**Hangman Solver: Data-Driven Strategy & Implementation**

**1. Dataset Analysis**

* The dataset contains fewer than 250K words, all unique.
* **Word length distribution:**
  + **Min-Max:** 1 - 29 (tested with 32 as a buffer).
  + **Mean Length:** 9.3 | **Median Length:** 9.
  + **Bell-shaped distribution** centered almost around 9.

**2. Strategic Inferences for Hangman**

**1. Early Vowel Guessing**

* Most frequent vowels: **e, a, i** → Prioritize them early in the game.
* **Weighted guessing:** If vowel ratio < 0.3, prioritize vowels; if > 0.6, guess a consonant.

**2. Leveraging Disjoint Testing Set**

* Training and testing sets are disjoint.
* **Avoid guessing words** that appear in the training set.
* If a guessed word belongs to the training set, pick the next best alternative.

**3. Positional Letter Frequency for Smarter Guesses**

* Construct a **relative letter position heatmap per word length**.
* Use **cosine similarity** to compare with training set distributions.
* Prioritize **high-certainty positions** for letter confirmation.

**4. High Information Gain Strategy**

* Since we only have 6 tries, **each incorrect guess must maximize information gain**.
* Use **entropy-based selection** to eliminate the most possibilities.

**5. Suffix-Based Exploitation**

* Common suffixes provide **strong hints for longer words**:
  + **-ing**, **-ion**, **-ous**, **-ted**, **-ate**, etc.
* Combine suffix probabilities with **positional data** for efficient guessing.

**3. It is important to first understand what not to use?**

**1. Basic Frequency Analysis**

* **Global letter frequency** is useful as a fallback but lacks **contextual accuracy**.

**2. Letter-Based Frequency Analysis**

* Length-based frequency (**Len(word) ± 2**) showed improvement.
* **Cosine similarity** verified alignment, but it ignored contextual dependencies.

**3. N-Gram Analysis**

* **Unseen n-grams** in testing data could lead to **zero probability errors**.
* Not reliable for **words outside the training set**.

**4. Rule-Based Strategies**

* **Vowel ratio** rarely exceeded **0.6** of total word length.
* **Hard-coded avoidance** of training dictionary guesses saved attempts.

**4. Alternative Approaches Considered**

**Bayesian & Probabilistic Methods**

* **Bayesian models** perform well on training data but **poorly on disjoint test sets**.
* Due to time constraints, **smoothing techniques were not explored**.

**Reinforcement Learning (RL)**

* Attempted **RL-based approach**, but could **not achieve convergence** within time constraints.
* A **reward-based RL structure** may offer long-term improvements.

**Char-Based Masked Language Model (MLM)**

* Created a **masked dataset (~3GB) for training**.
* **Exploited the top 11 global letter frequencies** as negative constraints.{If guessed wrong their value became 1}
* Trained on an **AMD GPU** due to dataset size constraints.
* **Most challenging but also the most rewarding** learning experience.

**5. Final Solution**

* **Letter frequency-based strategy** using **word length.**
* **MLM-based model** for **final guessing steps**.
* Combined **heuristic-based logic** for **high information gain within 6 guesses**.

**6. Acknowledgments**

* **Special thanks** to my **brother** for lending his **PC** for model training.
* Learned a **lot from this assignment**, especially in **heuristic-based modeling**.
* **Excited** to further refine these methods in **future AI/ML projects**.

Final accuracy was around :

overall success rate = 0.698