Project Proposal

Note: This proposal will be continuously developed past the due date to ensure a focused approach is outlined in detail. Further changes will be discussed within the group and brought forth to Dr. Anton.

Game Representation

Apples to Apples involves red and green “apple cards.” Each player picks up seven red noun cards, which become their “hand.” The judge places a green card on the table, which represents an adjective that the players will try to match with a red noun card from their hand. After each player plays a card, the winner of the round is chosen by the judge.

Our agent should attempt to win the round by choosing the card that best fits with the green adjective card.

Cards should be read from an external file and represented as strings in the program. This will allow the imported cards to be varied or expanded without much change to the agent program. The cards will be represented as strings because it will allow the agent to easily access the card data for the input sensors to access all the information needed to maximize the performance measure.

Using strings in C will require <string.h> to be included in the program files. Strings should make the cards' representation as simple and efficient as possible while representing the cards accurately.

After representing the cards in the program, we should be able to represent the hand as a larger datatype that can include up to 7 card datatypes.

If the agent is a judge and all players have chosen their red card, it will choose the best one as if it were a player. In this way, the agent can play as both a judge and a player.

Environment

The judging player experiences a fully observable environment, while the non-judging players experiences a partially observable environment.

Agent Logic

As a Judge:
The Judge should act similarly to the player. Choosing the “best” option from a series of red cards based on a green card. The judge differs from the player in that the cards they choose from are not their hand but the cards selected by the players.

As a Player:
Simply put, the player agent should be able to take input as a hand of red cards and the relevant green card and output the red that best matches the green card. To decide the “best” card, the agent should take the input data from the sensors and the card data and use a method to determine the card to maximize the performance measure. Our agent aims to determine a sort of “semantic text similarity” between simple texts – single words.

To do this, there are a few options we could use:

- Randomness
    - The agent could determine the best card by generating a random number between 1 and the number of cards in the hand and playing that card. This method cannot improve or maximize the performance measure because the results will always be random.
- Evaluation Functions
    - Deep learning
        - Using deep learning, the agent could train on sample data and build a neural network to improve at matching noun cards to adjective cards.
    - Using an external API
        - Many NLP and other models could be helpful for our agent to match the noun cards to the adjective cards, such as ChatGTP or Copilot.

Ideally, the method we opt for makes the agent autonomous, i.e., its behavior results from its ability to learn and adapt. The problem of semantic matching has two modern approaches based on deep learning:

- Representation learning
- Matching function learning

  (Both of these will be expanded upon further literature review).

Considering the vast literature, a single approach will be chosen based on the project's time constraints and the group's motivations.

Efficiency

We can use some of the strategies discussed in class to improve the evaluation functions using minimax or alpha-beta pruning. Other efficiency improvements could have to do with training datasets in the case of deep learning.

Performance Measure

The performance measure for our agent should be the similarity/compatibility between the green adjective card and the red noun cards in their hand. In doing this, the player should be able to select the “best” option to maximize their chance to win the round.

Programming Language

C language that will be fast and efficient, but also one that we have experience developing in, and should not require an outsized effort from any group member to completely learn a new language.

Group Communication

Group communication and version control will be important for this project. For this reason, we will communicate via a group chat, and the version control will use Git and GitHub.

Resources

[1] Wikipedia contributors. (2024, January 2). *Apples to apples*. Wikipedia.

https://en.wikipedia.org/wiki/Apples_to_Apples

[2] *Chessprogramming wiki*. (n.d.). https://www.chessprogramming.org/Main_Page

[3] *Semantic matching - Semantic matching*. (n.d.).

http://semanticmatching.eu/semantic-matching.html

[4] Naveenkumar, K. (2021, December 16). Deep learning for semantic text matching - towards

data science. *Medium*.

https://towardsdatascience.com/deep-learning-for-semantic-text-matching-d4df6c2cf4c5

[5] Xu, Jun, Xiangnan He, and Hang Li. "Deep Learning for Matching in Search and Recommendation." (2018).
Deep Learning for Matching in Search and Recommendation

[6] Hu, Baotian, et al. "Convolutional neural network architectures for matching natural language sentences." Advances in neural information processing systems 27 (2014).
Convolutional neural network architectures for matching natural language sentences