Dartmouth ENGS 106: Principles of Machine Learning

Moin Arz Mattar

Jan 18, 2025

Email: moin.a.mattar.28@dartmouth.edu
LinkedIn: `https://www.linkedin.com/in/moin-mattar-8987841b4/`

Prof. Peter Chin

1 Introduction

Say you are playing with your friend rock-paper-scissors one million times. And whoever wins, gets a billion dollars. What's your chances of winning? Whatever you think it is, it could be much higher!

How many times have you played rock-paper-scissors, and you just try to reverse-engineer what your opponent will put? First, you just test him, a luck game, or just put whatever you are used to doing... you have this secret strategy that might get you the first win.

Rock-Paper-Scissors (RPS) is a simple game, but human players often exhibit patterns. This project leverages Hidden Markov Models (HMMs) with transition and emission matrices to predict an opponent's moves and maximize the chances of winning.

2 Mathematical Formulation

We model the game as a Hidden Markov Model (HMM) with the following components:

- **States** $S = \{0, 1, 2\}$ representing the opponent's possible moves: Rock (0), Paper (1), and Scissors (2).
- **Observations** $O = \{0, 1, 2\}$ representing outcomes: Win (0), Lose (1), and Tie (2).
- **Transition Matrix** A, where A_{ij} is the probability of moving from state i to j:

$$A = \begin{bmatrix} P(0|0) & P(1|0) & P(2|0) \\ P(0|1) & P(1|1) & P(2|1) \\ P(0|2) & P(1|2) & P(2|2) \end{bmatrix} \tag{1}$$

- **Emission Matrix** B, where B_{ik} is the probability of observing outcome k given state i:

$$B = \begin{bmatrix} P(0|0) & P(1|0) & P(2|0) \\ P(0|1) & P(1|1) & P(2|1) \\ P(0|2) & P(1|2) & P(2|2) \end{bmatrix} \tag{2}$$

The matrices are initialized with counts, and later converted into probability matrices. The transition matrix tracks the likelihood of moving from one state (move) to another, and the observation matrix tracks the likelihood of observing a particular outcome (win, lose, tie) given a certain move.

3 Dynamic Matrix Updates

The transition and observation matrices are dynamically updated based on the training data from previous rounds of the game. Each time a new round is played, the matrices are updated with the new transition (previous move $\rightarrow$ current move) and observation (outcome $\rightarrow$ current move) counts. The matrices are then normalized to provide the current probability distribution for transitions and observations.

4 Prediction Using the Viterbi Algorithm

To predict the next move, we employ the Viterbi algorithm, which finds the most probable sequence of hidden states (opponent moves) given the observations (outcomes).

4.1 Initialization

Define initial state probabilities π, which assume a uniform distribution over all possible moves. The Viterbi matrix is initialized as follows:

$$V[0, i] = \frac{1}{3} \cdot B[i, o_0] \tag{3}$$

where o_0 is the first observed outcome.

4.2 Recursion

For each time step t and each state s_j:

$$V[t, j] = \max_{s_i} (V[t-1, i] \cdot A[i, j]) \cdot B[j, o_t] \tag{4}$$

where $V[t-1, i]$ represents the highest probability path leading to state s_i.

4.3 Backtracking

Store backpointers:

$$\text{backpointer}[t, j] = \arg\max_{s_i} (V[t-1, i] \cdot A[i, j]) \tag{5}$$

At the final time step T, select the most probable state:

$$s_T = \arg\max_{s_j} V[T, j] \tag{6}$$

Trace back the sequence using the stored backpointers to reconstruct the most likely sequence of moves.

5 Example Calculation

Suppose the transition matrix is:

$$A = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.5 & 0.3 \\ 0.3 & 0.3 & 0.4 \end{bmatrix} \tag{7}$$

And the emission matrix is:

$$B = \begin{bmatrix} 0.5 & 0.3 & 0.2 \\ 0.2 & 0.5 & 0.3 \\ 0.3 & 0.3 & 0.4 \end{bmatrix} \tag{8}$$

Given past moves and outcomes, the Viterbi algorithm computes the most probable next move.

6 Game Play Implementation

The game proceeds as follows:

- The player selects a move (Rock, Paper, or Scissors).
- The AI predicts the player's next move based on the outcomes of previous rounds using the Viterbi algorithm.
- The AI plays the move that will beat the predicted player's move.
- The outcome (Win, Lose, Tie) is determined based on the AI's move and the player's move.
- The matrices are updated based on the current round's outcome and moves.

The dynamic updates of the matrices and prediction of the next move ensure that the AI's strategy evolves over time.

7 Takeaway

This lab helped me understand the application of Hidden Markov Models and the Viterbi algorithm in predicting sequences based on observed outcomes. I gained hands-on experience in building a dynamic model that adapts over time, improving its predictions and learning from previous data.

By leveraging an HMM-based model with dynamically updated transition and observation matrices, we can predict an opponent's moves in Rock-Paper-Scissors. This allows the AI to adjust its strategy over time, maximizing its chances of winning by leveraging the patterns in the opponent's behavior.