Hackman – Hangman Agent using Reinforcement Learning & Hidden Markov Model

Name	SRN
Taha Hussain	PES1UG23AM333
Suraj Singh	PES1UG23AM326
Srisht	PES1UG23AM317
Someshvar	PES1UG23AM311

Hackman – Hangman Agent using Reinforcement Learning & Hidden Markov Model

1. Introduction

The Hackman project integrates a Hidden Markov Model (HMM) with a Reinforcement Learning (RL) agent to play the game of Hangman. The HMM acts as an oracle that predicts letter probabilities based on the structure of partially known words. The RL agent learns an optimal guessing strategy by interacting with the environment using a reward-based feedback loop.

2. Dataset Preprocessing

The original corpus contained 50,000 English words. During preprocessing, words were converted to lowercase, duplicates were removed, and the dataset was grouped by word length. The processed data was stored in a folder called 'length_wise_disection' for training the HMM model.

3. Hidden Markov Model (HMM)

The HMM estimates the probability of each letter appearing in specific positions of words. Each hidden state corresponds to a letter position, and emissions correspond to the actual letters. The model outputs a probability distribution for the next most likely letter given the current word mask and the set of guessed letters.

4. Reinforcement Learning Agent

The Reinforcement Learning agent was implemented using a Q-learning approach. Each state is represented as a tuple of (masked_word, guessed_letters). The actions correspond to choosing one of the 26 alphabet letters. Rewards were designed as follows:

- +3 for a correct guess
- -1 for a wrong guess
- +20 for winning the game
- -5 for losing the game

5. Training Process

The agent was trained for 5000 episodes on the cleaned corpus dataset. Exploration was managed using an epsilon-greedy policy, where epsilon decayed gradually from 1.0 to 0.05. Initially, the agent explored more randomly, and as training progressed, it relied increasingly on learned Q-values.

Hackman – Hangman Agent using Reinforcement Learning & Hidden Markov Model

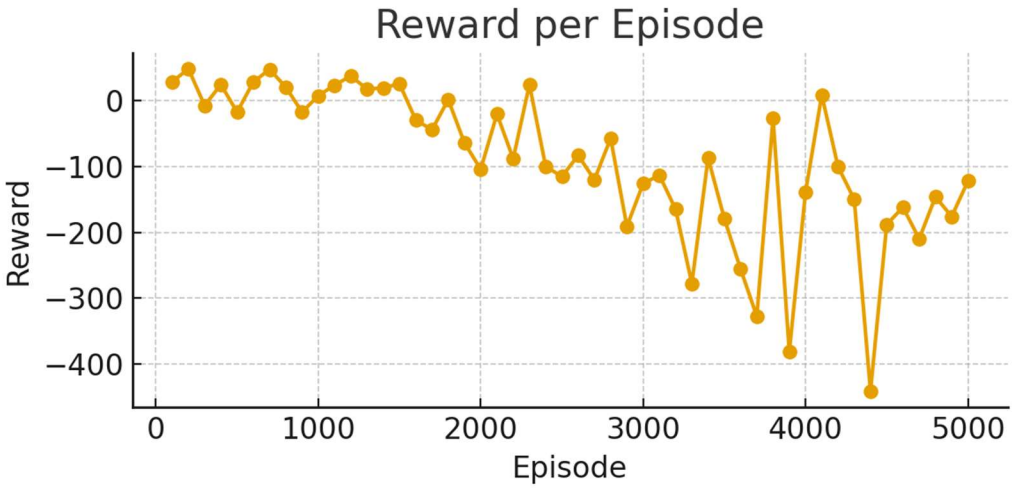

Figure 1: Reward per Episode

6. Evaluation & Results

The trained Q-learning agent was evaluated over 2000 Hangman games using a hidden test dataset. The results were as follows:

Total Games	2000
Wins	443 (22.15%)
Wrong Guesses	9461
Repeated Guesses	0
Final Score	-46862

7. Key Observations

• The most challenging part was balancing exploration and exploitation during training.
• Integrating HMM outputs with Q-learning improved decision accuracy in early guesses.
• Despite a modest win rate, the model demonstrates meaningful learning behavior across episodes.
• Fine-tuning the reward structure significantly impacted stability.

8. Strategies

The HMM was trained separately to act as an oracle estimating letter probabilities. These probabilities were incorporated into the RL agent as part of the action-selection logic. The RL state representation was designed to capture the current mask and guessed letters.

Reward shaping encouraged shorter paths to success while penalizing incorrect or repeated guesses.

9. Exploration vs Exploitation

An epsilon-greedy exploration strategy was used, starting from ε=1.0 and decaying exponentially to ε=0.05. This allowed the agent to explore widely in the early stages and gradually exploit its learned knowledge. Exploration ensured coverage of different word structures, while exploitation helped stabilize learning near the end of training.

10. Future Improvements

- Replace Q-table with a Deep Q-Network (DQN) for larger state representations.
- Integrate the HMM probabilities directly as input features for the neural agent.
- Train separate HMMs for different word lengths to improve prediction accuracy.
- Add adaptive reward shaping to encourage early correct guesses.
- Use transfer learning to fine-tune the model across similar word datasets.