MACHINE LEARNING HACKATHON
HACKMAN

SECTION : G

NAME	SRN
PIYUSH SANDEEP KHANDAKE	PES2UG23CS410
PRANAV MANOJ PUNCHDATH	PES2UG23CS428
NITIN MALLIKARJUN DHOTRE	PES2UG23CS400
MOHAMED TAHEER	PES2UG24CS814

Key Observations

One of the most challenging parts of this project was defining an effective state representation for the reinforcement learning agent. Representing each partially guessed Hangman word, while keeping the state space manageable, was difficult. Another major challenge was integrating the probabilistic outputs of the Hidden Markov Model (HMM) with the reinforcement learning process. The HMM provided probabilities for likely letters, while Q-learning focused on maximizing long-term rewards. Balancing both sources of information required careful parameter tuning.

Reward design was also tricky. Assigning proper values for correct guesses, incorrect guesses, and game outcomes was essential to avoid extreme behavior. If penalties were too strong, the agent became too cautious; if too weak, it guessed randomly. After experimentation, a reward system that gave +1 for correct guesses, -1 for incorrect guesses, -0.5 for repeated guesses, +10 for winning, and -5 for losing was found to be balanced. Another challenge was managing exploration stability. Early in training, when Q-values were mostly uninitialized, the agent performed random guesses. Gradual epsilon decay helped reduce randomness as learning progressed.

Through this process, several insights were gained. The combination of probabilistic modeling (HMM) and experience-based learning (Q-learning) significantly improved the agent's ability to make informed guesses. Over time, the agent learned to avoid repeating incorrect guesses and began prioritizing statistically stronger letters. It also became evident that shorter words were easier for the model to learn, while longer words exposed data sparsity problems in the HMM, suggesting the need for a larger dataset.

Strategies

For the HMM design, separate models were trained for words of different lengths. This improved accuracy, since the statistical structure of letter transitions changes with word length. Transition probabilities modeled how likely one letter follows another (for example, “q” is often followed by “u”), while emission probabilities captured the likelihood of seeing a particular pattern given hidden letter states. The HMM thus provided a probabilistic “hint” to the RL agent, narrowing down likely letters to guess.

In the reinforcement learning component, the state representation included the current masked word (for example, “_a_e”), the list of guessed letters, and the number of remaining attempts. Each action corresponded to choosing a single letter from the alphabet. The reward structure motivated efficient guessing while discouraging repetition. The Q-learning algorithm was chosen for its simplicity and stability, updating the Q-values using the Bellman equation to reinforce actions that led to better outcomes.

This hybrid strategy — combining the structured probabilistic reasoning of the HMM with the adaptive experience-based learning of Q-learning — gave the agent both intuition and adaptability.

Exploration vs. Exploitation

The agent managed exploration and exploitation using an epsilon-greedy policy. With a probability epsilon, the agent explored by making a random guess; otherwise, it exploited its knowledge by choosing the letter with the highest Q-value. Initially, epsilon was set to 1.0 to allow full exploration, then gradually decayed by a small factor (for example, 0.995 per episode) until reaching a minimum threshold, typically around 0.05. This gradual decay allowed the agent to learn broadly in the early stages and focus on refined exploitation later.

This approach worked effectively. In early training, the agent experimented with many letter combinations and word types, learning from both correct and incorrect outcomes. As epsilon decreased, the model relied more on its learned policy and the probabilistic cues from the HMM, resulting in steady improvement in accuracy and efficiency.

Future Improvements

If given additional time, several enhancements could further improve the model's performance. First, replacing the Q-table with a deep neural network (Deep Q-Learning) would allow it to generalize across much larger state spaces, especially for longer words. The HMM could also be upgraded using a larger corpus or replaced by a character-level language model such as an n-gram model or a recurrent neural network to capture more complex letter dependencies.

Another improvement would be to make the reward system adaptive — giving higher rewards for correct guesses when fewer attempts remain, encouraging the agent to take calculated risks. Implementing experience replay would stabilize learning by letting the agent learn from stored past transitions. Training multiple agents in parallel could speed up convergence and reduce overfitting to specific word patterns. Finally, adding an interactive visualization or a simple GUI would make it easier to observe how the agent's probability estimates evolve during play, offering deeper insights into its decision-making.

Summary

In conclusion, this project successfully developed a hybrid learning agent that combines the probabilistic reasoning of a Hidden Markov Model with the adaptive learning of Q-learning. The model demonstrated meaningful progress in learning to play Hangman intelligently, improving its success rate through self-play and probabilistic prediction. With future refinements such as deep learning integration, larger datasets, and adaptive rewards, the system could evolve into a highly capable and language-aware guessing agent.