PokerMind

CT Systems

Christopher Dowdy - Project Manager, Backend/ML Developer

Tristen Klute - Scrum Master, Frontend/Backend Developer

Dr. Emma MacKie (emackie@ufl.edu)

February 02, 2025

Abstract

The purpose of this project is to develop PokerMind – an LLM-powered poker agent that can learn, adapt, and play poker at a professional level. The agent will undergo a fine-tuning pipeline combining supervised fine-tuning (SFT) on poker hand datasets with reinforcement learning (RL) in a custom game environment. This project aims to develop an LLM model that performs better at poker than state-of-the-art general models; exploring how LLMs combined with RL can extend beyond natural language processing to decision-making in incomplete-information games.

Table of Contents

Team Contract

Overall objectives: What the team is trying to achieve?

- Contribute to advancements in multi agent simulations by creating a video game for LLM powered operators to play and interact.
- Dive deep into AI development by building something new in the rapidly advancing field. Our goal is to learn as much as possible and build a product that gains active users.

Individual responsibilities: What each team member is expected to do?

- Each team member will be a full stack engineer that is responsible for completing their designated tasks for each sprint. Chris will be the scrum master. Each team member will equally contribute to all canvas assignments.

Values and Agreement Statements: Team agreements are written by everyone and they help define comfortable working parameters for everyone.

- Inclusion of all ideas
- Commitment to excellence
- Clear and open communication
- Mutual respect between team members
- Everyone is happy to help if anyone gets stuck

Software Configuration Management Protocol: refers to a set of defined procedures and standards used to track, control, and manage changes made to the software system to ensure consistency and quality of all modifications, management of merge conflicts, and version control.

- Use Github repo for project, git version control protocol:
    - Create issue for a sprint task
    - Create branch for issue
    - Make code changes to branch
    - Submit PR of code changes for review
    - Other team members review / approve code changes and merge PR to main
    - Delete Branch
- Create and actively manage a sprint board:
    - Every month have a sprint planning meeting where tasks are created and delegated between each team member equally
    - Every week team members will give an update on their task progress and any blockers

Meeting times: When the team will meet outside the meeting with the advisor?

- Every Sunday at 4:00 pm

Meeting times with advisor: Frequency and Order of Meeting Facilitator (Everyone is required to lead/facilitate at least 1 meeting with the advisor; everyone must be present for all meetings with your advisor).

- Bimonthly meetings beginning February
- Order of meeting facilitation: 1. Chris, 2. Tristen

Communication: How team members will communicate with each other; what platforms?

- Use Discord chat for daily communication and meet in-person for weekly meetings.

Conflict resolution protocol: How will the team solve problems?

- If we arrive at conflict, each party has the responsibility to maintain open communication and a commitment to professionalism. It is important to hear the perspectives from all parties and arrive at a solution that all parties agree with.

Consequences: What happens if someone violates the agreement?

- If anyone violates the agreement, other members will let the advisor know who violated the agreement and how it was violated. The issue will be escalated to the advisor for further guidance, resolution, and possible consequences.

Signatures: All team members must sign the contract.

- Tristen J. Klute
- Christopher A. Dowdy

Introduction & Motivation

Poker is an incomplete information game that requires a combination of skills including math, reasoning, planning, and strategy to be a successful player. Existing AI systems for poker have been able to achieve game theory optimal (GTO) performance; however, they are often computationally expensive, upper bounded by the size of a game tree, and not adaptable to different gameplay styles (Zhuang 2025). With the emergence of “thinking” large language models (e.g. OpenAI-O1, DeepSeek-R1) and reinforcement learning based post-training architectures, our goal is to create an LLM-powered poker agent that has the skill of a professional poker player (Daya 2025). We will fine-tune a base model using reinforcement learning, create an environment for LLMs to play poker against each other, and develop a web app where users can play poker against the agent. We want to contribute to the capabilities of LLMs outside of a typical chatbot setting and explore their success in incomplete information environments.

Literature Survey

Article 1: PokerBench Training large language models to become professional poker players.

This paper dives into the optimal poker playing ability of LLMs. Multiple LLMs are tested on their poker playing abilities, which the results show are quite lackluster. The researchers propose a large dataset used to fine-tune llama 3-8B and other models on thousands of simulated poker hands. These fine-tuned models are able to outperform the larger models. The study also shows that the scores from PokerBench translate to real superior poker skills, which provides a good benchmark for CT Systems models to compare.

Article 2: Are ChatGPT and GPT4 Good Poker Players?

This article compares the ChatGPT and GPT4 models' poker playing abilities by testing them against the game theory optimal way of playing poker. The models have a very good understanding of the game and basic strategy, even changing their tactics based on what starting position they have on the poker table, however, they do not play according to the optimal game theory. When asked to play GTO, ChatGPT plays too passively and GPT4 plays too aggressively. This shows that both models have a lack of understanding the style in which they are playing the game. As we can see from this study, models without fine-tuning have a hard time playing poker due to the lack of awareness of the game state and how it relates to their playstyles.

Article 3: DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning.

In this paper, DeepSeek researchers share their approach to enhancing LLM reasoning capabilities through reinforcement learning by presenting two models: DeepSeek-R1-Zero and

DeepSeek-R1. DeepSeek-R1-Zero was post-trained using pure RL without any supervised fine-tuning, using the GRPO (Group Relative Policy Optimization) framework and rule-based rewards to improve reasoning performance. This fine-tuning stage increased performance on the AIME 2024 benchmark from 15.6% to 71.0%. Building on these findings, they developed DeepSeek-RL which incorporated both cold-start data fine-tuning and reinforcement learning based fine-tuning. This approach demonstrated that the RL fine-tuning, combined with selective supervised fine-tuning, produced the best results; DeepSeek-R1 achieved performance comparable to the SOTA OpenAI o1 model across various reasoning tasks.

Proposed Work

Plan of Work:

1. fine-tune a base LLM model with supervised fine-tuning (SFT) on publicly available poker hand datasets
2. Create custom reinforcement learning game environment for LLM to play poker
3. Reinforcement learning based fine-tuning where model interacts with custom game environment to play poker hands and receive rewards based on performance
4. Evaluate performance and adjust fine-tuning parameters as needed
5. Develop a full-stack application for playing against the agent / interacting with it via chat

Languages/Technologies:

- Base model: LLaMa or DeepSeek
- Supervised fine-tuning: Pytorch, Hugging face Transformers
- Reinforcement Learning Game Environment: Python, PufferLib, PyPokerEngine
- Full-stack web app: Frontend: React, Backend: Python, FastApi, Database: MongoDB

Features:

- LLM powered poker agent that autonomously plays poker versus other agents or user
- Agent is able to adapt to different players' strategies and not have one certain play style
- Web app where you can interact with agent and watch them play

Target Audience:

- AI/ML researchers who are interested in LLM applications and the capabilities that reinforcement learning unlocks in LLM based agents
- Poker enthusiasts interested in the cross section with AI

Requirements Document

1. LLM Fine-Tuning

1.1: Base Model Selection

- Decide which base model to use based on whichever performs best on PokerBench (LLaMa vs Deepseek)

1.2: Supervised Fine-Tuning (SFT)

- Fine-tune the base LLM using publicly available poker hand datasets
- The dataset includes:
  - Hand histories, Strategy explanations, GTO solver outputs for decision-making

1.3: Initial Model Evaluation

- The fine-tuned model will be evaluated on hand classification tasks via PokerBench, aiming for superior performance to base models

2. Reinforcement Learning & Environment

2.1: Poker Simulation Environment

- The system will provide a custom RL training environment using:
  - PufferLib for multi-agent RL
  - PyPokerEngine for poker logic simulation
  - A framework to simulate various game modes (Heads-Up, 6-Max)

2.2: RL Algorithm Implementation

- The system will optimize the poker agent using Group Relative Policy Optimization (Daya 2025)
- The reinforcement learning setup will allow:
  - Policy iteration via self-play and adversarial training
  - Reward modeling for EV maximization, bluffing, and adaptation.

2.3: Reward System

- The RL environment will compute rewards based on:
  - Immediate hand success (win/loss)
  - Expected Value (EV) calculations
  - Bluff success rates and opponent adaptation
  - Long-term profitability over multiple hands

2.4: Model Self-Play & Training Cycles

- The agent will train through self-play to develop adaptive strategies
- Opponent modeling will be implemented to adapt the agent's aggressiveness, passivity, and bluff frequency

2.5: RL Performance Benchmarking

- The system will evaluate the RL-trained model against:
  - GTO solvers
  - Fixed poker strategies (tight, loose, aggressive)
  - Human play

3. Full-Stack Application Development

3.1: Web Interface

- The system will provide a web application for interacting with the poker AI.
- The application will support:
    - Playing against the AI in real-time
    - Observing AI-vs-AI poker matches
    - Chat-based interaction for poker strategy analysis

3.2: Backend API & Model Hosting

- The system will host the AI model as an API using:
    - FastAPI for API communication
    - MongoDB for storing hand histories and user interactions
    - WebSockets for real-time decision-making

Project Plan

01/21/2025 – Contract due

02/02/2025 – Project Proposal due

Weekly Status Report 02/09 – Setup and run base model locally

Weekly Status Report 02/16 – Supervised fine-tuning experiments

Weekly Status Report 02/23– Create custom reinforcement learning game environment

PRESENTATION 1 DUE 02/23

Weekly Status Report 03/09 – Reinforcement Learning based fine-tuning experiments

Weekly Status Report 03/14 – Model evaluation and benchmarking versus other SOTA models

PRESENTATION 2 DUE 03/30

Weekly Status Report 04/06 – Frontend development of web app

Weekly Status Report 04/13 - Backend development of web app

PROJECT AND DOCUMENTATION DUE 04/20

FINAL PRESENTATION DUE 04/23

Github Link

https://github.com/6hris/pokermind

References

Guo Daya, Yang Dejian, Zhang, Haowei(2025). *DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning*. arXiv. https://arxiv.org/abs/2501.12948

Gupta, A. (2023). *Are ChatGPT and GPT-4 good poker players? – A pre-flop analysis*. arXiv. https://arxiv.org/abs/2308.12466

Zhuang R., Gupta A., Yang R., Rahane A., Li Z., & Anumanchipalli G. (2025). *PokerBench: Training large language models to become professional poker players*. arXiv. https://arxiv.org/abs/2501.08328

Advisor Signature: Emma MacKie