Reinforcement Learning for Clash Royale

CS 175: Project in Artificial Intelligence

Winter 2026

The Elixir Optimizers

Alan Guo weihaog1@uci.edu
Josh Talla tallaj@uci.edu
Lakshya Shrivastava lshrivas@uci.edu

1 Summary of the Project

Our project aims to develop a **reinforcement learning agent** capable of playing **Clash Royale**, a real-time strategy card game with over 500 million downloads. Clash Royale presents a unique AI challenge that combines elements of collectible card games (strategic deck building and card selection) with real-time strategy games (unit placement, timing, and spatial reasoning). Unlike turn-based games where AI has achieved superhuman performance, Clash Royale requires continuous decision-making under time pressure with **partial observability**—the opponent's card hand and elixir count remain hidden throughout the match.

The agent will interact with the actual game running on PC via **Google Play Games**, using a **non-embedded approach**: it perceives the game state through screen capture and computer vision (object detection for units, OCR for elixir/timer), and executes actions through automated mouse input. This mirrors how a human player would interact with the game, making our approach applicable to the real game environment rather than a simplified simulation.

Input: Screen captures from the game client, processed through a computer vision pipeline (YOLOv8 for unit detection, OCR for numerical values) to extract a structured game state representation including:

- Unit positions and types on the arena
- Tower health values (friendly and enemy)
- Available cards in hand (4 cards)
- Current elixir count
- Match timer

Output: Actions consisting of:

- **Card selection** — which of the 4 available cards to play
- **Placement coordinates** — where on the arena to deploy the unit/spell

Executed via automated input (PyAutoGUI) to the game client.

2 Project Goals

Minimum Goal

Build a complete **end-to-end pipeline** that can:

1. Capture game state from Google Play Games via screen capture
2. Extract relevant features using computer vision (YOLOv8, OCR)
3. Train a **behavior cloning** agent on collected expert gameplay data
4. Execute actions in the real game via automated input

The agent should demonstrate basic competence by making contextually reasonable card placements (e.g., placing defensive units when under attack).

Realistic Goal

Train a reinforcement learning agent using **imitation learning as a warm start**, followed by **online RL fine-tuning (PPO)**, that can consistently defeat the in-game "Trainer" Bot. We will:

- Conduct systematic experiments comparing behavior cloning, offline RL (Decision Transformer), and online RL approaches
- Analyze what strategies the agent learns through visualization of its decision-making patterns
- Achieve >60% win rate against Trainer Bot

Moonshot Goal

Develop an agent that achieves **competitive performance** against the Trainer Bot and high trophy real human players. Additionally:

- Demonstrate **transfer learning** by training on one deck archetype and successfully adapting to others
- Demonstrate real-time adaptation by adjusting strategy mid-match based on the opponent's deck archetype and playstyle

3 AI/ML Algorithms

We plan to use **imitation learning** (behavior cloning) to bootstrap a policy from expert demonstrations, followed by **model-free reinforcement learning** with neural function approximators (specifically **Proximal Policy Optimization / PPO**) for online fine-tuning, with the computer vision pipeline utilizing convolutional neural networks (**YOLOv8** for object detection, **ResNet**-based classifiers for card recognition).

Component	Algorithm	Properties
State Extraction	YOLOv8 + OCR	Real-time object detection
Policy (Initial)	Behavior Cloning	Supervised learning on expert data
Policy (Fine-tuning)	PPO	Model-free, on-policy RL
Alternative	Decision Transformer	Offline RL, sequence modeling

4 Evaluation Plan

4.1 Quantitative Evaluation

Our primary metric will be **win rate** against Trainer Bot, measured across multiple matches to account for game variance. We will establish the following baselines:

Baseline	Description	Expected Win Rate
Random Policy	Uniform card/position selection	∼0%
Greedy Heuristic	Play cards ASAP at fixed positions	∼20–30%
Behavior Cloning	Imitation learning only (no RL)	∼40–50%
Target (PPO)	**IL + RL fine-tuning**	**60–80%**

Secondary metrics include:

- Average crowns per game (0–3 scale)
- Elixir efficiency (damage dealt per elixir spent)
- CV pipeline: detection accuracy (mAP) and inference latency

4.2 Qualitative Evaluation

We will verify the system works through several approaches:

1. **CV Pipeline Visualization:** Overlay detections on game frames to confirm accurate state extraction; verify unit identification and position accuracy.
2. **Decision Annotation:** Record gameplay videos with annotations showing which card was selected and the reasoning (based on attention weights or feature importance).
3. **Strategy Analysis:** Categorize agent behavior into recognizable patterns:
   - Does it learn to counter-push after defending?
   - Does it manage elixir effectively (not overcommitting)?
   - Does it respond appropriately to opponent actions?
4. **Failure Analysis:** Examine lost games to identify root causes—CV errors, poor timing, or strategic mistakes.

Success criteria: A successful agent should exhibit hallmarks of competent play: waiting for sufficient elixir before committing, placing units in strategically sound positions, and responding appropriately to opponent actions.

5 AI Tool Usage

We used AI tools (Claude) in the following aspects of this proposal:

1. **Literature Review:** AI assisted in searching for and summarizing relevant academic papers (e.g., the KataCR paper on arXiv, the Clash Royale Challenge paper from FedCSIS 2019) and open-source projects (KataCR, CRBot-public).

2. **Technical Research:** AI helped identify available datasets (Clash Royale Replay Dataset), compare different technical approaches (online vs. offline RL, simulator vs. real game), and understand the architectures used in prior work.

All AI-generated content was reviewed, verified against primary sources, and edited by team members. The core project ideas, specific technical decisions and evaluation criteria reflect our own analysis and judgment. We will continue to document AI tool usage throughout the project in our progress reports.