Mining and Evaluating Verb tags and Other Important POS tags inside Software Documentation

Design Document

Team 17

Client: Hung Phan & Hiep Vo

Advisor: Ali Jannesari

Team Members/Roles:

William Sengstock - Team Leader

Kelly Jacobson - Team Organization

Jacob Kinser - Component Design

Zachary Witte - Component Design

Samuel Moore - Component Design

Dan Vasudevan - Component Design

Austin Buller - Component Design

Team email: sdmay22-17@iastate.edu

Team website: https://sdmay22-17.sd.ece.iastate.edu/

Executive Summary

Development Standards & Practices Used

- Software Practices
    - Continuous Testing
    - Proper documentation
    - Code Simplicity
- Engineering Standards
    - ISO-IEC 9001: Quality Management
    - ISO-IEC 830: Software Requirements Specifications
    - ISO-IEC 12207: Software Life Cycle Processes

Summary of Requirements

- Need to apply knowledge of mathematics, science, and engineering.
- Involves students from several programs, i.e., CprE, EE, and SE.
- Requires students to identify, formulate, and solve problems.
- Allows students to use the necessary techniques, skills, and engineering tools for this practice.
- Collect datasets of software documentation to identify which types of incorrectness usually happen.
- Focus on the study of verbs in natural language processing and other important tags in the corpus.

Applicable Courses from Iowa State University Curriculum

- COM S 227
- COM S 228
- SE 309
- SE 319
- SE 329
- MATH 207
- SP CM 212
- ENGL 314

New Skills/Knowledge acquired that was not taught in courses

- Natural Language Processing Libraries (Spacy, NLTK, StanfordNLP)
- Neural Networks (RNN, LSTM)
- Data preprocessing techniques such as stemming, lemmatization, and tokenization
- Vectorization strategies (CBOW, Skip-Gram) and libraries (word2vec, fasttext, TFIDF)
- Python, Jupyter Notebook
- Data Science Libraries (Numpy, MatPlotLib, SciKitLearn)

Table of Contents

1 Team

1.1 Team Members

- William Sengstock
- Kelly Jacobson
- Jacob Kinser
- Zachary Witte
- Samuel Moore
- Dan Vasudevan
- Austin Buller

1.2 Required Skill Sets for Your Project

- Apply knowledge of mathematics, science, and engineering.
- Should identify, formulate, and solve engineering problems.
- Optimal communication skills.
- Knowledge of coding languages and practices.

1.3 Skill Sets covered by the Team

- William Sengstock - Covers all skills
- Kelly Jacobson - Covers all skills
- Jacob Kinser - Covers all skills
- Zachary Witte - Covers all skills
- Samuel Moore - Covers all skills
- Dan Vasudevan - Covers all skills
- Austin Buller - Covers all skills

1.4 Project Management Style Adopted by the team

For this project, we will be using the waterfall project management style. This style best suits the project because we have been building on our knowledge as the semester progresses.

1.5 Initial Project Management Roles

William Sengstock - Team Leader

Kelly Jacobson - Team Organization

Jacob Kinser - Component Design

Zachary Witte - Component Design

Samuel Moore - Component Design

Dan Vasudevan - Component Design

Austin Buller - Component Design

2 Introduction

2.1 Problem Statement

Our project is focused on researching the application of Natural Language Processing techniques to software documentation. The objective is to improve Part of Speech (POS) tagging to make information mining more effective when applied to software languages. Current NLP models are used for English and other human languages. We want to apply these models to software languages like Java, Python, C, etc.

2.2 Requirements & Constraints

Research Requirements/Constraints

- Variety of topics to research:
    - Data pre-processing techniques
        - Stemming
        - Lemmatization
        - Tokenization

    - Word vectorization
        - Skip-Gram
        - Continuous Bag of Words (CBOW)
    - Word embedding and word clustering
    - Algorithms
        - Gradient Boosting
        - Random Forests
    - POS tagging techniques
    - Supervised /unsupervised Learning
        - Clustering algorithms
- Each member of the project is expected to research on their own and cite credible sources

Programming Requirements/Constraints

- Models will be written in Python using Jupyter Notebook (constraint)
- Some models we will research and experiment with are:
    - Natural Language Toolkit (NLTK)
    - StanfordNLP
    - spaCy
    - Term Frequency - Inverse Document Frequency (TF-IDF)
- Data used for research will come from approved sources, such as www.kaggle.com and those given by the client.
- Evaluate the performance of each model compared to manual review.

2.3 Engineering Standards

Engineering Standards:

- ISO-IEC 9001: Quality Management
    - Helps ensure quality throughout the project, as well as continuous improvement
- ISO-IEC 830: Software Requirements Specifications
    - Influences structure of the project, along with communication between users
- ISO-IEC 12207: Software Life Cycle Processes
    - Influences project design, maintains product throughout stages

2.4 Intended Users and Uses

Our end goal for this project is to improve NLP processes on software documentation. Our project will directly benefit people researching NLP, including our client, and will be most

useful as a reference for other projects. In the long run, our research will be the basis of other programs and projects that use POS tagging for software documentation. For example, a search engine for software documents would utilize POS tagging to identify similar documents. This could help a researcher looking for Python documentation find relevant information. Another example used is searching for similar code blocks across various sources. It would also be beneficial to someone inheriting a project who could use the same NLP processes we research to get a better understanding of their project. There are many possible uses of NLP and POS as applied to software documentation. Our results will become a basis for future progress by experimenting with different word embeddings, algorithms, and more.

3 Project Plan

3.1 Project Management/Tracking Procedures

Our team will stick to a waterfall project management style. This style best suits the nature of research as we continuously build upon what we have learned before. It is also beneficial because it allows for changes to be made to the project plan. The goals for this project require extensive planning so we know what data we are going to use and why we will use it, the method in which we will analyze the accuracy of numerous NLP models, and how we will train the models to increase efficiency. The waterfall model allows us to keep our plan up to date with the project's needs.

Our team has been using Google Drive to keep track of documentation, including weekly meeting notes and assignments. We are using Github to keep track of research documents shared by our client and using individual repositories to keep track of code development. We are using Discord for conversation regarding meeting times, deadlines, and group work. We have found Discord to be very advantageous as it allows our team to communicate with each other very easily; it also allows us to meet via voice chat instantly and easily communicate with our TA and client. Throughout the course of this semester and next, we will be utilizing these platforms to ensure that we develop our project in an organized and efficient manner.

3.2 Task Decomposition

Part 1: Research

Our project is based on Natural Language Processing techniques, and before we can begin working with these techniques, we need to understand them. The first part of our project then is to research NLP and try to understand the processes involved in creating an NLP model.

1. Basics of NLP (week 1)

2. Data Pre-Processing (weeks 2-3)
    a. tokenization
    b. data cleaning techniques
3. Vectorization Strategies (weeks 2-3)
    a. Skip-Gram
    b. Continuous Bag of Words (CBOW)
    c. Term Frequency - Inverse Document Frequency (TF-IDF)
    d. N-Gram
    e. Count Vectoring
    f. word2vec
    g. Fasttext
4. Unsupervised Learning vs. Supervised Learning (weeks 2-3)
5. Word Clustering (weeks 2-3)
    a. K-means
    b. Density-Based
    c. Hierarchical

Part 2: Model Building

The second part of this project involves implementing what we learned while doing research. We will build multiple models to experiment with different techniques and libraries.

Each model will include some techniques from the following steps:

- Data Pre-processing
    - Clean formatting, remove stop words
    - Stemming
    - Lemming
    - Tokenization
- Vectorization
    - word2vec
    - fasttext
    - TFIDF
- POS Tagging and Word Embedding with different Libraries
    - NLTK
    - StanfordNLP
    - Spacy
    - Glove
- Analysis
    - Word Similarity
        - Cosine Similarity

- Sentiment Analysis

2.1 NLP for text

First, we want to implement various NLP models so we can understand how the different python libraries work. These models are just for understanding how NLP works and use plaintext as data.

2.2 NLP for software documentation

Second, we want to create NLP models specifically for software documentation data. This begins the process of experimenting with different NLP libraries to see what works best with software documentation.

Part 3: Model Testing & Evaluation

For this part of the project, we will be evaluating the models we have built. We have to decide on evaluation metrics, the processes that are most important to us, and how to define a good model.

- Compare each NLP model to manual review of data
- Compare each model's performance against each other
- Build a python tool for result analysis
- Choose the libraries and techniques best suited for this project

Part 4: Second Semester Improvements

After building and comparing various NLP models, we want to improve them. The second semester will be devoted to changing current models to better-fit software documentation data. This will involve a cycle of making modifications, testing and evaluating them to see what works best, deciding on the changes to keep, and beginning again making more modifications.

The general process will go something like this:

- Identify areas that can be improved
- Experiment with possible modifications
- Test and Evaluate those changes
- Keep the ones that improve the model
- Document the modifications

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Part 1: Research

A key milestone for this task was to develop a better understanding of data pre-processing, vectorization strategies, unsupervised learning vs. supervised learning, and word clustering. This knowledge would give us a better understanding of what we would be implementing in later steps.

Part 2: Model Building

A key milestone for this task is to start implementing what we researched in part 1. We will begin testing what we learned on different sets of data. We will experiment with different model builds and analyze the results in part 3. There are various word embeddings and algorithms that can be used for natural language processing, and we have to experiment with them to understand how they all work.

Part 3: Model Testing & Evaluation

This milestone for our project involves model testing and evaluation. That is because there are various word embeddings and algorithms that can be used for natural language processing. By testing and evaluating the models made in part 2, we can find learn how each of them operates and how they differ from one another. Using that information, we can identify and choose the ones that best suit our end project goals.

Part 4: Second Semester Preparation

In this milestone, we will evaluate all of the work we have done throughout the semester and determine what we need to do next. For the second semester, we will be familiar with natural language processing, and from there we can dive deeper into the project.

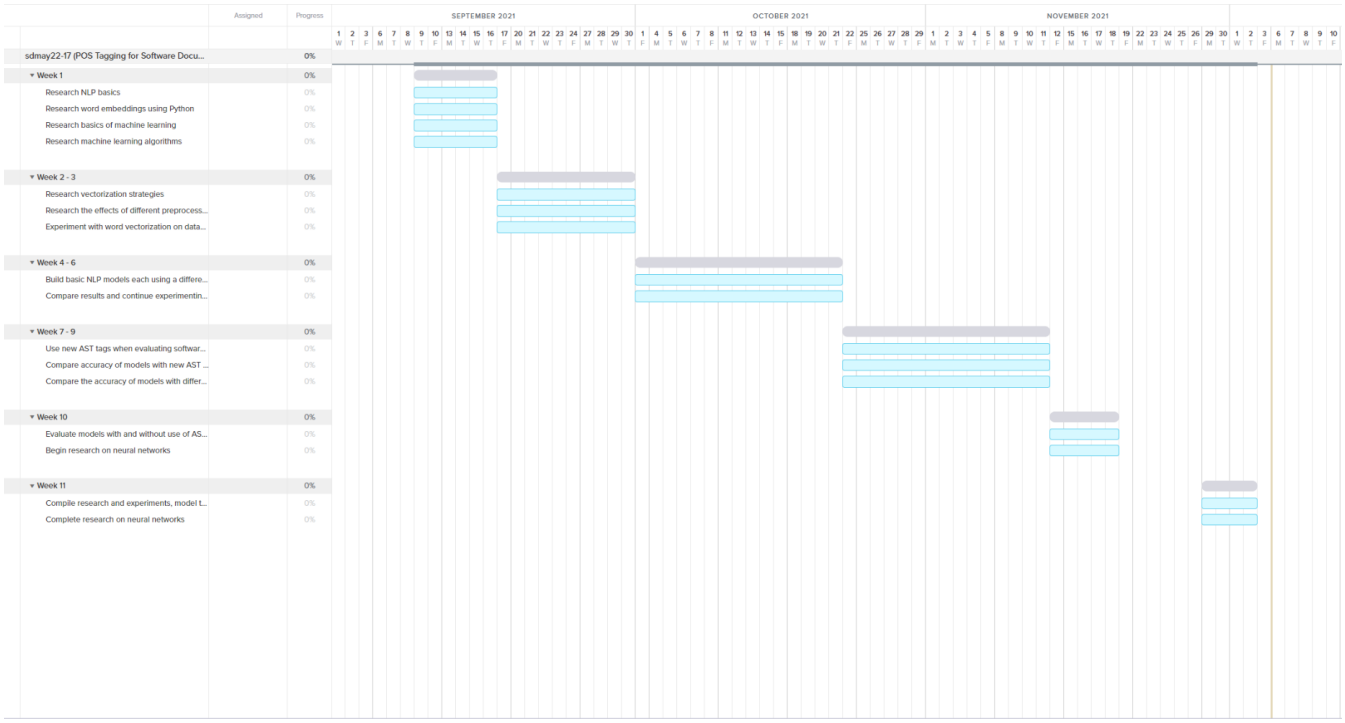
3.4 Project Timeline/Schedule

- First Client Meeting Thursday, 9/9
- Week 1: 9/9-9/16
    - Research on NLP basics and word embeddings using Python
    - Research basics of machine learning and machine learning algorithms
- Week 2-3: 9/16-9/30
    - Research strengths and weaknesses of different vectorization strategies
    - Research the effects of different preprocessing methods

    - Use word vectorization on different data sets
- Week 4: 9/30 - 10/7
    - Team members will start to build individual NLP models using different libraries
- Week 5: 10/7 - 10/14
    - Team members will continue to get more practice using different NLP libraries (Spacy, StanfordNLP, etc), using datasets of their choice
- Week 6: 10/14 - 10/21
    - Comparison of different NLP models
- Week 7: 10/21 - 10/28
    - Use new AST in combination with POS tags on software documentation data.
    - Compare the accuracy of models with the new AST tags
    - Compare the accuracy of models with different preprocessing steps
- Week 8: 10/28 - 11/4
    - Continued work with NLP models, comparing algorithms
- Week 9: 11/4 - 11/11
    - Continued work with NLP models, comparing algorithms, model testing
- Week 10: 11/11 - 11/18
    - Evaluate models with and without the use of AST tags
    - Begin to research neural networks
- Thanksgiving Break
- Week 11: 11/18 - 12/2
    - Compile research and experiments, model testing
    - Neural network research
- Week 12: 12/2 - 12/9 (Dead Week)
    - Second-semester preparation
    - Faculty presentation
- Finals Week

Second Semester = 15 Weeks (not including Finals week or spring break)

- Week 1:
    - Initial meeting with team and client to set weekly meeting times and goals, and set up an initial task schedule for the semester
- Week 2 - 14:
    - Working on the specified goal, continuing research, updating project plan
- Week 15:
    - Finalize research and give a final presentation



3.5 Risks And Risk Management/Mitigation

Task 1: for all NLP models, ensure that code runs correctly in Jupyter Notebook. 0.1 probability for risk. The risk is relatively low because the code needs to be correct in order to run.

Task 2: for each word embedding that is tested, properly utilize the different packages in each to compare and contrast. 0.2 probability for risk, could see a repetition of certain packages, but still relatively low.

- There are a low number of risks because the project consists of running and comparing code. Therefore, the only risks that will need to be assessed are making sure the code runs correctly, and properly differentiating word embeddings.

3.6 Personnel Effort Requirements

Task	Hours Needed (per person)

1) Research	5
2) Model Building	15
3) Model Testing and Evaluation	10
4) Second Semester Preparation	3

The hours calculated are a rough estimate of what we believe each person should put in for this project. Tasks 2 and 3 require more hours than the first because they will span for several weeks. During these tasks, more time will be needed to experiment and test different models, compared to the initial research phase. For the fourth task, there are fewer hours due to the fact that we are essentially compiling our research throughout the semester, and setting ourselves up for the next semester.

3.7 Other Resource Requirements

- Personal computer in order to test and run code
- Jupyter Notebook with Python
- Discord, Google Drive, Zoom, Webex, and Github accounts

There are only a few materials that are needed for this project due to the fact that it will be done entirely online.

4 Design

4.1 Design Context

4.1.1 Broader Context

Our project involves research into natural language processing as it applies to software documentation. The ultimate goal is to improve NLP models to process software documentation data more accurately. Potentially, anything that uses NLP to process human language could

process code in the same way. This would lead to search engine optimization specifically for code, better code auto-complete functions in IDEs, better filtering of software documents, and give access to analysis of software documentation. Our work could expand the NLP field and improve the understanding of how code is analyzed with NLP techniques.

Area	**Concerns**
Public health, safety, and welfare	Our project should have almost no effect on public wellbeing. Our primary stakeholder is our client. They hope to use our project as part of their research into the broader field of NLP. If we do a good job, it makes their life easier and allows them to do more with respect to their own NLP research. We only want to improve a knowledge base that already exists. In the future, if someone can build on what we find, it may improve the quality of life of anyone who needs to find and read software documentation data.
Global, cultural, and social	Because our project is not public, it should not have any social impact. Ethically, our biggest challenge is being aware of our own biases in software development.
Environmental	We have no physical product that could take up resources, we only have software. Our biggest environmental impact is the environmental cost of the electricity used to power our own devices.

Economic	Our research product has no direct economic impact. We are research-focused for the sake of expanding the field of NLP, not for economic gain.

4.1.2 User Needs

A student working on a programming assignment needs to find the right Java documentation related to their project because they want to understand how a piece of code works.

The lead developer on a team of programmers needs to analyze the code written by their team because they want insight into how their team approaches coding.

A new hire needs to understand how an outdated program works because they are tasked with updating and debugging it.

These user groups all need some functionality that lets them more easily look at code and the documentation of code in order to understand something about that code.

4.1.3 Prior Work/Solutions

Similar research into NLP for software documentation has been done by IEEE in a published research paper "Choosing an NLP Library for Analyzing Software Documentation: A Systematic Literature Review and a Series of Experiments." [1] Our project will be following similar steps.

First, a variety of NLP models were chosen, including Google's Syntaxnet, Stanford Corenlp Suite, Nltk Python Library, and Spacy. The researchers then experimented on software documentation from Stack Overflow, GitHub ReadMe files, and Java API documentation. Using their own comparison tool created specifically for this project, they compared how each model accomplished tokenization, and Part of Speech (POS) tagging both with general and specific POS tags. They compared the models with each other as well as a manual review of the data. Notably, each model was used with its default settings and without any modifications.

Our project has the same goal and therefore a similar approach. We will also be experimenting with different NLP models including NLTK, StanfordNLP, Spacy, and Glove. These were suggested by the client. We will use more models if we have time. The client will be giving us

some software documentation data to run these models on. We will compare these models with each other to see which one performs best depending on the task, such as tokenization, POS tagging, word clusterization, etc.

An advantage of referencing the IEEE paper is that while we are not conducting the exact same experiment as the IEEE model, it can give us some hints as to how to perform our work. For instance, the researchers created a tool for comparison. We do not have a comparison tool besides manually checking each model to see if they did what we expected. In the future, we will most likely be building a comparison tool, but it will be specific to the needs of the client. The paper also gave us a preview of how well each model performed. For example, when we are doing our own experiments, we can expect that Spacy will perform best in POS tagging.

One disadvantage of the IEEE paper is that it did not account for modifying the NLP models. While we will begin our research only using the default models, but in the future, we will be modifying the NLP models to see if we can improve their performance. This will be a late-stage task. The results of modification may completely change the results of the initial experiments and may give rise to unforeseen circumstances.

The biggest difference between the IEEE research paper and our own work is that the IEEE experiments were done by professionals in the field of NLP. Our team started with no experience in NLP. We have been learning as we go. This means we have to be extra careful that we are using each model correctly, and it may mean we cannot find problems as easily as an expert might. Still, the IEEE paper can give us hints to work in the right direction.

4.1.4 Technical Complexity

Our project's technical complexity comes from the fact that we are all working with things we have no experience with. Coming into this project, none of us have ever worked with Natural Language Processing before. We started this project by just performing research to improve our individual understanding of NLP as a field. Secondly, we will not be working with only one NLP model, but several. As much as we have time to research and test. Each model is used differently and you have to learn the syntax and functionality for each individual model.

Not only that but NLP as a field is very broad and complex. There are a lot of different NLP techniques out there, as well as different things you can do with NLP models. We have to understand these different techniques and functions to decide what we want to use in our project.

Our project is not only based on current industry standards of NLP but also has the goal of improving them. Our goal is to improve how a variety of NLP models handle software documentation data. The process to reach this goal includes research on each model, implementation of each model, and modifications to improve the model.

4.2 Design Exploration

4.2.1 Design Decisions

Key Design decision 1: NLP Toolkit

NLP Toolkits are useful for cleaning text, tokenizing text, and Parts of Speech tagging. Different NLP Toolkits offer different advantages and our team must decide which one best fits our needs.

Key Design decision 2: Method of training for NLP model

We need to train our model, so it is able to make accurate predictions given different data inputs. We need to find the best way to teach our model, whether it is supervised or unsupervised learning, structured or unstructured data, and what techniques will be the most helpful when teaching a model to look at software documentation.

Key Design decision 3: Word embedding technique

Word embedding will be key in our design for recommending different software documents based on the one the user is currently reading. Different word embedding techniques allow us to understand text in different ways to determine its meaning.

4.2.2 Ideation

When trying to find an NLP Toolkit, we experimented with different open source options to determine which would work best for our project. We identified the following as potential options:

- NLTK
- SpaCy
- Stanford NLP
- Glove
- Apache

In our first client meeting, we were tasked with researching the difference between the open-source NLP tools above. Since the researching phase, we have been testing these tools on different sets of data to see which one will best suit our needs.

4.2.3 Decision-Making and Trade-Off

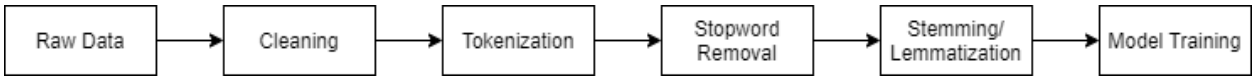
To identify the pros and cons of each option, we have been experimenting with the different toolkit options on our own datasets that we found online. We haven’t fully decided on which

toolkit we would like to use, but we have a few that we believe will fit our needs the best. As of now, our group thinks that spaCy and NLTK provide the best tools for us to use when building NLP models.

As we create our own models on different datasets, we are looking at which toolkits are the easiest to understand and use, while still giving us quality data to analyze. Each toolkit provides its own methods for cleaning text, tokenizing text, stemming, lemming, tagging, etc. We are applying these toolkits and their functions to datasets and then manually analyzing them to see if the output being provided is accurate and what we expect. For example: Are all the proper stop words removed? Are words with similar meanings being represented as such in vector format? Are common phrases being properly grouped together? Are the parts of speech being tagged correctly? We also have to consider how these models would work when analyzing software documentation in particular, and if they provide enough for us to improve upon current NLP models for software documentation. We also want to see if using a more complicated toolkit and spending the extra time to learn how to use it will be better in the long run when trying to analyze software documentation. Some toolkits might be easier to use, but they might also not provide as many features that could be useful in the future. These are all things we are considering as we continue to experiment and create our own models.

4.3 Proposed Design

4.3.1 Design Visual and Description



The image above shows a typical NLP pipeline for processing data. Just like other NLP pipelines, our design will follow the same steps but the way the steps are implemented will be different.

Raw Data

To start, raw data/text is given to the software to process. For this design, the incoming data will be software documentation. The data can be raw, straight from a website with no formatting on the text, or in a file where the data has been cleaned. Data is typically in a .csv file, separated by category into columns.

Cleaning

This step involves preparing the incoming data to be tokenized, and this is also where the design will vary from typical NLP pipelines. Usually, in the cleaning step, all non-letters are removed

from the data, but in software documentation, missing punctuation can change the meaning of code and that can affect the accuracy of the final model.

Tokenization

Tokenization is the process of splitting up the cleaned data into individual units called tokens, these tokens can be either words, subwords, or characters. This allows a program to evaluate every piece of the data individually.

Stop Word Removal

Stop word removal involves removing words from the data that add no context or value to what the text is about. For example, “a”, “the”, “is”, “are”, are all stop words that would normally be removed. However, in the context of software documentation, stop words like “if”, “or”, “for”, “in”, are common in code and could change the meaning if removed. This means that certain stop words should stay while others can still be removed.

Stemming and Lemmatization

Stemming and lemmatization are both common methods of shortening words down to a common base. Stemming simply cuts off the end of a word i.e., “runners” is shortened to “runner”, and is faster than lemmatization. Lemmatization shortens a word down to its base i.e., “runners” is shortened to “run”, and while it is more computationally heavy it creates a smaller corpus of words than stemming.

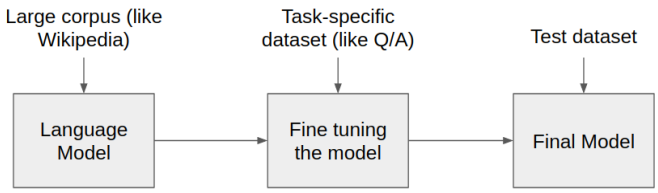
Model Training

Model training is the process of creating word associations. This is what allows the software to make predictions based on the given data. The results of these models will vary based on how the data is processed in the previous steps.

One important step of the model training process is vectorization. Vectorization is the process of converting tokens into vectors. These vectors are used in model-specific calculations that can identify similarity between words, can predict the next word given a group of words, and can identify new words the model has not yet processed. The two common types of vectorization we have researched are the continuous bag of words (CBOW) and skip-gram models. CBOW predicts a missing center word from the surrounding words that give context. Skip-gram is used to find the words most closely related to a single given word.

4.3.2 Functionality

Our design is specifically intended to operate in a software environment. This means the models we are using and the manner in which we are training and feeding them data has software documentation data at the forefront of its creation process. Natural Language Processing was not necessarily created with software documentation in mind, and our attempt is to train a custom model to bridge the gap between regular text and software-related text.



Currently, our model satisfies a number of functional and non-functional requirements. It satisfies functional requirements such as taking into account that it must output quantitative data in order for our team to understand how well it is working and where there is room for improvement. Our model must also be able to take in a large corpus of data to allow the best chance of success, which our design takes into consideration. In regard to non-functional requirements, our design encourages continuous improvement in efficiency and choice of models/tokenization strategies. This is very important because a higher rate of efficiency allows our model to be trained on larger datasets, ultimately resulting in an increased inaccuracy.

4.3.3 Areas of Concern and Development

The main concern with our current design is that we may not be able to create a model that completely understands Software Documentation. As mentioned in section 3.3.2, NLP models were initially designed to comprehend natural human language. However, software documentation isn't written in traditional English. There are a variety of abbreviations and word structures used in software documentation that wouldn't be used outside of it. Therefore, making our model understand what the text is saying in a software documentation context will be very difficult.

Our immediate solution to this is to look at other NLP models that have already been created for software documentation. Even though there isn't a perfect model out there, we can use previous designs to improve our current design. The client will be helpful for this because they have already analyzed previous NLP models that have been made for software documentation purposes.

4.4 Technology Considerations

The technology we have been using for this project has provided everything that we have needed throughout the semester and will continue to do so throughout next semester. Our tasks require reading different documents of text, using several different libraries, and manipulating data in several different ways; all of this can be achieved using Jupyter Notebook. Python itself is a language that functions at a very high speed when dealing with large amounts of data, which is a huge advantage in this case. Python is not the fastest language we could use. For example, C and C++ might have the upper hand when it comes to speed in many cases, but Python provides an array of NLP libraries that are rather simple to use and have many tutorials online; this is a tradeoff our team has decided to make.

There are no pressing problems with the technology that we have chosen to use. Our mentor has experience in the area of NLP and has experience working with Python; he has ensured us that this is a very efficient way to go about our project and the best approach given our team's experience.

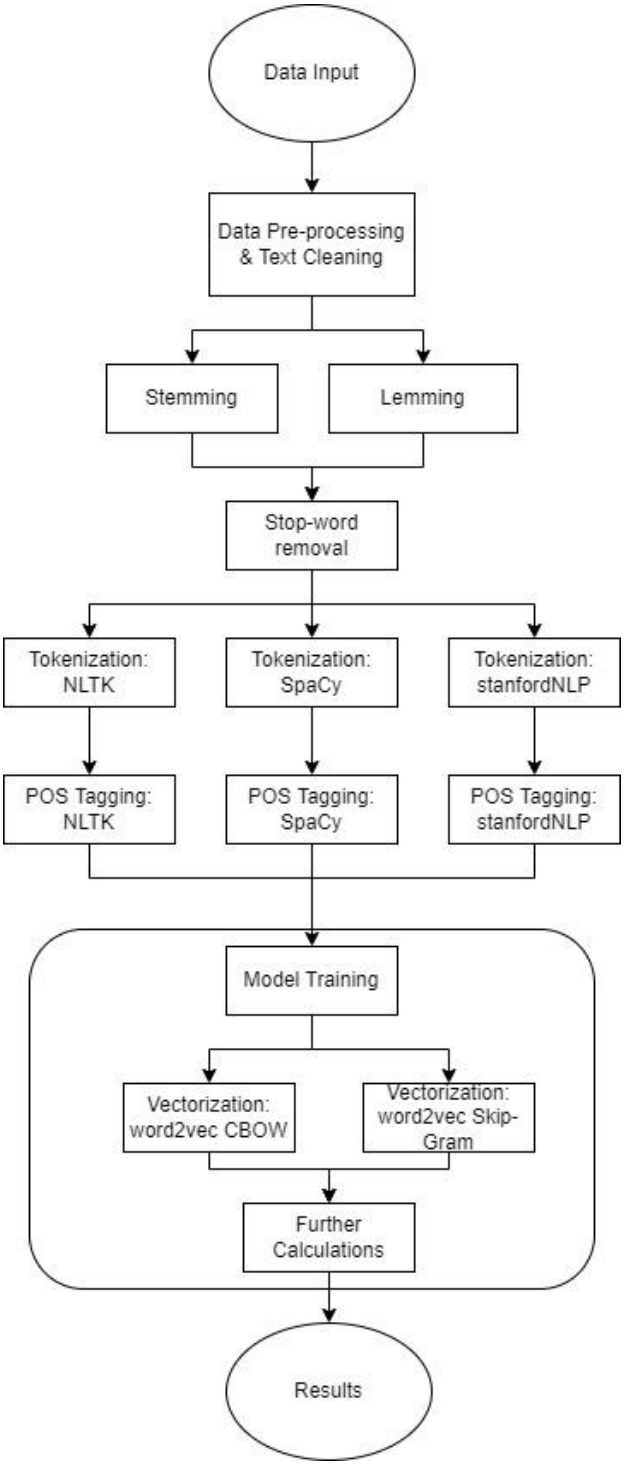
4.5 Design Analysis

Using the steps from 4.3 to create a working model has worked for us over this semester. Following these steps, we have been able to create functional NLP models that have provided us with data we can analyze to determine what factors have the greatest impact when trying to apply POS tagging to software documentation. While we have not been able to create an ideal model for processing software documentation, we have a solid foundation that we can build upon using the data we have gathered over the course of the semester to aid us in creating our final NLP model next semester.

Throughout the last month, we have been researching which preprocessing techniques would be most beneficial when analyzing software documentation. These preprocessing techniques include tokenization, stop word removal, and stemming/lemmatization. We are considering how these different techniques affect software documentation and potentially modify our initial design based on our findings.

4.6 Design Plan

Our program design has several steps that are the same for every model, just executed differently depending on the libraries used and expectations of the model. Our general design is still similar for each model we have been experimenting with, and will be in future models.



- Data pre-processing involves preparing the text to be processed and is the same for every model.
- Stemming, Lemmatization, and stop-word removal are steps you can skip, but it will affect the final results of the model
- Tokenization is usually handled by the NLP software package, such as SpaCy or NLTK.

- Basic POS tagging can be done after tokenization with the help of those NLP libraries.
- Model Training is the final step of the process, and is heavily reliant on libraries used previously.
- Vectorization is a step of model training that converts each token to a vector that the model can use for calculation.
- Model Training can involve different calculations depending on the libraries you are using and what kind of results you are trying to get from the model. Our work next semester will involve more research into this part of the process.

5 Testing

5.1 Unit Testing

Units that will be tested throughout the duration of this project will be different types of word embeddings. Our project revolves around analyzing POS tagging, tokenization, and more for different word embeddings for software documentation. As we use these embeddings to train our data, we want to test them to find the one that is best suited for software documentation. We want to test the different types of functions, the libraries, and how the unit works overall. Examples of these units would be NLTK, Word2Vec, Stanford NLP, SpaCy, etc. Python will be our tool to implement these tests because for this project we are writing and comparing code.

5.2 Interface Testing

The interfaces for our design that will ensure multiple components will work as expected would be our models. For this step, we want to be sure that our components will produce the proper result when combined, and we can see those results when we train the model for the data at hand. Once the model has been trained, we can see what word embeddings and their features work well for the data, and improve on the ones that don't. Doing so will allow us to improve our understanding of NLP and help apply it to properly tag software documentation. As stated in the previous section, Python will be our main tool to write the code for the project.

5.3 Integration Testing

Our individual modules would involve pre-processing data, POS tagging, tokenization, vectorization, and clustering. Since different modules depend on others' outputs as their input, these modules would be expected to all run together and produce the desired result. Additionally, we will be testing several models and the best parts of these modules will eventually be spliced together to create a final model that does everything we need it to do. Each piece should be easily modifiable so they can seamlessly integrate with different models. As we are testing different implementations for each module, integration testing is important to ensure our system still produces the correct results. The tool used for integration testing would be Python.

5.4 System Testing

Our system-level testing strategy is going to test the end-to-end code of the model. This means that we are going to check if each step from importing data to outputting accuracy of the NLP model works as expected. The previous unit tests and integration tests will validate that data is being put into the model properly. However, with system testing, we will assess the accuracy of POS tagging and our predictive model altogether and how it fairs against the software documentation. This process will also make sure that our code is written in an efficient manner in terms of runtime and also that each part of the code is commented out and well explained. The same tools used for unit testing should be used for this as well.

5.5 Regression Testing

Incremental testing will be done in order to ensure that any new additions will not break old functionality. Our team will make sure that if a new component were to be added, that new component would have to be tested extensively before attempting to integrate it with a copy of the current system. Since we will be utilizing Github, a team member will be able to pull the current components we have in place and experiment with adding the new functionality to it. Critical features that cannot break will be mostly surrounded by the way the processor takes in data. If we want to feed it a new type of data, we will have to go back and make changes to the method in which we read/tokenize that data; this is not desirable. These processes of adding components will be driven by our client's requirements and the tools that our client urges us to use.

5.6 Acceptance Testing

Team members will demonstrate that the design requirements are being met by showing our client how we are adapting software documentation to the various word embeddings. Through communicating with one another and meeting at our weekly times, our clients can give us feedback on whether we are on track or not. We would involve our client in the acceptance testing by showing him the progress we have made with training our models. From that point, our client can give us the next set of goals to work on for the next time we meet.

5.7 Security Testing (if applicable)

Our project will not be testing for security, because it revolves around analyzing software documentation and doesn't deal with any private information.

5.8 Results

At this stage in the project, we have done minimal testing. Our focus has been on building and exploring code models. In the future, when we start shaping a final model, we will test to ensure it performs as expected. To ensure compliance with the project requirements, we will rerun tests frequently and with a variety of data inputs. We expect our model to have a similar performance no matter what data is passed. We will also be reporting on points of failure or inconsistencies that we find while testing our model. This part of the project is still in the future so we do not have a concrete plan yet for complete testing, but we will keep it in mind as we move forward.

6 Implementation

Our project implementation is covered in the Design section.

7 Professionalism

7.1 Areas of Responsibility

Area of Responsibility	Definition	NSPE Canon	IEEE Code of Ethics for SE
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	Perform services only in areas of their competence; Avoid deceptive acts.	Profession. Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
Financial Responsibility	Deliver products and services of realizable	Act for each employer or client as	Product. Software engineers shall ensure that their products

	value and at reasonable costs.	faithful agents or trustees.	and related modifications meet the highest professional standards possible.
Communication Honesty	Report work truthfully, without deception, and understandable to stakeholders.	Issue public statements only in an objective and truthful manner; Avoid deceptive acts.	Colleagues. Software engineers shall advance the integrity and reputation of the profession consistent with the public interest. Judgment. Software engineers shall maintain integrity and independence in their professional judgment.
Health, Safety, Well-Being	Minimize risks to the safety, health, and well-being of stakeholders.	Hold paramount the safety, health, and welfare of the public.	Management. Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance. Self. Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

Property Ownership	Respect property, ideas, and information of clients and others.	Act for each employer or client as faithful agents or trustees.	Client and employer. Software engineers shall act in a manner that is in the best interests of their client and employer, consistent with the public interest.
Sustainability	Protect the environment and natural resources locally and globally.		
Social Responsibility	Produce products and services that benefit society and communities.	Conduct themselves honorably, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession.	Public. Software Engineers shall act consistently with the public interest.

1. Work Competence

While there is a lot of overlap on which SE principles cover each area of responsibility, the principle of Profession is most similar to Work Competence. The principle of the profession for software engineers includes work integrity, obeying all laws governing their work, and being honest about the work they are doing. They should take responsibility for any problems caused by their work, and do everything they can to correct their errors. The NSPE codes are similar in that it is expected engineers will not take on work outside their expertise. NSPE also mentions that engineers should not fake their credentials, or do anything to falsely claim their skills.

2. Financial Responsibility

The principle of Product covers financial responsibility. This code of conduct covers the expected deliverables of a software engineer, including the expectation of reasonable cost. Some items covered are ensuring that the client and employee are on the same page when it comes to what work is expected of them and what repayment the employee will receive for their work. There is a realistic expectation of the work that can be done and the cost of that work. The engineer is responsible for meeting those expectations. This is different from NSPE because NSPE only mentions the employee being faithful to their employer. It does not mention the expectation that engineers should be paid fairly for their work.

3. Communication Honesty

There are two SE codes that fit this area, Colleagues and Judgement. Colleagues cover the fact that engineers are expected to be honest about what is their work and what is their colleague's work, listen to their colleague's concerns about the work, and not interfere with their colleague's work. Judgment covers the expectations that engineers will be objective when evaluating software, and tell everyone it concerns when conflicts of interest arise. The NSPE differs in that it requires communicating openly and honestly with the public about the work that is done, as well as being honest about their abilities as an engineer.

4. Health, Safety, Well-Being

There are two SE codes that cover this topic. The principle of Management expects that engineers have enough knowledge about their project so they can properly protect the sensitive information of the client. It expects engineers to behave ethically in all situations. The principle of Self also covers Health, Safety, and Well-Being. First, an engineer is expected to continuously learn how to create better protective systems in software. They are also expected to treat everyone fairly and without prejudice. The NSPE neglects the topics covered in Self but does include the expectations of Management. It expects engineers to always keep sensitive data confidential, as well as never engage in any activity that could put private information at risk, and never take any unlawful action.

5. Property Ownership

The SE code of Client and Employer best covers Property Ownership. It is expected engineers use the property of the client only in authorized ways, including only accessing the data they need to do their jobs, and always with the consent of the client. Keep open communication with the client about any concerns with their deliverables, since the entire project would be considered property of the client. The engineer should also not disclose any confidential information about the client's properties. The NSPE codes also cover these topics. Engineers are expected to only use the property of the client that they are allowed to use for their work.

6. Sustainability

Neither the SE principals nor the NSPE codes have a specific principle covering sustainability. There are parts of some SE principles that relate to the environment, including the Public principle that engineers should not knowingly use software or equipment that significantly harms the environment, and should inform the client about potential dangers to the environment.

7. Social Responsibility

The SE principle dealing with Social Responsibility is the principle of Public. Engineers are responsible for only doing work that will benefit the client or the public. They should not approve of any software that does not meet ethical standards and should notify their client or authorities of possibly dangerous software. They should also answer the concerns of the public with regards to their work, and consider all factors that affect who might use their software, and how it might be used. The NSPE codes cover these expectations and expect engineers to accept personal responsibility for their actions, and always act in an ethical manner that benefits the public.

7.2 Project Specific Professional Responsibility Areas

1. Work Competence

Work competence is needed for this project. Competence in Python, natural language processing, machine learning, and various NLP libraries, are all needed to implement the project successfully and to create meaningful results. Our team is performing high in work competence, we are creating working NLP models with different libraries. These models generate data on how NLP works with software documentation.

2. Financial Responsibility

Financial Responsibility does not apply to this project. Our team is not being paid or spending money to complete this project. Therefore, there is nothing we have to be financially responsible for.

3. Communication Honesty

Communication Honesty is important to this project. If we were to falsify the results of this project, it would hurt the interest of the client, and could also affect others who choose to research the same topic in the future.

4. Health, Safety, Well-Being

Health, safety, and well-being are relevant to this project. The health of the team and the client is always important, as a healthy team will create better results. The team is also responsible for hosting meetings online and working remotely.

5. Property Ownership

It is important that we responsibly handle the data/information given to us by the client. The client has trusted us with Python code and software documentation data that they developed with their research and it is important that we respect the data and only use it with the consent of the client.

6. Sustainability

This project is being worked on with sustainability in mind. Our team and client recognize that software documentation and tools used to analyze it and break it down are ever-changing. With that in mind, we develop this project in such a way that allows frequent and constant change and maintenance. Our team is performing at a high level when it comes to ensuring sustainability throughout the project.

7. Social Responsibility

There have been instances in the past where Artificial intelligence has been used maliciously. It is our responsibility to only use our software in the way the client wants it and ensure that none of the technologies we use can negatively impact the public.

7.3 Most Applicable Professional Responsibility Area

One area of professional responsibility that is important to our project would be work competence. For the scope of this project, it is vital that group members understand the workings of natural processing languages, and how to compare them in Python. If group members are not able to perform good work competence, our end goal of selecting a word embedding to improve on would not happen. Throughout the semester, we have all shown good work competence, by understanding the differences between word embeddings, and working with the various NLP libraries. By following this professional responsibility area, we have been able to craft valid results that will prove beneficial to the overall goal of the project. All members will continue to strive to perform valid work competence, so we can provide satisfactory results to our clients.

8 Closing Material

8.1 Discussion

So far, our project has been primarily focused on testing different Natural Language Processing technologies and assessing their effects on processing and POS tagging Software Documentation. Based on this semester's experimentation, there has been repeated evidence indicating that current NLP libraries can neither process nor POS tag the data accurately. We have used models such as Spacy and NLTK to POS tag the data and the tags are inaccurate for a majority of the time. However, when we use these technologies to POS tag normal English documentation, it does it with high accuracy. Additionally, we have used these technologies to see if they can vectorize individual words in the documents and compare the words' meaning to other words in the document. Overall, these technologies are highly inaccurate in understanding software documentation but are highly accurate in understanding standard English documentation.

8.2 Conclusion

Up to this point, our group has conducted research and experiments on different NLP topics. Our ultimate goal for this project is to improve NLP models to process software documentation data more accurately. This semester we aimed to understand and create models for different Natural Language Processing Libraries, Neural Networks, Data Preprocessing techniques, Vectorization strategies, Python, and Data Science Libraries. Our group decided to use a waterfall project management style to help us achieve our goals. Our group was able to gain an understanding of all of these topics but were not able to create working models for all of them. Our biggest constraint was time. One thing that could have been done differently is spending less time working on models for certain topics to allow more time to create new models.

8.3 References

[1] F. N. A. Al Omran and C. Treude, "Choosing an NLP Library for Analyzing Software Documentation: A Systematic Literature Review and a Series of Experiments," 2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR), 2017, pp. 187-197, doi: 10.1109/MSR.2017.42

8.4 Appendices

Any additional information that would be helpful to the evaluation of your design document.

If you have any large graphs, tables, or similar data that does not directly pertain to the problem but helps support it, include it here. This would also be a good area to include hardware/software manuals used. May include CAD files, circuit schematics, layout etc,. PCB testing issues etc., Software bugs etc.

8.4.1 Team Contract

Team Members:

1) ____Austin Buller_____ 2) _____Kelly Jacobson_____

3) ____ Jacob Kinser_____ 4) _____Sam Moore_____

5) _____William Sengstock_____ 6) _____Zach Witte_____

7) _____Dan Vasudevan _____

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:

   Meeting with TA Jacob Conn every Thursday at 5:30 pm virtually. These meetings will happen on the Discord server.

   Meeting with Hung Phan and Hiep Vo every Thursday at 6:00 pm on Webex

   Link: https://iastate.webex.com/meet/vohi01

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):

   We will use Discord to communicate. It is expected that everyone will check in on Discord at least once a day. The TA is also a member of the server so he can check in on us and we can ask him questions there.

3. Decision-making policy (e.g., consensus, majority vote):

   For decisions that we cannot all agree on, we will use the majority vote to decide what to do. At the end of the day, the best decisions are the ones that help the group as a whole.

4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):

   All useful documents will be shared in the team's Google Drive. While the client will be writing their own meeting notes, Kelly has taken on the role of writing and sharing another set of meeting notes for client meetings. These will be available in the Meeting Notes folder of the shared Drive.

   Client-shared files will be available on the team Github.

Link: [hiepvo01/Senior-Design-491: Analyze POS tags and Word Embeddings for Software Documentations (github.com)]

This will also be our central repository for code development under the management of the client, Hiep Vo.

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:

It is expected that every team member attends every general team meeting, TA meeting, and client meeting arrives on time and participates to the best of their ability. If a team member cannot attend a meeting for any reason, they should inform the team at least a day before.

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:

Each team member is expected to complete their fair share of work that is given by the TA (Andrew Vo). Each sub-team will present a presentation of what they have learned or worked on during the weekly TA meeting. If assignments are not completed by the deadline, they should be completed within the next 48 hours of the deadline.

3. Expected level of communication with other team members:

Individuals should communicate with the team on Discord at least once a week and should communicate with their sub-teams every 2-3 days when working on the sub-team assignments.

4. Expected level of commitment to team decisions and tasks:

Each team member should be fully committed to the team decisions and tasks. Every individual should put enough effort into the project to result in quality work for every task.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):

Team Leader: William Sengstock

Team Organization: Kelly Jacobson

Component Design: All members

When it comes to the discussion with our TA/client, all group members will be expected to participate in what they have done in the past week. That way all members will be contributing to the project, and by doing so we can make sure that everyone is keeping track of what the group has to get done as a whole. For every meeting that we have, William will also keep track of how long the meetings take place.

As in terms of turning in assignments, before a member submits the assignment it will be looked over to be sure that nothing had been left out. Using Discord, the group will be able to communicate with one another to ensure that all parts of the assignment are completed in a satisfactory manner. As the semester progresses, more specific roles may be introduced to other members to make sure the group stays on top of the current goal.

2. Strategies for supporting and guiding the work of all team members:

With all our group members having classes at different times, we will use Discord and constant communication to support and guide the work of one another. If anyone is having trouble or needs a question answered, Discord will serve as a way to ask those questions and get feedback from the other team members.

3. Strategies for recognizing the contributions of all team members:

During the group's weekly meetings the contributions of all members will be discussed. That way everyone gets a chance to show their progress and everyone else gets to see what they have done.

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the Team.

Austin Buller - Software Engineering

I am experienced with Java, C, MySQL, and learning Python this semester. I'm looking forward to learning about the topics covered in this project, especially the machine learning aspect, which is something I've always thought was interesting.

Kelly Jacobson - Software Engineering

I know Java and Python, though I know Java better. I have experience with mySQL and SQL in a professional environment and have outside-of-class experience with database management. I'm currently taking Com S 472: Principles of Artificial Intelligence so hopefully stuff I learn in that class will help with this project too. I would not say I'm an expert on anything, but I am willing to learn! I listed this project as one of

my preferences because I have an interest in languages and linguistics, and I want to learn more about natural language processing and machine learning.

Jacob Kinser - Software Engineering

I am experienced most in Java and C, I have used python before but I am not as experienced in it. Although I am willing to learn more Python for this project. I also have experience with SQL through multiple classes taken at ISU.

Although this project was not on my preferred projects list, I look forward to working with the team and working towards our main goals in this course.

Samuel Moore - Software Engineering

I am experienced in using the languages Java, C++, Swift, Python, SQL, Git, C, and JavaScript. I believe that I can provide an efficient, driven, and experienced perspective to this project. I am looking forward to learning more about this topic and I am excited to get to work.

William Sengstock - Software Engineering

I am experienced in Java and C, but I will also try to learn the basics of Python because it is one of the main languages that we will use. I also have experience with SQL, because I have used it in prior classes. Although I did not list this project as one of my preferences, after our first meeting I found the topic interesting and I am ready to dive into it.

Dan Vasudevan - Computer Engineering

I am most experienced in Python and Java but have also used C, SQL and Javascript. I don't have much experience with NLP but I have done some work with deep learning specifically related to Spiking and Convolutional Neural Networks as an undergraduate research assistant. The tools I used included the Brian library, tensorflow/keras and python data science libraries like NumPy MatPlotLib. I also have experience scripting in python from my cloud security internship. I have always been interested in the application of NLP into our daily lives and I am excited to learn more in this project.

Zach Witte - Software Engineering

I am most experienced in Java, C, and C++ for coding languages. I have some experience using JavaScript as well. I am learning a bit of python for other classes I have this semester, and I am willing to learn more for this project. I have experience with SQL in a professional environment, and I am taking a class over database management this semester as well. I also have experience using Git.

2. Strategies for encouraging and support contributions and ideas from all team members:

- Make sure everyone has a chance to talk and a chance to give their ideas and feedback
- Take all ideas into consideration. We want to encourage discussion and want to make sure everyone knows that their input is important.
- Give everyone a leadership role based on their strengths and interests. If everyone has a role, it will foster a more collaborative and friendly environment.

3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)

Most of our communication will be through discord. Team members can raise their concerns by messaging our discord server. We will then set up a meeting where the whole team can listen to the questions or concerns of the team member. We can then discuss what actions we will take to make sure said team member feels included and comfortable in our work environment.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:

1. Our most important team goal is to deliver a successful project in the form of new research into natural language processing for software documents that satisfy our clients' requests.
2. We want to follow this predetermined contract to the best of our ability because we believe it will guide our team to success throughout this semester and next semester.
3. We want to become comfortable with each other as team members so no one hesitates to reach out for help, guidance, or support.
4. We want to be able to execute every task that our client/mentor throws at us with efficient, honest, and relentless work.
5. Finally, we, as a team, want to exceed the expectations of our client/mentor and deliver a project that we are genuinely proud of.

2. Strategies for planning and assigning individual and team work:

This project will bring many specific tasks that will require different levels of interest, experience, and hours put in. This level of demand requires a strategic and dynamic approach. When it comes to assigning individuals and teamwork, we will use a volunteer/voting strategy to make a decision. This way, we can assure that the task at hand is being dealt with by a team member or multiple team members that are passionate about that specific task and have experience that may help with the completion of said task.

Planning the execution of the task will be done by those who are assigned to the task. Our goal is to create thorough steps in order to execute. This will involve breaking down the task at hand studying the best ways to go about completing it through research, consultation with our mentor/client, and team meetings.

3. Strategies for keeping on task:

Keeping on task will be very important as we are taking on more and more work. During the planning phase for the execution of a task, a timeline will be created. This will be a strict timeline and it must be followed to ensure our team is staying on track. Regular checkups with each other will also take place over Discord to emphasize the importance of staying accountable as we progress through the project this semester and the following semester.

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?

We will have a team meeting to discuss the infractions to try and resolve them in an effective and positive way.

2. What will your team do if the infractions continue?

Any continuing infractions we will bring up with our TA to help us come up with a solution. If we can not find a solution there we will bring up the issue with our professor.

****

a) I participated in formulating the standards, roles, and procedures as stated in this contract.b) I understand that I am obligated to abide by these terms and conditions.

c) I understand that if I do not abide by these terms and conditions, I will suffer the

consequences as stated in this contract.

1) _____Jacob Kinser_____DATE ____9/17/2021_____

2) _____Austin Buller_____DATE _____9/17/2021_____

3) _____Kelly Jacobson_____ DATE _____9/19/2021____

4) _____Sam Moore_____DATE _____9/19/2021____

5) _____Zach Witte_____DATE _____9/19/2021____

6) _____William Sengstock_____DATE _____9/19/2021____

7) _____Dan Vasudevan_____ DATE _____9/19/21_____