Guiding genAI to play Diamonds Game

veda sree konka

25-03-2024

Introduction

In this report, I'll share how I taught genAI,to play Diamonds Game. Diamonds isn't just any card game—it's like a clever puzzle where bidding and strategy are key. I'll walk you through how I helped genAI learn the game and how to play it, and how it played the game.

Rules of the Game

Diamonds is a two or three player game (In case of two player game, one of the suits is kept aside, out of play.) Diamonds are the drivers of the game, and all the diamonds are shuffled and kept face down. Each player gets all the cards of one suit in their hand. Now, the topmost diamond is drawn face up. This is the value being bidded over. Both players decide to bid some amount and place a card with a rank equal to what they want to bid, face down. Both the bids are turned face up at the same time, and the higher bid wins the diamond card being auctioned. The winner is the player with a greater total face value of diamond cards won this way.

Prompts Given

To teach genAI, I explained the game in simple terms. We discussed what makes a diamond worth bidding on and which ones to pass on. I showed genAI why it's important to pick the right cards to bid on and make smart choices.

Teaching genAI the Game

So, to get genAI up to speed with this Diamonds game where we both know each other's hands, I focused on helping genAI understand the whole picture. We looked at the cards genAI has and the diamonds still on the table. I taught genAI to see which diamonds are worth bidding on and which ones to let go

Strategy Discussed

Our strategy is to make to genai use what it knows about both players' cards to make good bids. Instead of just guessing, it looks at the value of the diamonds and its own cards. Then, it sets some rules for itself about how much to bid.

Code for the Strategy and Practical Results

Below is the Python code that implements the bidding strategy for genAI:

```
import random

def calculate_diamond_score(diamond_value, hand):
```

```
    higher_cards = [card for card in hand if card > diamond_value]
    return len(higher_cards) + diamond_value

def choose_bid_card(remaining_diamonds, hand, safe_bid_threshold):
    bid_cards = []
    for diamond_value in remaining_diamonds:
        diamond_score = calculate_diamond_score(diamond_value, hand)
        if diamond_score >= safe_bid_threshold:
            bid_card = max([card for card in hand if card > diamond_value])
        else:
            bluff_probability = (diamond_value / 13) * 0.5
            if random.random() < bluff_probability:
                bid_card = min(hand)
            else:
                bid_card = max(hand)
        bid_cards.append(bid_card)
    return bid_cards

# Example usage
genAI_hand = [10, 8, 6, 5, 3]
remaining_diamonds = [9, 7, 4, 2]
safe_bid_threshold = calculate_safe_bid_threshold(remaining_diamonds, genAI_hand)
genAI_bids = choose_bid_card(remaining_diamonds, genAI_hand, safe_bid_threshold)
print("genAI's bids:", genAI_bids)
```

Conclusion

Teaching genAI to play Diamonds was a bit challenging as it got confused many times. I had to carefully analyze every round and provide feedback to help genAI modify its strategy after every conversation. GenAI made some mistakes repetitively and struggled with bidding and counting the value of diamonds. Despite the challenges, with patience and guidance, genAI gradually improved its understanding of the game.