Lucifer NBA Prediction System

A sophisticated NBA game prediction system that combines neural networks, reinforcement learning, and real-time data processing to provide accurate game predictions and insights.

🏀 Overview

This system leverages state-of-the-art machine learning techniques to predict NBA game outcomes with high accuracy. It integrates multiple data sources, including live NBA statistics and betting odds, to generate confidence-calibrated predictions in real time. The architecture supports continuous learning through reinforcement learning, optimizing betting strategies based on prior game results.

🏗️ System Architecture

Core Components

1. **Dashboard (/dashboard)** A Flask-based web application providing a user-friendly interface that displays real-time NBA game predictions, live game statuses, and odds comparisons. It utilizes asynchronous data fetching and WebSocket updates to ensure data freshness.
2. **Model System (/model)** Contains the core Neural Network (NN) prediction model saved as nn_calibrated.keras. The NN is custom-designed with feature scaling, imputation, and temperature scaling techniques to produce calibrated probabilistic outputs. This module also manages model validation and supports retraining workflows.
3. **Reinforcement Learning (/RL)** Implements a Deep Q-Network (DQN) agent responsible for learning optimal betting policies. The RL agent uses game features and NN confidence scores as its state space, exploring an action space of different betting strategies. The reward function reflects the success of bets, enabling the agent to improve its decision-making over time.
4. **Data Processing and Integration** A pipeline that performs feature engineering including rolling statistics, team normalization, and API data ingestion (NBA stats, odds). It caches data efficiently to reduce API calls and preprocesses raw inputs to ensure consistency and completeness.
5. **Inference Scripts (/inference)** Contains standalone prediction scripts (predict_nn.py and predict_with_rl.py) for running predictions outside the dashboard interface. These scripts allow for batch predictions and integration with external systems.
6. **Data Scraping and Caching** The system includes several data scraping utilities in dashboard/routes/ (e.g., injury_scraper.py, scrape_advanced_stats.py) that populate data caches. Cached data is stored in the cache_stats/ directory as JSON files, reducing API calls and improving response times.
7. **Training Notebook** The basketball_model_colab.py notebook contains comprehensive model training code, including data preprocessing, model architecture definition, and training workflows. This notebook is designed to run in Google Colab for GPU acceleration.
8. **Utilities** Additional utility scripts and modules:

- dashboard/utils/schedules_utils.py: Handles schedule management with Redis integration
- test_all_apis.py: Comprehensive API testing suite
- Various helper scripts for data processing and system maintenance

🚀 Setup Instructions

Prerequisites

- Python 3.8+ — Ensures compatibility with TensorFlow and modern libraries.
- pip — Python package manager for installing dependencies.
- Virtual environment (recommended) — To isolate project dependencies from the system.

Installation Steps

1. Clone the repository to your local machine:

```
git clone [repository-url]
cd Lucifer
```

2. Create and activate a Python virtual environment:

```
python -m venv venv
source venv/bin/activate  # Windows: venv\Scripts\activate
```

3. Install Python dependencies:

```
pip install -r dashboard/requirements.txt
```

4. Install additional packages (if needed):

```
pip install tensorflow pandas numpy requests beautifulsoup4 flask apscheduler
```

Configuration

1. **API Keys:**
   - Place your NBA and Odds API keys in the designated config files or environment variables for secure access.
2. **Model files:**
   - Ensure the neural network and RL model files are stored in the correct directories (/model and /RL) and update config paths accordingly.

📊 API Documentation

Main API Endpoints

1. **GET /api/predictions**
   - Returns current NBA game predictions with confidence scores and matchup details.

2. **GET /get_upcoming_games**
    - Retrieves scheduled upcoming NBA games.
3. **GET /get_live_games**
    - Provides live data and statuses for ongoing games.
4. **GET /get_odds**
    - Returns the latest betting odds from integrated APIs.
5. **GET /health**
    - Health check endpoint to verify system status.

Response Format Example


```
{
 "predictions": [
  {
   "home_team": "Los Angeles Lakers",
   "away_team": "Golden State Warriors",
   "prediction": 1,
   "confidence": 83.7,
   "matchup": "Golden State Warriors @ Los Angeles Lakers"
  }
 ]
}
```


🧠 Model Details

Neural Network (NN) Model

- **Architecture:** Fully connected deep neural network optimized for binary classification of game outcomes.
- **Input Features:** Aggregated team stats, player performance metrics, recent game history, and opponent strength indicators.
- **Temperature Scaling:** Applied post-training calibration to convert raw logits into reliable probability estimates.
- **Outputs:** Probability of home team winning and associated confidence level.

Reinforcement Learning (RL) Agent

- **Algorithm:** Deep Q-Network (DQN) utilizing experience replay and epsilon-greedy policy for exploration.

- **State Space:** Feature vector composed of NN predictions, confidence, game contextual data, and historical betting performance.
- **Action Space:** Discrete set of betting strategies (e.g., no bet, standard bet, high-risk bet).
- **Reward Function:** Designed to incentivize accurate predictions and profitable betting decisions, penalizing losses and encouraging risk management.

💻 Usage Guide

Running the Dashboard

To start the web interface:

cd dashboard

python test_app1.py

Open your browser and navigate to:

http://localhost:5000

Features include:

- Real-time display of game predictions and confidence.
- Live updates on ongoing games.
- Odds comparison for informed betting insights.

Training Models

 Neural Network Training Train the NN model using the provided Colab notebook:

python basketball_model_colab.py

Reinforcement Learning Train the RL agent by running:

python train_rl.py

Model Validation Validate RL model using:

python validate_rl.py

Policy Updates Adjust RL policies dynamically with:

python policy_updater.py

Data Management

 Running Scrapers To update cached data:

cd dashboard/routes

python injury_scraper.py

python scrape_advanced_stats.py

2. **Cache Management**

- Cached data is stored in cache_stats/
- Clear cache if needed: rm -rf cache_stats/*
- Monitor cache size and update frequency in configuration

📦 Dependencies

Core Python libraries used:

- Flask — Web framework for dashboard.
- TensorFlow — Deep learning model development.
- Pandas, NumPy — Data manipulation and numerical computations.
- Requests, BeautifulSoup4 — API access and web scraping.
- APScheduler — Scheduling periodic tasks like data refresh.

👷 Development Guidelines

- Follow PEP 8 Python style standards for clean, readable code.
- Use type hints for function parameters and return types to enhance code clarity and facilitate static analysis.
- Maintain modular, reusable components separated by functionality.
- Write comprehensive unit and integration tests to ensure reliability, especially for core data processing and model inference code.
- Document functions, classes, and modules thoroughly for maintainability.

📝 License

This Software is provided "as is," without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose, or non-infringement. In no event shall the author be liable for any claim, damages, or other liability, whether in an action of contract, tort, or otherwise, arising from, out of, or in connection with the Software or the use or other dealings in the Software.

If you wish to obtain permission to use, license, or collaborate on this Software, please contact:

Akash Tamang
Email: suzuki887tamang@gmail.com
GitHub: https://github.com/waibaboy/lucifer_training

Unauthorized use is strictly forbidden.