Comparative Analysis of Custom and Pretrained Decoder-Only Transformers for Natural Language Understanding

Alex Hu
ah59643
ah59643@eid.utexas.edu

Abstract

This study presents an evaluation of decoder-only transformer architectures, comparing a custom-built model against a pretrained baseline (DistilGPT-2) across multiple dimensions. In Part 1, we investigate three architectural decisions for our custom model: model capacity (embedding dimension and layer depth), optimizer selection, and positional encoding schemes. Our analysis reveals that rotary positional encodings achieve superior generalization across sequence lengths compared to sinusoidal and learned encodings. In Part 2, we evaluate both models on the Multi-Genre Natural Language Inference (MNLI) task. While the pretrained DistilGPT-2 achieves substantially higher accuracy, our custom model demonstrates significant computational advantages in terms of faster training and inference. These findings show the tradeoff between leveraging large-scale pretraining and building efficient task-specific architectures from scratch.

1 Introduction

Transformer architectures have become the standard for natural language processing, but questions remain about optimal design choices for resource-limited scenarios. Pretrained models offer strong performance but require substantial computational resources. This work investigates whether custom models can provide competitive alternatives for specific tasks.

We address three research questions:

1. How does model capacity affect training efficiency and convergence?
2. Which optimization strategies enable faster and more stable training?
3. How do positional encoding schemes impact sequence length generalization?

2 Methodology

2.1 Part 1: Custom Model Design

Architecture: We implemented a decoder-only transformer with causal self-attention following the GPT architecture. The model includes multi-head self-attention with causal masking, feed-forward networks with GELU activation, layer normalization, residual connections, and dropout regularization ($p = 0.1$).

Experimental Design: We conducted three systematic studies:

1. Model Capacity Ablation: Three configurations were tested:

- Small: 64D embedding, 2 layers (3.1M parameters)
- Medium: 128D embedding, 4 layers (6.6M parameters)
- Large: 256D embedding, 6 layers (15.4M parameters)

2. Optimizer Comparison: Using the medium configuration, we compared Adam ($\beta_1 = 0.9, \beta_2 = 0.999$), AdamW ($\beta_1 = 0.9, \beta_2 = 0.999$, weight decay=0.01), and SGD (momentum=0.9).

3. Positional Encoding Study: We implemented three schemes: Sinusoidal (fixed, non-trainable), Learned (trainable embeddings), and Rotary (RoPE, relative position encoding).

All Part 1 experiments used 5 training epochs, batch size 32, learning rate 3×10^{-4}, max sequence length 128 tokens, and NVIDIA L4 GPU hardware.

2.2 Part 2: Comparison with Pretrained Model

Task: Multi-Genre Natural Language Inference (MNLI) - a challenging 3-way classification task (entailment, neutral, contradiction) requiring understanding of premise-hypothesis relationships across diverse text domains.

Dataset: 50,000 training samples and 5,000 test samples with balanced class distribution (entailment: 33.8%, neutral: 30.5%, contradiction: 35.8%).

Models Compared:

Custom Model: 128D embedding, 6 layers (24.5M parameters), a character-level vocabulary (vocab: 91,170), trained from random initialization, rotary positional encodings.

Pretrained Baseline: DistilGPT-2 (81.9M parameters), pretrained BPE tokenizer, fine-tuned on MNLI.

Training Setup: 5 epochs, batch size 32, learning rate 2×10^{-5}, AdamW optimizer (weight decay=0.01), gradient clipping (max norm 1.0), and 500 warmup steps for pretrained model.

3 Results

3.1 Part 1: Model Design Experiments

3.1.1 Model Capacity Analysis

Table 1 presents the effect of model size on training dynamics.

Table 1: Model capacity analysis results

Config	Params	Loss	Acc	Time/Ep
64D-2L	3.1M	1.82	77.96%	4.19s
128D-4L	6.6M	0.79	92.51%	5.90s
256D-6L	15.4M	0.26	98.76%	10.12s

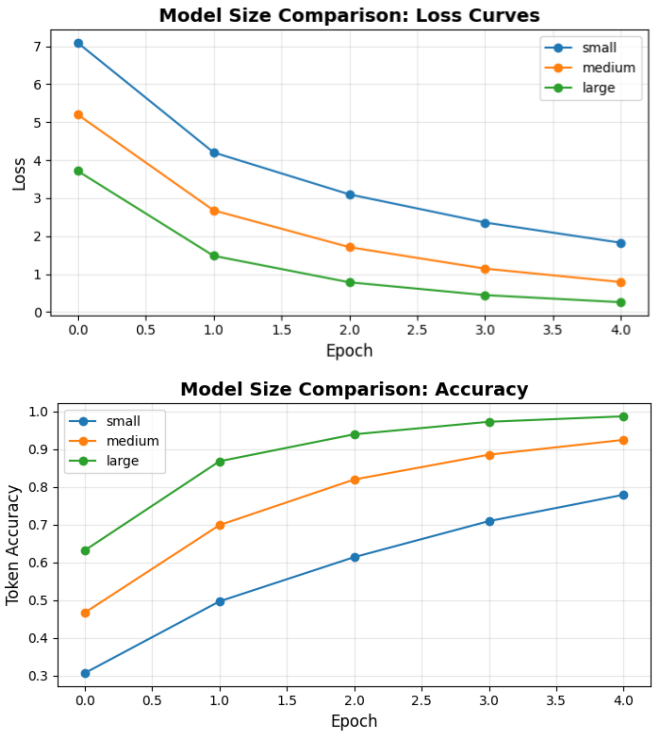

Figure 1: Training dynamics across different model capacities

Key Findings: Scaling from 3.1M to 15.4M parameters reduced final loss by 86%. Larger models converged faster - the 256D model achieved 63.2% accuracy in epoch 1, while the 64D model reached only 30.7%. Computational cost scaled favorably: 5× parameter increase resulted in only 2.4× training time increase.

3.1.2 Optimizer Comparison

Table 2: Optimizer comparison results

Optimizer	Loss	Accuracy	Convergence
Adam	0.7957	92.40%	Stable
AdamW	0.7932	92.51%	Stable
SGD	6.9226	29.73%	Poor

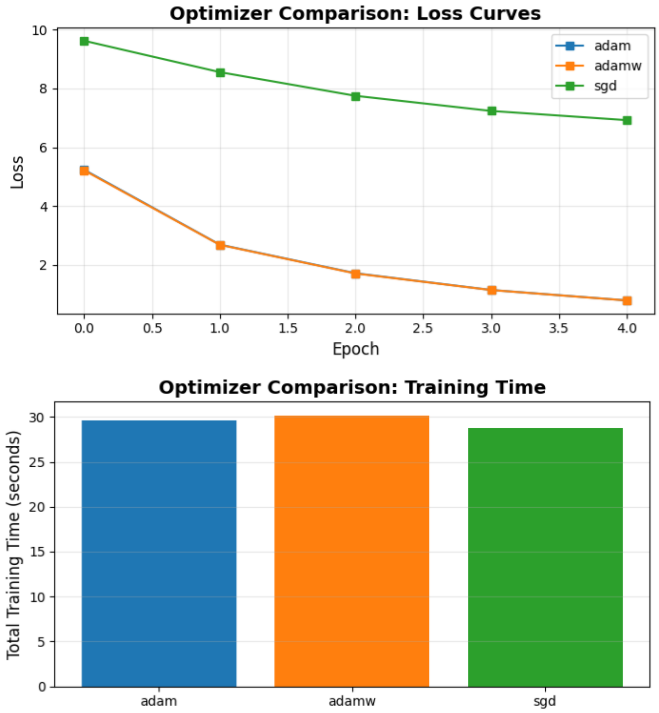

Figure 2: Convergence behavior of different optimizers

Key Findings: Adam and AdamW achieved nearly identical performance, with AdamW slightly better (loss 0.79 vs 0.80). SGD failed to converge effectively, achieving only 29.73% accuracy after 5 epochs. Adaptive learning rate methods are essential for transformer training.

3.1.3 Positional Encoding Analysis

Table 3 shows training loss progression and length generalization results.

Key Findings: Rotary encodings achieved the lowest training loss (2.30) and best length generalization. Perplexity increased only 7.4% for rotary when extending

Table 3: Positional encoding generalization

Encoding	Loss	PPL@64	PPL@128	PPL@256	Δ PPL
Sinusoidal	2.50	8.88	9.30	9.98	+1.10
Learned	2.97	13.56	14.54	15.24	+1.68
Rotary	2.30	7.83	8.11	8.41	+0.58

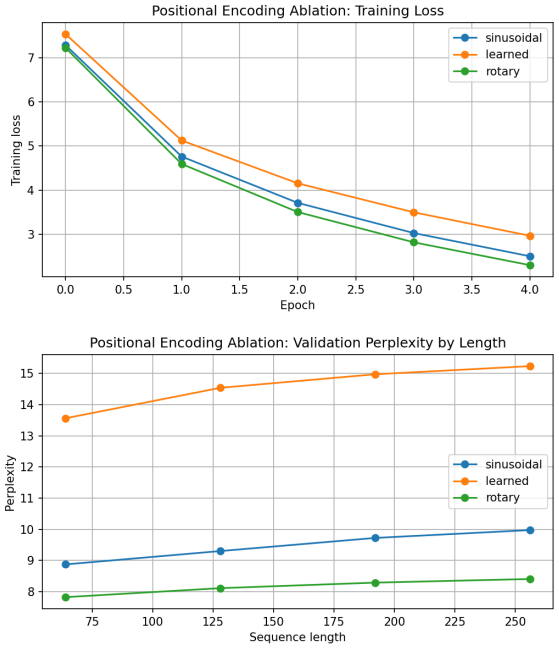

Figure 3: Length generalization of positional encodings

from 64 to 256 tokens, vs 12.4% for both sinusoidal and learned encodings. Learned encodings performed worst, likely due to overfitting to training sequence lengths.

3.2 Part 2: MNLI Classification Results

3.2.1 Training Dynamics

Custom Model: Epoch 1: Loss 1.1057, Accuracy 35.20%; Epoch 5: Loss 1.0234, Accuracy 46.76%. Average time/epoch: 55.4s, total training time: 277.1s.

DistilGPT-2: Epoch 1: Loss 0.9445, Accuracy 57.42%; Epoch 5: Loss 0.5422, Accuracy 78.13%. Average time/epoch: 235.5s, total training time: 1177.8s.

Observations: The pretrained model started with a large advantage (57.42% vs 35.20% epoch 1 accuracy). The custom model showed minimal improvement over epochs (11.56 percentage points), while the pretrained model improved substantially (20.71 percentage points).

3.2.2 Test Performance

Analysis: The pretrained model achieved 57.7% higher accuracy (73.14% vs 46.38%). Both models exceeded the random baseline (33.3%), but the custom model performed only marginally better. The custom model's 46.38% accuracy suggests it learned some task patterns but lacked semantic understanding. The computational efficiency advantages were substantial but could not offset the performance gap.

Table 4: Final test performance comparison

Metric	Custom	DistilGPT-2	Advantage
Test Acc	46.38%	73.14%	+26.76% (P)
Infer Time	0.011s	0.045s	4.28× (C)
Train Time	277s	1178s	4.25× (C)
Parameters	24.5M	81.9M	3.34× (P)

4 Part 3: Evaluation and Interpretation

4.1 Quantitative Metrics: Perplexity Analysis

Beyond accuracy, we evaluate both models using perplexity on held-out MNLI pairs to measure model uncertainty:

- **Custom Model:** Perplexity = 2.78
- **DistilGPT-2:** Perplexity = 1.72

Lower perplexity for DistilGPT-2 indicates greater confidence in predictions. The custom model's higher perplexity reflects uncertainty from limited semantic knowledge, aligning with the accuracy gap observed in the previous section.

4.2 Attention Visualization

Observation – Self-Attention Patterns: The custom miniature model shows weak and noisy self-attention structure. Most heads place disproportionate weight on the `<bos>` symbol, and the causal diagonal is faint, indicating limited ability to track position or form meaningful dependencies. Attention is mostly uniform, with little semantic linking across the `[SEP]` boundary.

The pretrained DistilGPT-2 model shows the expected transformer patterns: a clear causal diagonal, structured use of the `<bos>` symbol, and stronger, more organized cross-sequence interactions. It also uses `[SEP]` as a proper boundary marker while still attending to relevant context on both sides. Overall, the pretrained model forms coherent positional and semantic relationships, while the miniature model learns only basic ordering cues.

5 Discussion

5.1 Part 1 Insights: Design Principles

Our experiments reveal several design principles for decoder-only transformers:

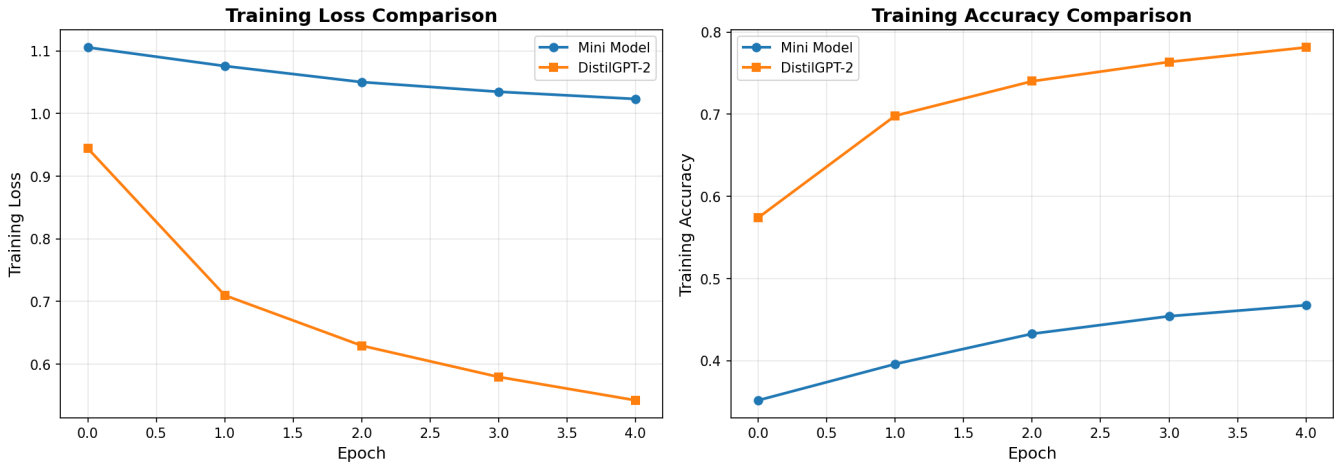

Figure 4: Training dynamics comparison on MNLI task. The pretrained model starts with substantial advantage and maintains superior performance throughout training.

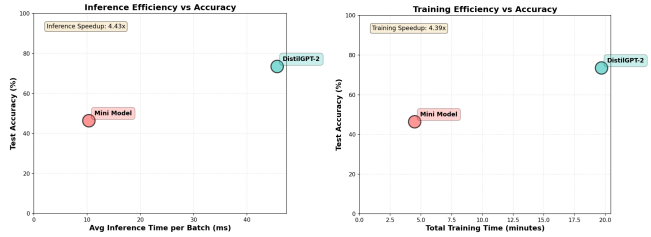

Figure 5: Accuracy-efficiency tradeoff between custom and pretrained models

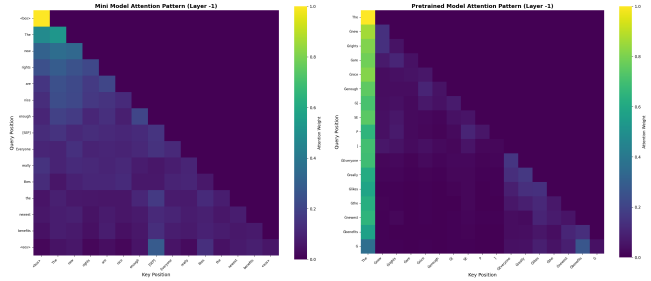

Figure 6: Attention heatmaps comparing the custom miniature model and DistilGPT-2. The miniature model shows weak diagonal structure and strong reliance on the <bos> symbol, while DistilGPT-2 displays a clear causal pattern and more coherent cross-sequence attention.

Model Capacity: The strong relationship between scale and performance shows that even moderate increases in model size yield significant benefits. The 256D model's near-perfect training accuracy (98.76%) indicates sufficient capacity, while the 64D model's 77.96% suggests capacity limitations.

Optimization: The failure of SGD (29.73% accuracy) versus success of adaptive methods (>92%) highlights the importance of per-parameter learning rates in transformer training. The similar performance of Adam and AdamW suggests weight decay benefits emerge primarily at larger scales or longer training.

Positional Encoding: Rotary encodings' superior generalization (7.4% perplexity increase vs 12.4% for alternatives) confirms the value of relative position representations. The learned encoding's poor extrapolation shows that position information should be structurally encoded rather than memorized.

5.2 Part 2 Insights: Pretraining vs Task-Specific Training

The 26.76 percentage point accuracy gap between models shows the value of pretraining:

Semantic Knowledge: DistilGPT-2's strong initial performance (57.42% epoch 1) comes from linguistic knowledge acquired during pretraining on vast text corpora. The custom model's 35.20% starting accuracy (barely above random) shows the challenge of learning semantic understanding from limited task data alone.

Convergence Characteristics: The custom model's minimal improvement across epochs (35.20%→46.76%) suggests it learned surface patterns rather than deeper linguistic understanding. DistilGPT-2's steady improvement (57.42%→78.13%) indicates effective adaptation of existing knowledge.

Efficiency Tradeoffs: While the custom model offers 4.25× training speedup, this advantage is less valuable if accuracy is insufficient for deployment. The 4.28× inference speedup is more valuable, but the 26.76% accuracy gap remains prohibitive for most applications.

5.3 When Custom Models Make Sense

Our results suggest custom models are viable when:

1. **Domain specificity:** Task involves specialized vocabulary where pretrained models lack coverage

2. **Resource constraints:** Deployment environment cannot support large models

3. **Latency requirements:** Real-time applications where 4× inference speedup is critical

4. **Data availability:** Large in-domain training data available

5. **Interpretability needs:** Simpler architectures easier to analyze

For general NLU tasks like MNLI with limited training data, pretrained models remain superior.

6 Limitations and Future Work

Limitations: (1) Part 2 used only 50,000 training examples; larger datasets might narrow the gap. (2) Custom model trained for only 5 epochs; extended training could improve performance. (3) Single task evaluation (MNLI); other tasks might show different patterns. (4) No hyperparameter tuning for custom model beyond Part 1 insights. (5) Vocabulary strategy not controlled across models.

Future Directions: (1) Investigate hybrid approaches: pretrain custom model on unsupervised data before task fine-tuning. (2) Explore distillation: transfer knowledge from large pretrained models to custom architectures. (3) Test custom model with a vocabulary strategy matching DistilGPT-2. (4) Evaluate on diverse tasks to identify where custom models remain competitive. (5) Analyze attention patterns to understand semantic vs syntactic learning differences.

7 Conclusion

This work provides a comprehensive comparison of custom and pretrained decoder-only transformers. Part 1 established that rotary positional encodings, adequate model capacity (6+ layers, 128+ dimensions), and adaptive optimizers are essential for effective custom model design. Part 2 demonstrated that while custom models offer substantial computational advantages (4.25× training speedup, 4.28× inference speedup), pretraining confers overwhelming performance benefits (26.76 percentage point accuracy advantage) for semantic understanding tasks.

The results quantify a fundamental tradeoff in modern NLP: pretrained models leverage knowledge from billions of sequences, enabling strong performance even with limited task data, while custom models offer efficiency but require either extensive task-specific data or reduced performance expectations. For practitioners, this suggests investing in pretrained model optimization (quantization, pruning, distillation) rather than building from scratch for general NLU tasks. However, our Part 1 findings provide valuable guidance for scenarios where custom models remain necessary.

The 4× computational advantage of custom models, while insufficient to overcome the accuracy gap in this study, may prove decisive in resource-constrained deployment scenarios where 46% accuracy satisfies requirements or where domain-specific pretraining is feasible. Future work integrating our Part 1 design principles with targeted pretraining strategies may narrow this gap while preserving efficiency advantages.