Application of Game Theory to Analysis of Machine versus Human Strategies

New Mexico

Supercomputing Challenge

Final Report

April 1, 2026l

Welch Homeschool

Team Members:

Helena Welch

Kalliope Luna Welch

Teacher:

Cindy Welch

Project Mentor:

Paul Welch

Executive Summary

As machine learning becomes increasingly prevalent in our current society, it is assuming a more and more influential role upon individuals' decisions. The interplay of such decisions made by different players affects many aspects of life, including finance and politics. Machine learning algorithms have the potential to aid in these decisions, but only when examined first to ensure that their choices are rational.

This study aims to evaluate whether or not a game theoretical simulation of human interaction, the Consensus Game, can improve large language model (LLM) decision-making capabilities. We find that the Consensus Game tends to improve results for LLMs with small parameter sizes and for questions involving the Nash Equilibrium. This indicates that the Consensus Game performs well in simulating a human conversation in which multiple players with distinct mentalities reach a solution benefitting both players, thus modeling human decision-making processes more effectively than raw LLM output.

Introduction

Role of LLMs in Decision Making

As machine learning becomes increasingly prevalent in our current society, it is assuming a more and more influential role upon individuals' decisions. The interplay of such decisions made by different players affect many aspects of life, from finance and politics to biological systems(kind of unclear, don't have an alternative except just cutting what comes after life). [**?**] Machine learning algorithms have the potential to aid in these decisions, but only when examined first to ensure that their own choices are rational.

Purpose of Project

By the end of this project, we expect to understand whether the Consensus Game improves the accuracy of LLMs in answering fact-based versus game-theory-based questions. In addition, we hope to identify differences in various LLMs' initial and final accuracies, based on their parameter sizes, and evaluate why certain question-model pairs improve to a greater extent.

Prior Works

Several studies have examined the processes of LLMS in decision-making compared to the decision making processes of humans. In addition, Jacob et al. first developed and introduced the Consensus Game. They tested it using two models, Llama 7B and 13B, to answer knowledge-based and problem-solving questions.

Other works include Rasal, Sumedh & Hauer 2024 [1], which introduces a methodology for improving LLM decision-making by prompting it with multiple choice questions, and Xiao & Wang 2025 [4], which concluded that humans are better at finding the Nash Equilibrium than LLMs.

Methods

First, we pass a series of knowledge-based questions and game theoretical scenarios to two separate instances of an LLM, the Generator and Discriminator. The Generator is prompted with several possible answers to a question, while the Discriminator must choose whether the Generator's answer is correct. Each question is asked 100 times, creating a probability distribution of answers for each instance of each LLM queried. These probability distributions, known as the policies,

model	parameters	vocabulary size
Mistral-small [3]	22 billion	32k
Command-r [1]	35 billion	128k
Llama 3.1 8b	8 billion	64 k
Mistral-nemo [2]	12 billion	128k

Table 1: Characteristics of the large-language models used.

are then updated 5,000 times based on each other's previous policies, simulating a 'conversation' between the Generator and Discriminator. The accuracies of the initial and final policies are then calculated for each data set and question.

Data Collection and Preparation

4 LLMs with varying parameter spaces were used to compare variability in answer choices between machines. Following Jacob et al., zero-shot prompting was used, so no examples were given before the actual question was passed. We found that LLMs mostly gave clean answers with this format, although LLMs with smaller vocab sizes (llama 3.1 8b and mistral-small) sometimes produced words surrounding their answers rather than the single digit asked for.

Prompts are given below:

Generator prompt: "[Question] Please choose an answer from the following messages: [Answer Choices] Only give the number of your answer, 1, 2, 3, or 4, with no other text surrounding it. Do not provide any other notes or commentary."

Discriminator prompt: "[Question] Is the following the correct answer to this question: [Answer Choice] Provide ONLY a single digit response. State 0 for true and 1 for false. No further comments are allowed."

We also attempted to use Chat-GPT 5o, a larger model, but were banned for asking it the same knowledge-based questions repeatedly as if we were collecting data to train our own model.

Knowledge-Based and Decision-Based Queries

Deriving the Consensus Game

The Consensus Game was derived per Jacob et al. and .

Having collected 100 answers for every question from the Generator and Discriminator each, we calculate the probability that a given answer was chosen and use this to define the initial policy π_G and π_D. From here we can define a standard utility, or the amount of payoff for each right answer from both the Generator and Discriminator for a given question.

$u_G(\pi_G, \pi_D) = \frac{1}{2} \sum_{v \in \text{correct, incorrect}} \sum_{y \in Y} \pi_G(y|x, \nu) \cdot \pi_D^{(t)}(\nu|x, y)$

We regularize this utility with the Kullback-Leibler Divergence, which prevents the policies from updating and diverging too rapidly from the previous policy.

This divergence term is analogous to negative entropy in statistical mechanics.

$D_{KL}[\pi_G(\cdot|x, \nu)||\pi_G^{(1)}(\cdot|x, \nu)] = \pi_G \log(\frac{\pi_G}{\pi_G^{(1)}})$

Negative entropy: $S = \sum_i p_i \log p_i$ (see ref.[24])

This new definition of the utility is analogous to the Gibbs free energy of statistical mechanics.

$\tilde{u}_G(\pi_G, \pi_D) = u_G - \lambda_G D_{KL}[\pi_G(\cdot|x, \nu)||\pi_G^{(1)}(\cdot|x, \nu)]$

The regret, or how much the Generator and Discriminator are penalized for a given is defined below as the maximum difference of current and ideal utilities.

$\text{Reg}_G^{(T)} = \max_{\pi^* \in \triangle Y} \sum_{t=1}^{T} (u^{(t)}(\pi_G^*|x, \nu) - u_G^{(t)}(\pi_G^{(t)}(\cdot|x, \nu)|x, \nu))$

$\pi_G(y|x, \nu) \propto \exp(\text{Reg}_G^{(T)})$

$\pi^{t+1} = \arg\max_{\pi \in \triangle Y} \{(\sum_{t'=1}^{t} \tilde{u}_G(\pi_G, \pi_D)) - \frac{1}{\eta} \sum_{y \in Y} \log \pi(y|x, \nu)\}$ Follow-the-Regularized Leader algorithm

guarantees a bound on the regret

$= \arg\max_{\pi \in \triangle Y} \{\eta(\pi_G u_G - \lambda_G \pi_G \log(\frac{\pi_G}{\pi_G^{(1)}})) - \sum_{y \in Y} \log \pi(y|x, \nu)\}$

$= \arg\max_{\pi \in \triangle Y} \{\eta \sum_{\pi_G} (t\lambda_G \log \pi_G^{(1)} + \sum_{t'=1}^{(t)} u_G)\pi(y|x, \nu) - (1 + t\lambda_G \eta) \sum_{y \in Y} \pi(y|x, \nu) \log \pi(y|x, \nu)\}$

$= \arg\max_{\frac{\eta}{1+t\lambda_G \eta} \pi \in \triangle Y} \{\eta \sum_{y \in Y} (t\lambda_G \log \pi_G^{(1)} + \sum_{t'=1}^{(t)} u_G)\pi(y|x, \nu) - \sum_{y \in Y} \pi(y|x, \nu) \log \pi(y|x, \nu)\}$

Uses max entropy

Softmax Function

$w^{t+1}(\pi_G) = \frac{\eta}{1+t\lambda_G \eta} \sum_{y \in Y} (t\lambda_G \log \pi_G^{(1)} + \sum_{t'=1}^{t} \tilde{u}_G)$ Softmax is the analog of the Boltzmann distribution

$\pi_G^{t+1} = \frac{\exp\{w^{t+1}\}}{\sum \exp\{w\}}$

$\pi_G^{(t+1)}(y|x, \nu) \propto \exp\{\frac{Q_G^{(t)}(y|x,\nu) + \lambda_G \log \pi_G^{(1)}(y|x,\nu)}{1/(t\eta_G) + \lambda_G}\}$

Final Equations

The final equations are given based on the above derivation:

Initial Policies

$$\pi_G^{(1)}(y|x,\nu) = \frac{P_G(y|x,\nu)}{\sum_{y'}\sum_{\nu'} P_G(y'|x,\nu')} \quad \textbf{(1a)}$$

$$\pi_D^{(1)}(\nu|x,y) = \frac{P_D(\nu|x,y)}{\sum_{\nu'}\sum_{y'} P_D(\nu'|x,y')} \quad \textbf{(1b)}$$

Policy Updates

$$\pi_G^{(t+1)}(y|x,\nu) = \frac{e(t,Q_G^{(t)},\pi_G^{(1)})}{\sum_{y'} e(t,Q_G^{(t)},\pi_G^{(1)})} \quad \textbf{(2a)}$$

$$\pi_D^{(t+1)}(\nu|x,y) = \frac{e(t,Q_D^{(t)},\pi_D^{(1)})}{\sum_{\nu'} e(t,Q_D^{(t)},\pi_D^{(1)})} \quad \textbf{(2b)}$$

where:

$$e(t,Q_G^{(t)},\pi_G^{(1)}) = \exp\{\frac{Q_G^{(t)}(y|x,\nu)+\lambda_G \log \pi_G^{(1)}(y|x,\nu)}{1/(t\eta_G)+\lambda_G}\}$$

$$e(t,Q_D^{(t)},\pi_D^{(1)}) = \exp\{\frac{Q_D^{(t)}(\nu|x,y)+\lambda_D \log \pi_D^{(1)}(\nu|x,y)}{1/(t\eta_D)+\lambda_D}\}$$

$$Q_G^{(t)}(y|x,\nu) := \frac{1}{2t}\sum_{\tau=1}^{t} \pi_D^{(\tau)}(\nu|x,y) \quad \textbf{(3a)}$$

$$Q_D^{(t)}(\nu|x,y) := \frac{1}{2t}\sum_{\tau=1}^{t} \pi_G^{(\tau)}(y|x,\nu) \quad \textbf{(3b)}$$

Calculating Accuracies

$$S_G(t) = \frac{\sum_{x'} \pi_G^{(t)}(y^*_{x'}|x',\nu_c)}{N_X} \quad \textbf{(4a)}$$

$$S_D(t) = \frac{\sum_{x'} \pi_D^{(t)}(\nu^*|x',\nu_c)}{N_X} \quad \textbf{(4b)}$$

Definitions

$x =$ question

$y =$ answer choice

$\nu =$ correctness parameter

$t =$ update run

$P =$ probability distribution

$\pi^{(t)} =$ policy on a given run

$Q =$ running average of π

$y^* =$ accurate answer choice

$\nu^* =$ accurate correctness

$N_x =$ number of questions asked

$S =$ accuracy

$\tilde{u} =$ utility

$D_{KL} =$ Kullback-Leibler Divergence

λ determines amount of regularization

Reg $=$ regret

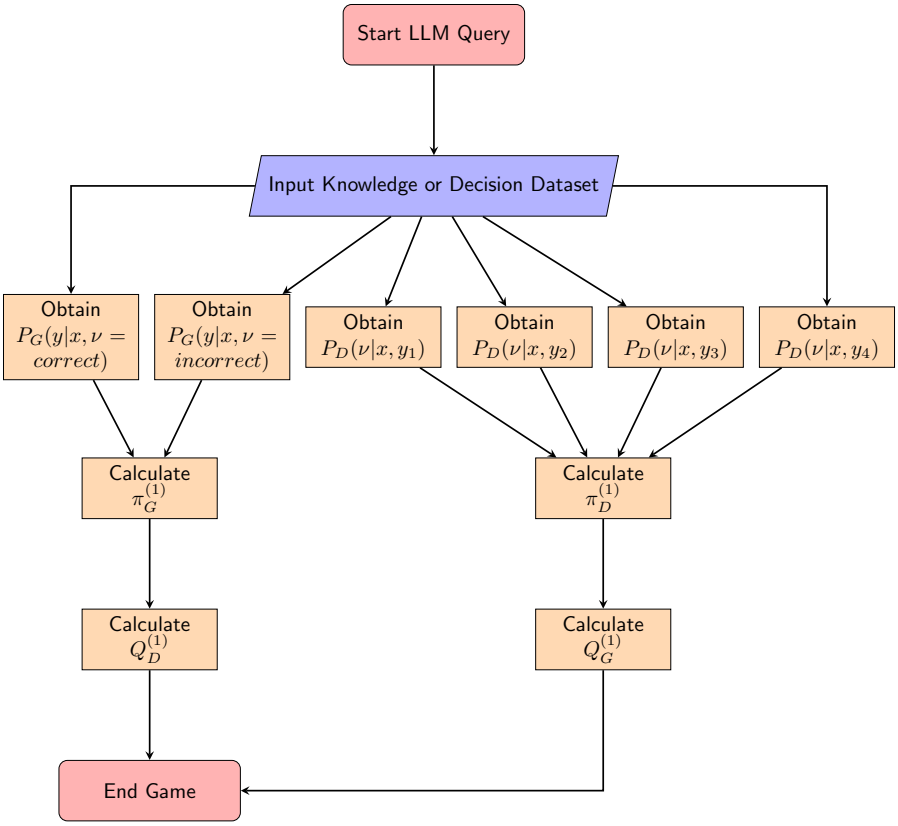
$\eta =$ learning rate

Running the Consensus Game

Example question taken from knowledge-based MMLU dataset [7]:

Why is Saturn almost as big as Jupiter despite its smaller mass?

1. Jupiter's greater mass compresses it more, thus increasing its density.
2. Saturn has a larger proportion of hydrogen and helium than Jupiter and is therefore less dense.
3. Saturn is further from the Sun thus cooler and therefore less compact.
4. Saturn's rings make the planet look bigger.



Graphic made by student using TikZ [20], 2026

Figure 1: .

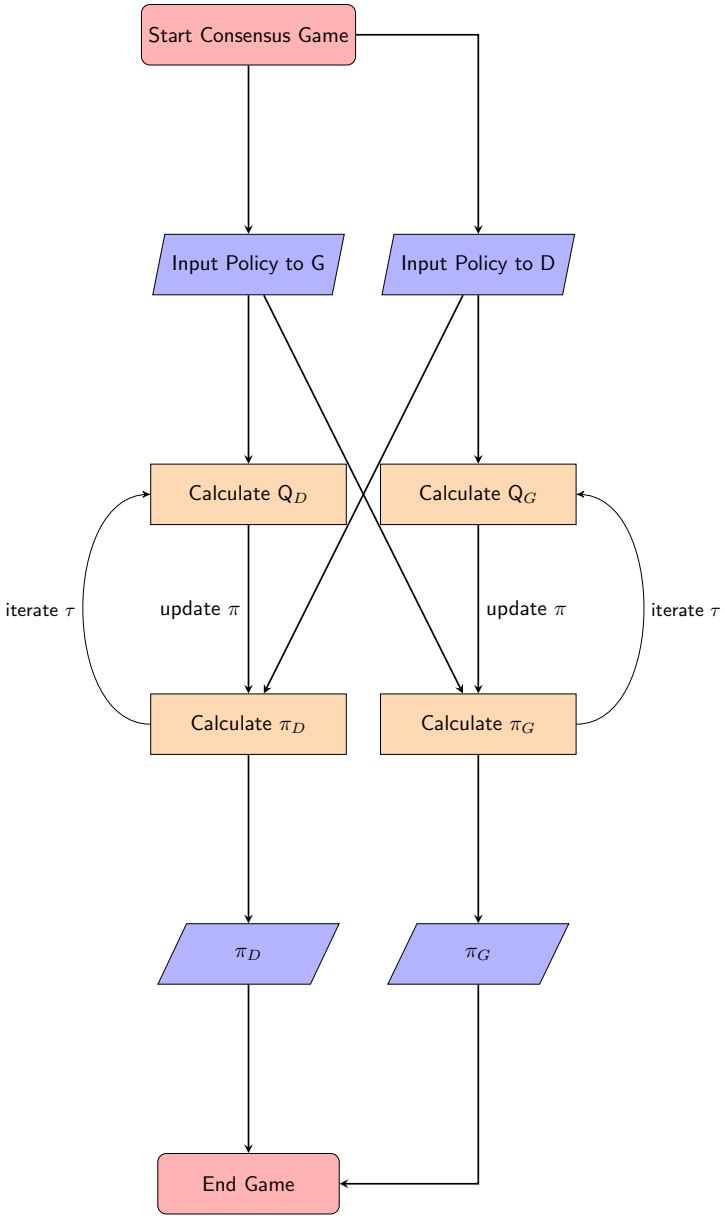


Figure 2: .

Hardware and Computational Summary

Verification and Validation

Several steps were taken to ensure that code ran correctly and efficiently. First, policy updates were calculated by hand for several test probability distributions and checked against our code's computation. In addition, code was run on multiple computers, .

Unit tests are also performed on each function used in performing the Consensus Game calculations. These are performed using the library pixi and check whether the functions return the expected answer.

Results

We now analyze results from the Consensus Game. We find that repeated updating of the policies makes the Generator and Discriminator very 'confident' of their answers. The initial probability of choosing a given answer is relatively low, but after running the policy updates 5000 times, the probability of choosing one of the four answers increases logarithmically, often to near 100% probability after the first 2000 runs (see Figure). As seen in Figure, the answers of each LLM are thus polarized, as one answer choice becomes most popular and the others are almost never chosen. Consequently, through repeated 'interaction', both the Generator and Discriminator become extremely confident of their answers. The goal is that their choice correctly answers a given question.

Note that a certain complexity is introduced by asking for the LLM to select an incorrect answer as opposed to a correct option. Often an initial answer increases in probability before being replaced by a different choice (see Figure). This indicates the greater difficulty that comes with the greater sophistication of such a question.

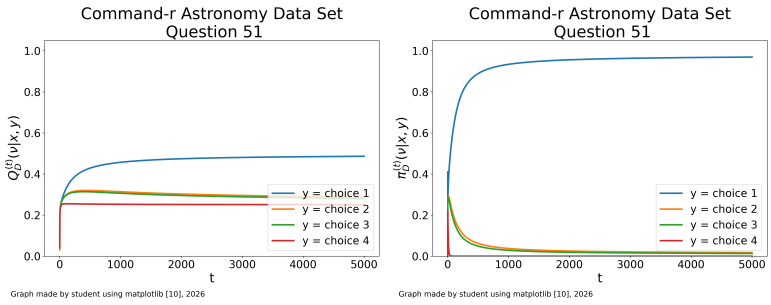


Figure 3: Evolution of Q-value and policy π_D over time step t.

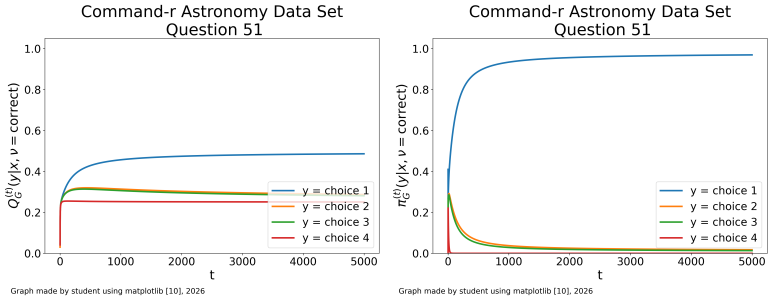


Figure 4: Evolution of Q-value and policy π_G over time step t for correct answer choices.

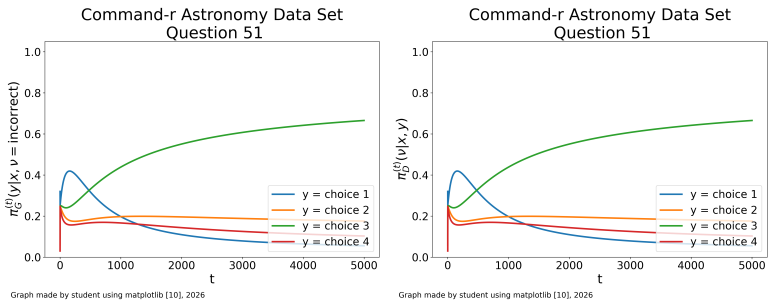


Figure 5: Evolution of Q-value and policy π_G over time step t for incorrect answer choices.

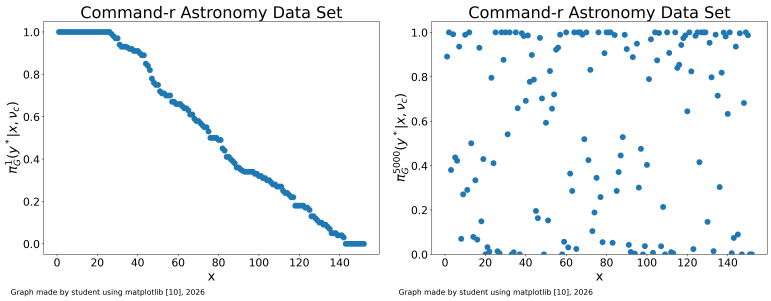


Figure 6: Accuracy of Generator in answering ordered set of astronomy questions before and after the Consensus Game is played.

Optimizing Hyperparameters

To determine how much the policy should change after each update, we iterate through values of λ_G, λ_D, η_G, and η_D to determine which would produce the highest accuracy. Overall, we find that accuracy for decision-based questions decreases the greater that λ and thus the Kullback-Leibler Divergence become, or the more that the policy is changed from its previous value. Thus, changing the probability distribution in small steps is most effective.

In addition, we find that the learning rate η has little effect on the final accuracy of each LLM on any given decision-based dataset; however, increase η shortens the number of times the Consensus Game must be played before the updated policy plateaus and becomes constant. As such, increasing the learning rate η increases the efficiency of the algorithm, as the Consensus Game needs to be performed fewer times before coming to a final answer (see Figure).

Conclusions

Overall, we find that the Consensus Game improves accuracy of knowledge-based questions only marginally, as seen in Figure . Decision-based questions, on the other hand, resulted in large increases in accuracy, often to almost 100%. Thus, our hypothesis that the Consensus Game would be more useful in improving the decision-making capabilities of LLMs was proved (see Figure). We also find that the Discriminator tends to improve more drastically than the Generator.

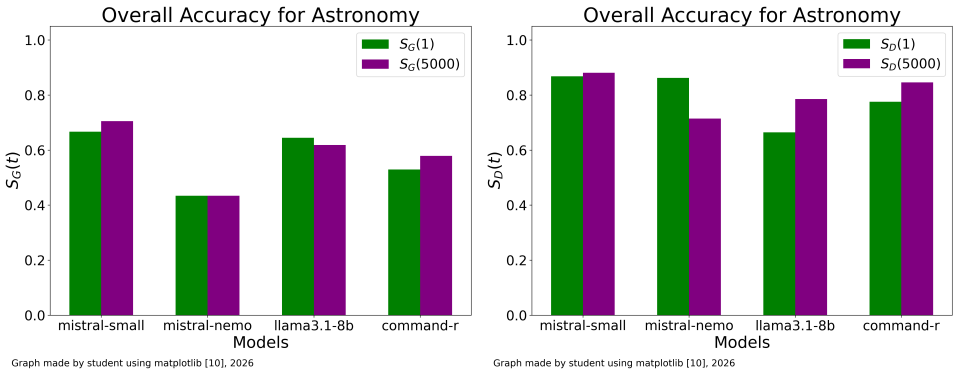


Figure 7: Overall accuracy of each LLM in answering astronomy questions from the MMLU database.

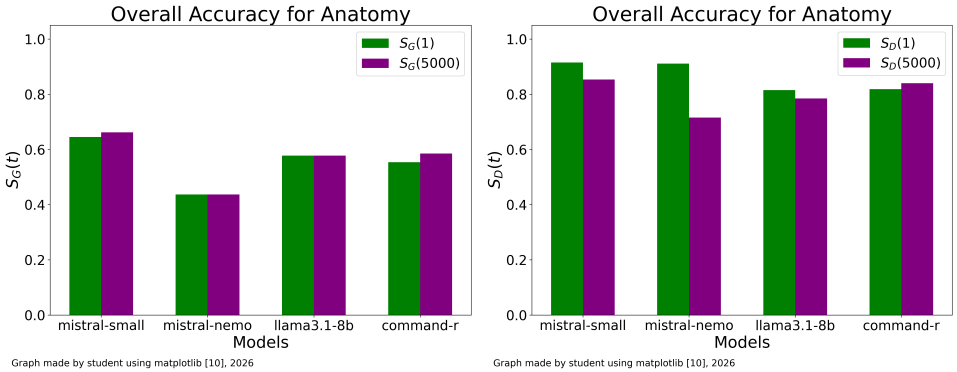


Figure 8: Overall accuracy of each LLM in answering anatomy questions from the MMLU database.

Analysis of Results

Analysis of Biases and Limitations

Next Steps

Summary

Battle [8]

Two teenagers with separate tastes must agree on an activity. Their choices are basketball and shopping–the boy prefers basketball, and the girl prefers shopping, but they both prefer to go together, wherever they go..

Boy \Girl	**Basketball**	**Shopping**
Basketball	3,2	1,1
Shopping	0,0	2,3

Table made by student using LaTex. [18]

Prisoner's Dilemma [8]

Two suspects are arrested with a potential five year sentence. If neither confess to the crime, they will be jailed for one year. If one confesses and the other does not, the one who admits will go free while the other serves eight years of jail.

1 \2	**Defect**	**Cooperate**
Defect	-5,-5	0,-8
Cooperate	-8,0	-1,-1

Table made by student using LaTex. [18]

Chicken [22]

Two drivers speed towards each other on the same road. The one who swerves first is the game's loser, or 'chicken.' If neither swerve, they crash.

1 \2	**Swerve**	**Straight**
Swerve	0,0	-1, 1
Straight	1, -1	-1000, -1000

Table made by student using LaTex. [18]

Stag Hunt [22]

Two hunters may hunt stag or hare. Hare produces less meat, but stag cannot be caught without

the help of the other hunter.

1 \2	**Stag**	**Hare**
Stag	10,10	0, 2
Hare	2, 0	2, 2

Table made by student using LaTex. [18]

Assurance Game [22]

Two companies may produce a cheap or expensive product. Their profits will be lower if they do not choose the same product.

1 \2	**A**	**B**
A	50, 50	15, 15
B	15, 15	25, 25

Table made by student using LaTex. [18]

Traveler's Dilemma [23]

Two airline passengers have identical damaged briefcases. They must choose a reparation value. If one chooses a lower value than the other, the lower value will be accepted and the one who chose the lower value will gain some portion of the other passenger's reparations.

1 \2	**Lower Amount**	**Higher Amount**
Lower Amount	20, 20	30, 10
Higher Amount	10, 30	80, 80

Table made by student using LaTex. [18]

Appendix

Mathematical Glossary

Linguistic Glossary

Acknowledgments

This project could not have been initiated without the helpful advice from my parents; their encouragement and the deeper understanding of science, computer science, and mathematics they have given me have been irreplaceable. I would also like to thank the many judges and reviewers from Science Fair and the Supercomputing Challenge, who generously provided me with feedback along the way, and the authors of the numerous papers that contributed to my knowledge of the project's background. Together they have given me the motivation to see this project to completion.

References

[1] Aidan Gomez. Introducing command r7b: Fast and efficient generative ai. https://cohere.com/blog/command-r7b, 2024.

[2] Mistral AI. Mistral nemo. https://mistral.ai/news/mistral-nemo, 2024.

[3] Mistral AI. Mistral small 3. https://mistral.ai/news/mistral-small-3, 2025.