**CS 6320 – Natural Language Processing**

**Fall 2020**

**Dr. Mithun Balakrishna**

**Course Project**

# Project Steps and Deadlines:

* **Project Group Formation**:
  + Due by **Tuesday, 10/06/2020, 11:59pm**
  + A maximum of two (2) students per project group
  + The group should decide on an appropriate group name
  + One group member should submit a document containing the group name and the group member information i.e. Group name and Group member names, via eLearning
    - Please name the document following the convention “ProjectGroupInfo-GROUPNAME.pdf”, where GROUPNAME is your project group’s name.
    - Submit the document to the “Group Information Submission” assignment inside the “Final Project” folder listed in the course home page on eLearning.
    - Students that want to work on the project individually should also submit this document
  + Students that need help to form a group should meet the Instructor on **Tuesday, 10/06/2020,** at the end of the lecture session on **Blackboard Collaborate.**
    - Students that want to work on the project individually do NOT need to do this
* **Project Demo**:
  + Due date: **TBA**
  + Demo sign-up details: **TBA**
  + Submit your project source code and report via eLearning before your group’s allocated demo session:
    - One group member should submit a single zip file containing the following via eLearning:
      * Project source code/script file(s)
      * A ReadMe file with instructions on how to access the project demo
      * Project report in PDF or MS Word document format.
    - Please name the zip archive document following the convention “ProjectFinalSubmission-GROUPNAME.zip”, where GROUPNAME is your project group’s name.
    - Submit the document to the “Project Final Submission” assignment inside the “Final Project” folder listed in the course home page on eLearning.

# Project Report

Please write a project report (5 to 10 pages) with the following details:

* + - Problem description
    - Proposed solution
    - Full implementation details
      * Programming tools (including third party software tools used)
      * Architectural diagram
      * Results and error analysis (with appropriate examples)
      * A summary of the problems encountered during the project and how these issues were resolved
      * Pending issues
      * Potential improvements

# Project Description:

For this project, you will design and implement models for extracting relations between two named entities in a sentence. You will work with the TAC KBP dataset that contains well over 100,000 examples and covers 41 different relation types.

The contents of this project can be downloaded from the project folder in eLearning. The project contains a data folder containing train and test files. In each example, the following annotations are provided:

1. the spans of the two named entities between which relation holds (as indicated by delimiters e1 and e2)

2. the relation (and its directionality) that holds between the two entities

Some examples from the training set are:

A screenshot of a cell phone

Description automatically generated

The following are the tasks that need to be performed:

1. **Task 1**: Create a class CorpusReader that is able to read the data files and represent the information in a way such that your model can process it.
2. **Task 2**: Implement a deep NLP pipeline to extract the following NLP based features from the natural language statements:
   * Tokenize the sentence into words.
   * Lemmatize the words to extract lemmas as features
   * Part-of-speech (POS) tag the words to extract POS tag features
   * Perform dependency parsing or full-syntactic parsing to get parse-tree based patterns as features
   * Using WordNet, extract hypernymns, hyponyms, meronyms, and holonyms as features
   * Using an API like SpaCy, extract the named entity (NER) tags of the two named entities between which relation holds.
   * Some additional features that you can think of, which may make your representation better. You may want to consult resources like FrameNet, PropBank or NomLex to extract more meaningful features than the ones mentioned above.

Note: you are free to implement or use a third-party tool. Some useful resources are provided at the end of this document.

1. **Task 3**: Implement a machine-learning, statistical, or heuristic (or a combination) based approach to determine the relation and its direction for the given pair of arguments:
   * Run the above described deep NLP on the input corpus (train or dev set).
   * Using the train set, implement/apply a machine-learning, statistical, or heuristic (or a combination) based approach to learn a rules/model that can determine the relation and its direction.
2. **Task 4**: The performance of your system will evaluated on the test set.
   * Run the above described deep NLP on the test set.
   * On the test set, evaluate your system. Specifically, compute the accuracy and macro precision, recall and F-scores for the predictions made by your model under two settings:

1. Assuming that the relation is classified correctly (but not the direction)

2. Assuming both the relation and direction are classified correctly.

* + Report the time taken by your model to make predictions (in seconds).

To understand how this works, consider this toy example. Suppose there are five examples in the test set where our relation set contains only two relations (let’s call them R1 and R2). The true and predicted labels are as given below:

# True relation Predicted relation

1 R1(e1, e2) R1(e1, e2)

2 R1(e1, e2) R1(e2, e1)

3 R2(e1, e2) R2(e2, e1)

4 R1(e1, e2) R2(e1, e2)

5 R2(e2, e1) R1(e1, e2)

Under the first setting, for examples 1, 2 and 3, the relation is classified correctly; but relation for examples 4 and 5 are classified incorrectly. Thus, the accuracy will be 60%.

Under the second setting, only for example 1, the model correctly predicts both relation and its direction. Thus, the accuracy here will be 20%.

# Useful resources

Some resources that you may find useful for this project are listed below:

* [TextBlob](https://textblob.readthedocs.io/en/dev/): Python API for common NLP tasks
* [spaCy](https://github.com/explosion/spaCy): Python API commonly used in the industry
* [scikit-learn](https://scikit-learn.org/stable/): Python API for ML frameworks
* [NLTK](https://www.nltk.org/): Python API for common NLP tasks
* [Stanford NLP](https://nlp.stanford.edu/software/index.shtml): Java tool for common NLP tasks
* [OpenNLP](https://opennlp.apache.org/): Java tool that provides machine learning libraries for NLP tasks
* [MIT-IE toolkit](https://github.com/mit-nlp/MITIE): C, C++ and Python tools for Information Extraction
* [Charniak Parser](https://github.com/BLLIP/bllip-parser): C++ implementation of the Charniak parser

# Links to Related Papers

Some papers that implement similar systems are cited below:

* Rink, Bryan, and Sanda Harabagiu. "Utd: Classifying semantic relations by combining lexical and semantic resources." In Proceedings of the 5th International Workshop on Semantic Evaluation, pp. 256-259. 2010.
* Szarvas, György, and Iryna Gurevych. "TUD: semantic relatedness for relation classification." In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pp. 210-213. 2010.
* Negri, Matteo, and Milen Kouylekov. "Fbk\_nk: A wordnet-based system for multi-way classification of semantic relations." In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pp. 202-205. 2010.
* Esuli, Andrea, Diego Marcheggiani, and Fabrizio Sebastiani. "ISTI@ SemEval-2 Task 8: Boosting-Based Multiway Relation Classification." In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pp. 218-221. 2010.

# Project Point Distribution

1. Max points available: 100 points
2. Division of points:
   1. **Group information: 2 points**
   2. **Project implementation and demo: 90 points**
      1. Task 1: 5 points
      2. Task 2: 30 points
      3. Task 3: 35 points
      4. Task 4: 20 points
   3. **Project Report: 8 points**