Overview
The goal of this project is the effective optimization of transformer models using Dynamic Token Pruning, where tokens are reduced dynamically depending on their saliency score. The pruned model is effective since the computational cost is diminished with a minimum loss in performance, hence applicable on real-world applications that require computational efficiency.

Key Contributions

Locate the Efficiency Bottleneck
Profiling pointed out self-attention and feed-forward layers as the major contributors w.r.t. FLOPs and runtime.
Baseline Model FLOPs: 12.908G, CUDA Time: 12.490ms.

Optimization Technique
Dynamic Token Pruning:

Compute the token saliency scores using the L2 norm.
Prune tokens with assigned saliency score under some threshold, e.g., 13.0.
Dynamically adjust the sequence length at each layer.
Results
Accuracy: Baseline model is 52.5%, and the pruned model is 52.5%.
Efficiency Improvements:
FLOPs Reduction: 31.2% (12.908G → 8.874G).
CUDA Time Reduced: 20.2% (12.490ms → 9.968ms).
Implementation
Baseline and Pruned Models:

Baseline: Transformer encoder.
Pruned: Improved at each layer with token pruning.
Profiling: Measured FLOPs and memory usage using PyTorch profiling tools.
Conclusion
Dynamic Token Pruning represents a scalable solution toward the goal of optimizing transformer efficiency. It reduces computational overhead significantly while not sacrificing model accuracy, reaching both efficiency and performance goals.