AI GameBot for Street Fighter II Turbo

Ffinal-year project for Spring 2025: an AI agent that learns to play **Street Fighter II Turbo** on the SNES platform by leveraging the BizHawk emulator and machine learning techniques.

Table of Contents

Introduction

This project demonstrates how deep learning can be integrated with classic console games. We build an AI agent that:

- Interacts with **Street Fighter II Turbo** via the **BizHawk** emulator.
- Collects frame-by-frame game state data (positions, health, moves, timer).
- Trains a neural network to predict multi-button commands.
- Deploys the trained model for real-time gameplay.

Features

- **Automated Data Collection**: Logs player and opponent states + button presses.
- **Multi-label Classification**: Predicts 12-button combinations per frame.
- **Real-time Inference**: Deploys model with <16ms latency at 60 FPS.
- **Plug-and-play Emulator Integration**: Uses Lua scripts and Python sockets.

System Architecture

1. **Emulator Interface**: BizHawk with Lua scripts sends game state via sockets.
2. **Data Collector Bot**: Receives state and logs CSV after each round.
3. **Neural Network Model**: Keras-based multi-label classifier with sigmoid outputs.
4. **Inference Bot**: Loads saved model and scaler, predicts commands live.

Getting Started

Prerequisites

- Python 3.10+
- BizHawk Emulator (with Lua scripting enabled)
- Street Fighter II Turbo (U).smc ROM

Installation

Clone this repository:

```
 git clone https://github.com/yourusername/sf2-gamebot.git
cd sf2-gamebot
```

1.

Create a virtual environment and install dependencies:

```
 python -m venv venv
source venv/bin/activate  # or venv\\Scripts\\activate on Windows
pip install -r requirements.txt
```

Usage

Data Collection

```
# Run the data collector for Player 1
python controller.py collect --player 1
```

- Plays basic movements and logs to `data/game_data.csv`.
- Terminates after two full rounds.

Model Training

```
python model_trainer.py --data data/game_data.csv --output model/sf2_model.h5
```

- Preprocesses data, trains the neural network, and saves model + scaler.

Bot Deployment

Run the inference bot for Player 1
python controller.py play --player 1

- Select Gyroscope Bot in BizHawk, see `CONNECTED SUCCESSFULLY` in console.
- The AI will play live matches using the trained model.

Project Structure

```
├── data/                # Logged CSV files
├── model/               # Saved model and scaler
├── scripts/             # Lua scripts for BizHawk
├── src/                 # Python source code
│   ├── bot.py           # fight() logic and model inference
│   ├── controller.py    # connects to emulator
│   ├── data_collector.py  # gameplay logger
│   └── model_trainer.py   # preprocessing & training pipeline
├── requirements.txt     # Python dependencies
└── README.md            # This file
```

Dataset Format

Each CSV row = one emulated frame, including:

- **Round Info**: timer, has_round_started, is_round_over
- **Player / Opponent**: health, x/y coords, jumping, crouching, move ID
- **Pressed Buttons**: up, down, left, right, A, B, X, Y, L, R

Typical session: 5,000–10,000 frames.

Results

- **Button Prediction Accuracy**: >60% on test set
- **Stable Real-time Performance**: <16ms inference at 10 FPS