Part 1: Design Decisions for Developing the ML Algorithm

To develop an ML algorithm that can guess words based on their bigrams, several design decisions need to be made. As features are very diverse Decision tree will be a good fit for solving the problem. Below are the key design decisions made in developing the decision tree algorithm.

1.1. Criterion for Splitting Nodes

The primary criterion for splitting nodes in the decision tree is the presence or absence of specific bigrams in the bigram list of a word. The decision tree is built by examining each bigram and determining its importance in distinguishing between words in the dictionary.

- **Information Gain (IG)**: Information gain measures the reduction in entropy (uncertainty) when a node is split based on a particular feature (bigram).
- **Gini Impurity**: This criterion measures the frequency of misclassification at a node. The split that reduces the Gini impurity the most is chosen.

These metrics are computed for each bigram, and the bigram that maximizes information gain or minimizes Gini impurity is chosen for the split.

1.2. Stopping Criterion for Expanding Nodes

To prevent overfitting and ensure the decision tree does not become overly complex, clear stopping criteria for expanding nodes are necessary. These include:

- **Maximum Depth**: A maximum depth is set for the tree. If the current node reaches this depth, it becomes a leaf node.
- **Minimum Samples per Leaf**: If the number of samples (words) at a node is less than a specified threshold, splitting stops and the node becomes a leaf.
- **Minimum Information Gain**: If the information gain from a split is below a certain threshold, splitting stops.

1.3. Pruning Strategies

Pruning helps reduce the complexity of the tree and improve its generalization to unseen data. The following pruning strategies are employed:

- **Pre-Pruning**: Apply the stopping criteria mentioned above during the construction of the tree.
- **Post-Pruning**: After the tree is fully grown, cross-validation is used to prune branches that do not improve performance on a validation set.

1.4. Hyperparameters

The hyperparameters for the decision tree model include:

- **Max Depth**: The maximum depth of the tree.
- **Min Samples Split**: The minimum number of samples required to split an internal node.
- **Min Samples Leaf**: The minimum number of samples required to be at a leaf node.
- **Criterion**: The function to measure the quality of a split (`'gini'` or `'entropy'`).

Grid search with cross-validation is used to tune these hyperparameters and select the best combination.

1.5. Training and Validation

To train the decision tree model:

1. **Training Data Preparation**: Used the provided dictionary to create bigram sets for each word, remove duplicates, sorted them lexicographically, and retained only the first 5 bigrams.
2. **Model Training**: Trained the decision tree using the training data, splitting nodes based on the chosen criterion, and using the stopping and pruning strategies to control complexity.
3. **Validation**: Validated the model using cross-validation to ensure it generalizes well to unseen data.

1.6. Addressing Clashes

Given the constraints (up to 5 bigrams, potential 5-way clashes), the model will handle ties by making up to 5 guesses. If multiple words share the same bigram set, the model will include all possible words in its guess list, up to a maximum of 5.