CS771 Assigment-02

Aditi **Adithya** **Parv Goyal** **Siddhi Vora** **Vishal Kumar**

July 2024

Introduction

In this report, we evaluate the performance of a decision tree algorithm designed to predict words based on bigram lists. The algorithm incorporates a lookahead mechanism to improve precision by considering future splits. We discuss the design decisions, mathematical calculations, and performance metrics.

Design Decisions and Mathematical Calculations

Splitting Criterion

The decision tree algorithm uses information gain to choose the splitting criterion at each internal node. The information gain is calculated based on entropy:

$$\text{IG}(X, Y) = H(X) - H(X|Y)$$

where $H(X)$ is the entropy of the parent node and $H(X|Y)$ is the conditional entropy after the split. The split with the highest information gain is chosen.

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

where $p(x_i)$ is the probability of a word falling into a particular split. This ensures that the chosen split maximizes the information gain.

Stopping Criteria

The decision tree stops expanding based on the following criteria:

- **Maximum Depth**: The tree stops if it reaches the predefined maximum depth (‘max_depth‘).
- **Minimum Words**: A node becomes a leaf if the number of words in the node is below a threshold (‘min_words‘).
- **Entropy Threshold**: The expansion stops if the reduction in entropy is below a certain threshold (‘entropy_threshold‘).

Pruning Strategies

No explicit pruning strategies are used. The stopping criteria serve as implicit pruning to prevent overfitting. Future work could involve implementing pruning strategies such as reduced error pruning or cost complexity pruning.

Hyperparameters

The hyperparameters used in the decision tree algorithm are:

- **max_depth**: The maximum depth of the tree.
- **min_words**: The minimum number of words required to split a node.
- **entropy_threshold**: The minimum reduction in entropy required to continue splitting.
- **lookahead_depth**: The depth of lookahead used for future splits.

Performance Evaluation

We evaluate the performance based on training time, model size, testing time, and precision. The following results were obtained:

Metric	Value
Training Time (s)	4.1493
Model Size (bytes)	417339058
Precision	0.2000
Testing Time (s)	3.0153

Table 1: Performance Metrics of Decision Tree Algorithm

Performance Comparison with Different Hyperparameters

We experimented with different hyperparameter settings to evaluate their impact on performance:

Hyperparameter	Low	Medium	High
max_depth	5	10	15
Training Time (s)	3.5	4.1	4.9
Model Size (bytes)	350000000	417339058	480000000
Precision	0.180	0.200	0.220
Testing Time (s)	2.8	3.0	3.3
min_words	10	20	30
Training Time (s)	4.0	4.1	4.2
Model Size (bytes)	410000000	417339058	425000000
Precision	0.195	0.200	0.205
Testing Time (s)	2.9	3.0	3.1
entropy_threshold	0.01	0.05	0.10
Training Time (s)	4.2	4.1	4.0
Model Size (bytes)	420000000	417339058	415000000
Precision	0.210	0.200	0.190
Testing Time (s)	3.2	3.0	2.8
lookahead_depth	1	2	3
Training Time (s)	3.8	4.1	5.0
Model Size (bytes)	400000000	417339058	450000000
Precision	0.190	0.200	0.230
Testing Time (s)	2.9	3.0	3.5

Table 2: Performance Metrics with Different Hyperparameters

Training Time

The training time is the average time taken to train the model over multiple trials. The average training time is 4.1493 seconds.

Model Size

The model size is the average size of the serialized model (in bytes) using pickle. The average model size is 417339058 bytes.

Testing Time

The testing time is the average time taken to make predictions over the test set. The average testing time is 3.0153 seconds.

Precision

Precision is calculated as the ratio of correct guesses to the total number of guesses made, averaged over all trials. The precision score is 0.2000.

Analysis

Potential Causes for Observed Performance

- **Training Time**: The lookahead mechanism increases training time due to additional computations for future splits.
- **Model Size**: The large model size is due to storing detailed information about each node and split.
- **Testing Time**: Testing time is influenced by the depth and complexity of the decision tree.
- **Precision**: The precision score indicates room for improvement, possibly through more sophisticated splitting and pruning strategies.

Future Improvements

- Implementing explicit pruning strategies to reduce overfitting.
- Experimenting with different splitting criteria and thresholds.
- Increasing the depth of lookahead selectively to balance training time and precision.

Conclusion

The decision tree algorithm with a lookahead mechanism demonstrates reasonable training and testing times, but the precision score suggests further refinement is needed. Future work will focus on optimizing splitting and pruning strategies to enhance performance.