Creating an automated question responder for NMSU Computer Science

Nicholas Grijalva

December 1, 2021

1 Introduction

The front desk of Computer Science Department at NMSU often needs to answer many routine questions. The amount of questions that the front desk receives can be reduced by providing answers to common questions to students before contacting the front desk. This can reduce the amount of work that both the front desk and the students with questions need to do. My goal is to create a model using the current Computer Science Graduate FAQ questions[1][2] and answers to provide students the most related candid answer given their question.

2 Solution

I created 3 models total for attempting to provide candid answers given a question. Candid answers and questions are those provided by the Computer Science Departments FAQ[1][2].

2.1 Model 1: Cosine Similarity

My first model utilized Python 3's sentence similarity library [4]. For the similarity measure, I used cosine similarity. The model works by using the student's question (string) and taking the cosine similarity between the student's question and every candid question. The candid answer given to the student is the one whose respective candid question receives the highest cosine similarity with the student's question. A variation of this model was tested with euclidean distance instead of cosine similarity, but the results did not show any noticeable difference.

This model did not perform well enough for the situation of an FAQ system. It is unable to extrapolate complicated questions to the candid questions.

2.2 Model 2: Weighted Cosine Similarity

This model is similar to the first but uses the cosine similarity between student's proposed question and all of the candid answers in addition to the candid questions. The value for each candid answer/question pair is a chosen parameter ratio that signifies what percentage of each (cosine similarity with the question and cosine similarity with the answer) will contribute to the final value. From these combined values, the highest is chosen and the candid answer is given as an answer.

The performance of this model turned out to be worse than my first model. From this I learned that the candid answers are not as closely related to questions than I had anticipated.

2.3 Model 3: Synonym Comparison

My final model for this submission takes a different approach from the others. My goal is to find all the synonyms of every word in all of the questions. I call

these groups of synonyms for each question a 'syn bucket'. Utilizing the Natural Language Toolkit (NLTK) library [3], the preprocessing that I conducted for each syn bucket includes creating a list of all the synonyms related to a candid question, removing a set of unimportant connection words, setting all words to lower case, and stemming the words to their root. After obtaining this refined syn bucket, my model compares the bucket to the student's proposed question. The syn bucket that shares the most amount of words with the student's question is chosen. If there is a tie for the highest then it is broken by using the cosine similarity as Model 1. The model returns the indices to the top three candidate question/answer pairs.

From my testing, this is the best performing model. There is still a bit of more work to be done. Some of the improvements include expanding the synonym pool and developing a better similarity measure for sentences.

3 Prototype

My prototype of Model 3 consists of all of the necessary preprocessing for the data set as well as downloading all of the required libraries. The actual prototype is a user input driven simulation that takes in the user's question and then gives three candid questions that may be related to the initial question. The user is then prompted to choose one of the questions that they believe is most closely related to their question. After choosing, the user is given the candid answer for the candid question they chose. The user is also given the option to reject the given candid comparison questions and try re-wording their question. The model runs on a loop to answer as many questions as the user wants to ask until an exit sequence is given.

4 Reflection

I enjoyed working on this project. It was a good real-world problem to apply the material that is taught in the Machine Learning course. It was a challenge to figure out how to deal with the small data set and a refreshing break from developing the more standard types of machine learning models. I am not sure if I would be able to greatly expand on this project with my current knowledge. I believe I would have to further study natural language processing to be able to adequately improve the performance of these models.

References

[1] NMSU Department of Computer Science. https://computerscience.nmsu.edu/grad-students/prospective-graduates.html#FAQs, N/A. [Online].

[2] NMSU Department of Computer Science. https://computerscience.nmsu.edu/grad-students/graduate-student-faq.html, N/A. [Online].

[3] NLTK Project. https://www.nltk.org/, 2021. [Online].

[4] Susheel-1999. https://pypi.org/project/sentence-similarity/, 2021. [Online].