the data is tab separated and contains not only the raw sentences, but also the POS tags and NE tags. There is also an indicator (label) of whether the two sentences are paraphrases or not, as judged by 5 judges. Look into the file train.data or dev.data (within SemEval-PIT2015-github.zip).

```
Topic_Id | Topic_Name | Sent_1 | Sent_2 | Label | Sent_1_tag | Sent_2_tag |
```

The "Topic_Name" are the names of trends provided by Twitter, which are not hashtags.

The "Sent_1" and "Sent_2" are the two sentences, which are not necessarily full tweets. Tweets were tokenized by Brendan O'Connor et al.'s toolkit (ICWSM 2010) and split into sentences.

The "Sent_1_tag" and "Sent_2_tag" are the two sentences with part-of-speech and named entity tags by Alan Ritter et al.'s toolkit (RANLP 2013, EMNLP 2011).

The "Label" column for *dev/train data * is in a format like "(1, 4)", which means among 5 votes from Amazon Mechanical turkers only 1 is positive and 4 are negative. We would suggest map them to binary labels as follows:

```
paraphrases: (3, 2) (4, 1) (5, 0)
non-paraphrases: (1, 4) (0, 5)
debatable: (2, 3)  which you may discard if training binary classifier
```

3. The number of sentences in the dataset is quite large. If you are not too comfortable with programming with large datasets, you can just take a sample (let say 200 train, 50 dev, 50 test) to work with instead of the full dataset, which has train (13063 sentences), dev (4727 sentences) and test (972 sentences).

FOLLOW A SCIENTIFIC METHODOLOGY

You are free to develop any sentence comparison approaches you want. The purpose is to have as input two sentences and as output a yes/no classification for paraphrase or not. I want you to follow a scientific approach, as given below:

1. Develop a baseline algorithm (Algo A) to paraphrase detection (could be as simple as full exact string match for Yes and No otherwise).
2. Evaluate the results of your baseline on the Dev Set. Calculate recall/precision measures.
3. Develop another approach (Algo B). As one idea, your approach could be based on edit distance including various penalties for different POS. That would be an unsupervised approach. OR, if you are familiar with binary classifiers, such as the ones in sklearn for Python, or other packages for Java, you can use such classifiers.
4. Evaluate the results of Algo B on the Dev Set.
5. Redo steps 3 and 4 to improve on Dev Set. This means to modify your Algo B slightly, or think of another algorithm, and retest.
6. Choose the best method you have developed and evaluate on Test Set.

ATTENTION: You might use the TRAIN data if you build a supervised model. If not (if you code rules), you do not need the TRAIN data. I will not require a supervised model. I want you to follow a scientific method, and program to the level of your actual programming competency. The purpose of this assignment is not complex programming, it is (a) an exploration of paraphrases and (b) an exercise in following a scientific method.

CODING

- You are free to use any programming language you want.
- The programming does not need to be complex. I am interested in the scientific approach more than in the actual paraphrase detection approach. There is no time in such a short assignment to do something complicated. That's not the purpose.

☆☆☆ EVALUATION

- This assignment is worth 10%. Total is on 30 points.
- Your report will be evaluated according to the following content:
  - Intro (2 points)
    - Describe the problem of binary classification of paraphrases
  - Dataset (6 points)
    - Describe what part of the dataset you will use
    - Give your method for changing from graded evaluations to binary (you don't have to follow what is mentioned on the github page, you can decide what you want)
    - Give examples of paraphrases and non-paraphrases (in a table)
  - Methods (12 points – 4 points each)
    - Describe your baseline algorithm, provide an example to illustrate what it does
    - Describe your method 1 algorithm, also provide an example
    - Describe your method 2 algorithm, also provide an example
  - Results (6 points)
    - Show comparative result of 3 methods on Dev
    - Show result of chosen method on Test
    - Discuss the results, including a qualitative analysis (showing a few examples that the best method got right or wrong)
  - Conclusion (2 points)
    - Write a short conclusion about the experiment and future possibilities
  - References (2 points)
    - Provide any reference you use for your programming
    - Provide a reference for the dataset

QUESTIONS

- You can ask your questions in the Module 3 discussion forum on Brightspace.