Project Plan Lightning Talk

William Sengstock, Kelly Jacobson, Zach Witte, Austin Buller, Sam Moore, Dan Vasudevan, Jacob Kinser

Project Overview and Management Style

- Project Goal is to identify incorrectness that happens in POS tagging software documentation.
- Due to nature of our project, we will use a waterfall management approach
    - Different phases include
        - NLP research
        - Dataset discovery
        - Analysis of NLP models
        - Formulating new ways to train models for higher accuracy.
- Github will be used for version control of code
- Discord is used for communication and google drive stores our group assignments.

Task Decomposition

Research

- Learn basics
- Data pre-processing
- Vectorization
- Unsupervised vs. supervised learning
- Clustering

Model Construction/Development

- Election of libraries
- Tokenization experimentation
- POS tagging
- Vectorization
- Analysis
    - Gather results
    - Studying accuracy
    - Training model

Rough Schedule

- First Client Meeting Thursday, 9/9
- Week 1: 9/9-9/16
  - research on NLP basics
- Week 2-3: 9/16-9/23(skipped meeting)-9/30
  - research on different NLP techniques, vectorization, supervised/unsupervised learning
- Week 4 : 9/30 - 10/7
  - building our first NLP models
- Week 5: 10/7 - 10/14
  - more NLP models using different libraries (Spacy, StanfordNLP, etc)
- Week 6: 10/14 - 10/21
  - Analyze our respective models and the accuracy/training methods
- Week 7 - Final Week
  - Come to a consensus on what model to focus on
  - Study the different advantages and disadvantages of the chosen model
  - Work to better (train) the model to optimize efficiency regarding NLP in software documentation

Risks/Risk Mitigation

Task 1

- For NLP models, build running code in Jupyter Notebook
- 0.1 probability for risk is low, because code needs to be correct to run

Task 2

- Compare different packages for each word embedding technique
- 0.2 probability for risk, possibility of repetition in packages, but still relatively low

Overall

- Low number of risks because the project consists of running and comparing code
- Risks are limited to making sure the code runs properly, and successfully differentiating word embeddings