

# Progressive Learning Chess Engine

## A Hybrid Bayesian-LSTM Architecture with Curriculum Learning

Shyamal Suhana Chandra

Research & Development

November 17, 2025

# Overview

- 1 Introduction
- 2 Architecture
- 3 Learning Mechanisms
- 4 Implementation
- 5 Results
- 6 Multi-Agent Extension
- 7 Future Work
- 8 Conclusion

# The Challenge

- Traditional chess engines rely on brute force computation
- Deep learning approaches require massive resources
- No progressive learning from basic to advanced concepts
- Limited application of psychological learning principles

## Our Solution

A chess engine that learns progressively through curriculum learning, combining Bayesian networks, LSTM networks, spaced repetition, and Pavlovian conditioning.

- **Hybrid Architecture**

- Bayesian networks
- LSTM networks

- **Curriculum Learning**

- 10 difficulty levels
- Progressive advancement

- **Spaced Repetition**

- Long-term memory
- Adaptive intervals

- **Pavlovian Conditioning**

- Reward-based learning
- Association strength

# System Architecture



## Bayesian Layer

Models conditional probabilities for piece positions:

$$P(\text{move}|s) = \frac{\exp(\sum_i w_i \cdot f_i(s, \text{move}))}{\sum_{\text{move}'} \exp(\sum_i w_i \cdot f_i(s, \text{move}'))}$$

## LSTM Layer

Processes sequences of board states:

$$h_t = \text{LSTM}(\text{Bayesian}(x_t), h_{t-1})$$

## Difficulty Levels:

- ① Preschool
- ② Kindergarten
- ③ Elementary
- ④ Middle School
- ⑤ High School
- ⑥ Undergraduate
- ⑦ Graduate
- ⑧ Master
- ⑨ Grandmaster
- ⑩ Infinite

## Advancement Criteria

Advance if  $\frac{\text{correct}}{\text{total}} \geq 0.85$

## Benefits

- Prevents hallucinations
- Builds solid foundation
- Gradual complexity increase

# Spaced Repetition

## Adaptive Interval Calculation

$$I_{\text{next}} = I_{\text{current}} \times (2.5 + 0.5 \times (s - 1))$$

where  $s$  is the correct streak.

## Long-Term Memory Transition

Examples transition to LTM when correct streak  $\geq 5$ .

Streak	Interval Multiplier
1	2.5
2	3.0
3	3.5
4	4.0
$\geq 5$	LTM

# Pavlovian Conditioning

## Rescorla-Wagner Model

$$\Delta V = \alpha \times \beta \times (\lambda - V)$$

- $V$ : Association strength
- $\alpha$ : CS learning rate
- $\beta$ : US learning rate
- $\lambda$ : Maximum association (1.0 reward, -1.0 punishment)

## Application

- **CS**: Chess position
- **US**: Move evaluation (win/loss/draw)
- **Result**: Expected reward prediction

## Three Formats:

- ① FEN strings
- ②  $8 \times 8 \times 12$  matrices
- ③ Move sequences

### Matrix Encoding

- $8 \times 8$  board
- 12 channels:
  - 6 piece types
  - 2 colors
- One-hot encoding

# Training Pipeline

```
Initialize neural network, curriculum, Pavlovian learner
WHILE not converged:
    level = current curriculum level
    examples = get examples for level
    FOR each example:
        output = forward pass(network, input)
        loss = backward pass(network, target)
        Update weights
        IF correct:
            Pair CS (position) with US (reward)
            Update spaced repetition (success)
        ELSE:
            Pair CS (position) with US (punishment)
            Update spaced repetition (failure)
    IF accuracy >= 0.85:
        Advance to next level
```

# Test Suite Results

Test Suite	Passed	Total
Unit Tests	17	17
Regression Tests	7	7
A-B Tests	6	6
Blackbox Tests	7	7
UX Tests	8	8
<b>Total</b>	<b>45</b>	<b>45</b>

## All Tests Passing

- Neural network operations verified
- Curriculum progression validated
- Spaced repetition intervals correct
- Pavlovian associations learned
- Position evaluation stable

# Key Achievements

- **Progressive Learning:** Successfully learns from basic to advanced
- **Hallucination Prevention:** Curriculum ensures solid foundation
- **Long-Term Retention:** Spaced repetition maintains learned patterns
- **Reward Learning:** Pavlovian conditioning enables natural learning
- **Extensibility:** Multi-agent framework for various sports

# Extensibility to Sports

## Supported Games:

- Chess
- Football (Soccer)
- Basketball
- Baseball
- Hockey
- Tennis

### Generic Framework

- **GameState**: Generic state representation
- **Agent**: Individual learning agents
- **GameAction**: Action space definition
- **MultiAgentGame**: Game orchestration

- ① **Full MCTS:** Implement Monte Carlo Tree Search
- ② **Self-Play:** Training through self-play like AlphaZero
- ③ **Performance Comparison:** Benchmark against Stockfish and Leela
- ④ **Distributed Training:** Scale to larger networks
- ⑤ **Chess Variants:** Support Fischer Random, etc.
- ⑥ **Sports Applications:** Real multi-agent sports scenarios

## What We Built

A novel chess engine combining multiple learning paradigms:

- Hybrid Bayesian-LSTM architecture
- Curriculum learning with 10 difficulty levels
- Spaced repetition for long-term memory
- Pavlovian conditioning for reward learning
- Extensible multi-agent framework

## Results

- All 45 tests passing
- Successful progressive learning
- Ready for extension to sports

Thank You

# Questions? **Shyamal Suhana Chandra**

<https://github.com/Sapana-Micro-Software/progressive-learning-chess-engine>  
Copyright (C) 2025, Shyamal Suhana Chandra