Naïve Bayes Chatbot to Provide Probabilistic Responses to Slay the Spire Events

Seth Friesen
Computer Science
University of Texas at Dallas
Richardson, USA
sbf200002@utdallas.edu

Abstract

This report introduces a chatbot designed to assist players in their approach to the complex decision-making process within the game "Slay the Spire" (STS). Many games, including STS, present players with ambiguous decisions. The effects of these decisions are not always immediately apparent. The chatbot aims to mitigate this uncertainty by leveraging statistical data on the game's events. The model is based on the concept of naive Bayes, which is used for event detection. Also, probabilistic selection is used for response generation. Despite its current limitations, the chatbot demonstrates capabilities in assisting players with decision-making about events in STS.

Problem Description

Many games present the player with choices that have ambiguous or unknown answers from the player's perspective. Players are often blind to the consequences of their choices and are only really making their best guess. The use of chatbots trained on statistics about the choices games present can help alleviate this blindness. The goal of this developed chatbot is to capture requests from the user about the game Slay the Spire (STS) and respond to them in a reasonable manner. STS is one of these games of choices. The game essentially boils down to a sequence of events one after another. The chatbot's purpose is to be able to provide responses to any event the user may want help on. To do this, the chatbot must be able to recognize and capture the type of event that is happening. Additionally, the chatbot must know the best choices for the events, so that it may aid the player in the best way possible.

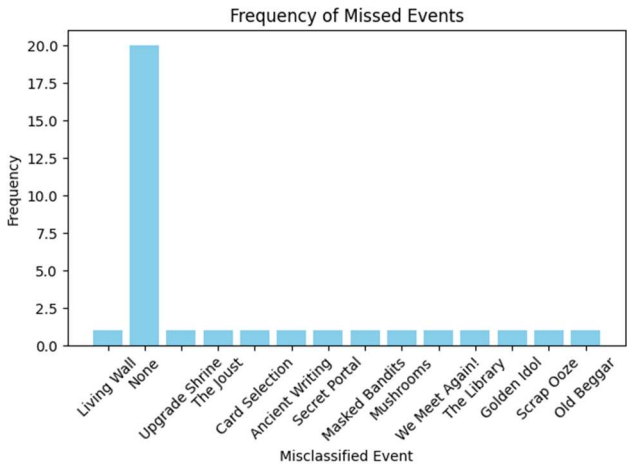
Approach

The approach that was taken for this chatbot was to use naïve bayes for event detection, and probabilistic selection for the response to the event. In order to help the user, the chatbot must be capable of parsing the request and identifying the problem. To do so, named entities were created to help capture the details of the requests. These named entities allowed the model to recognize the names of events in STS that were explicitly named.

However, if an event was not explicitly named, the chatbot still needs to be able to recognize the event. For example, if the user writes about the choices they are presented with instead of the name of the event. In this case, naïve bayes was used to identify the event. A table was constructed of bigram counts between events and the words correlated with them. Given words from the choices of the event or words about the event, the model can compute the most likely event using by accessing the bigram table. To help increase the probabilities of the likely events, the sum of logs property is used. That is, instead of multiplying probabilities, their logs are summed and then used as an exponent. This increases the likelihood of events and minimizes extremely small probabilities. Additionally, words that are unrelated to the events, i.e. articles and "fluff" words, are disregarded as they will only have a negative impact on determining the most likely events. The event with the greatest probability given the user prompt is the one selected for response. Should there be no event with a reasonably high probability, the chatbot proceeds with its best guess, but will warn the user of its lack of information and respond with a request for more information.

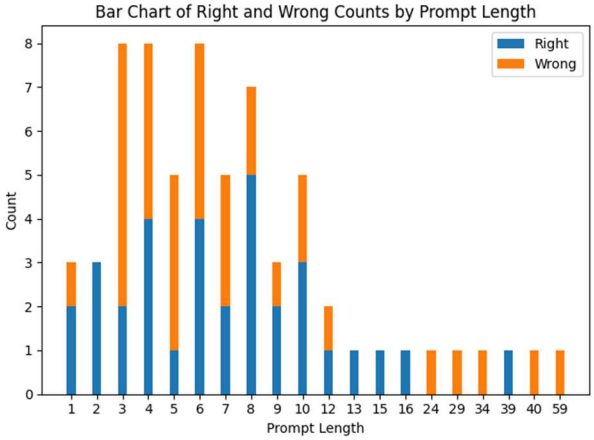
After an event is selected, the problem becomes what choice to make for the event. In STS, decisions are correlated with a win percentage for the run or the current playthrough. Essentially, the better a choice is, the more likely it will be to lead to a playthrough being successful. Just because a decision has a higher win-rate doesn't mean that it should be chosen exclusively, however. The model calculates the win-rates for each decision for the event. It then adjusts them to slightly favor decisions that are chosen more often and punish decisions that have a low pick-rate. This is done to prevent decisions with low pick-rates and high win-rates from being favored because of low sample size. After adjusting the win-rates, the model randomly chooses a decision based on the adjusted win-rate probabilities. Choices with higher adjusted win-rates will be chosen more often, but the chatbot is not restricted to exclusively choosing the most "optimal" decision every time. This is because the most "optimal" decision is not actually optimal for every situation and allowing probabilistic decision making accounts for this fact.

Figure 1: Bar Graph Depicting the Number of Times an Event is Misclassified



Data Description

All of the data taken for this chatbot is found on the SpireStars web app. This web app allows players to submit successful and unsuccessful runs to be dissected and converted into win-rates for cards, events, relics, and other



STS happenings. Every choice made in submitted runs contributes to the win-rates for those choices kept on the website.

The model uses data on events and cards. Specifically, the win-rates for decisions in events and the win-rates for choosing or skipping certain cards. This data is converted from the SpireStars web app into text files that the model references for its win-rate calculations.

Experiments and Error Analysis

Currently, the main issue with the chatbot is its inability to handle non-STS related prompts. Even prompts that are STS related but have excess context or "fluff" are hard to decipher. The chatbot was tested against sample prompts and compared against their expected outputs. The predominant issue was the chatbot incorrectly tagging something as irrelevant as seen in Figure 1. Here we can see that if an event was misclassified it was most likely misclassified as 'None'. Meaning, that chatbot was prompted for an event that should have been recognize but instead the chatbot identified it as unrelated to STS. This is likely because the chatbot requires specific, correctly spelled, words in order to correctly identify an event. Misspellings and synonyms to expected words lead to the chatbot not understanding the prompt. Prompt length may also correspond to misidentifications as seen in Figure 2. Very small prompt lengths and extremely long prompt lengths are both

Figure2: Right and Wrong Classifications by Prompt Length in Words

associated with more wrong classifications than prompts with more middling word counts, which is the intended use.

Despite these flaws, the bot is exceptionally good at identifying events given relevant wordings. The testing data used intentionally obscure and irrelevant information to trick the model. When being used correctly, the model performs well at identification. Additionally, warning prompts will occur when prompts contain superfluous information or if a prompt does not contain enough information. Lastly, choices are correctly being chosen more often when their win-rates are higher. That is to say, the model's probabilistic decision making is acting as intended.

Conclusion and Future Work

In conclusion, the STS chatbot uses naïve bayes to identify events, and chooses responses probabilistically. The chatbot responds very well when it receives relevant input and not so well when it receives prompts completely unrelated to STS. The chatbot still has room to improve. Aside from tackling the aforementioned problems, the chatbot can also be expanded to currently not supported decisions. These include events like shops where many items can be chosen at once, and relic decisions where the character is improved in unique ways. In its current state however, the chatbot is capable of identifying and responding to prompts about events.

References

[1] Murphy, K. P. (2006). Naive Bayes classifiers - datajobs.com. https://datajobs.com/data-science-repo/Naive-Bayes-%5BKevin-Murphy%5D.pdf

[2] Rish, I. (2001). *CITESEERX*. CiteSeerX. https://citeseerx.ist.psu.edu/

[3] Shimarisu. (2024). *Slay the spire statistics*. SpireStars. https://spirestars.web.app/