Impact of Transformer Depth on Small Datasets for Chatbot Models

Anonymous EMNLP submission

Abstract

This study investigates the impact of transformer layer variations (2, 4, 8, and 16 layers) on the performance and efficiency of chatbot models trained on a small curated dataset. From the original DailyDialog dataset containing 76,052 samples, 10,000 instances were selected after preprocessing, including duplicate removal and filtering by token count (3–60). Evaluation metrics include BLEU, ROUGE, METEOR, and qualitative assessments. Results indicate

1 Introduction

Modern NLP models, particularly transformers, have demonstrated exceptional performance across tasks with large-scale datasets. However, challenges arise when applying such architectures to small datasets. This study evaluates the impact of varying transformer depth (2, 4, 8, and 16 layers) on model performance and computational efficiency in a small-data setting.

2 Background and Related Works

TBU

3 Method

3.1 Dataset Preprocessing

The experiments were conducted using the DailyDialog dataset, a widely-used datasets for dialog generation tasks. The original dataset consists of 76,052 multi-turn conversational samples. To dapt the data for this study, the following preprocessing steps were applied:

1. **Duplicate Removal**: All duplicate samples were removed to ensure diversity in the training data and minimize inference of perplexity in model training.

2. **Token-length Filtering**: Dialog samples were filtered to retain only those with token counts between 3 and 60, ensuring meaningful yet manageable input lengths. 95% of total training datasets are included in this range.

3. **Sample Selection**: A subset of 10,000 instances was randomly selected from the preprocessed dataset to simulate a small-scale training scenario.

4. **Tokenization**: Tokenization was performed using the NLTK punkt tokenizer and a vocabulary size of 15,000 tokens was constructed.

3.2 Transformer Model Configuration

The configuration of the model follows the standard transformer design with a slight modification in embedding dimension size and variation in encoder and decoder layer numbers, ranging from 2, 4, 8 to 16.

- **Embedding Dimension (d_{model}):** 256
  The weights of the embedding layers in the encoder and decoder are shared, since the source and target languages are the same (English).

- **Number of Attention Heads (n_{heads}):** 8.

- **Feedforward Layer Dimension (d_{ff}):** 1024.

- **Dropout Rate:** 0.1.

- **Positional Encoding:** Maximum sequence length of 128 tokens.

3.3 Training Setup

The models were trained using the Adam optimizer with the following hyperparameters:

- **Epochs per model**: 10

- **Learning Rate Scheduler**: Warm-up steps of 4,000, followed by inverse square root decay.

- **Optimizer Parameters**:

$$\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1e - 9$$

- **Batch Size**: 64
- **Loss Function**: Sparse Categorical Crossentropy, with a padding mask applied to ignore <pad> tokens in the input and target sequences.

3.4 Evaluation Metrics

To assess model performance and training efficiency, following evaluations were performed:

1. **Quantitative Metrics for Text Generation** The 3 following evaluation metrics have scores between 0 to 1, and score above 0.2 is suggested for generative models.
    - **BLEU**: Measures n-gram overlap between generated responses and ground truth.
    - **ROUGE**: Evaluates overlap of longer sequences, and ROUGE-L was used for the experiments.
    - **METEOR**: Consider semantic similarity, penalizing over-reliance on exact matches
2. **Computational Efficiency Metrics**:
    - Training Time Per Epoch: Time taken to complete one training epoch
    - Convergence Speed: Number of epochs required to achieve optimal performance

4 Result

4.1 Training Efficiency

1. Time per Epoch: "Image"
    - The 2-layer model exhibited the fastest training time (91 seconds per epoch), making it highly efficient for small datasets.
    - Deeper models (e.g. 16-layer) had significant higher training times (658 seconds per epoch), increasing computational costs substantially.
2. Convergent Speed:
    - The 2-layer model achieved competitive performance in early epochs (e.g., within 10 epochs), particularly in METEOR and ROUGE, which suggests that shallower models may balance efficiency and performance during early-stage training with small dataset.
    - Deeper models, especially the 16-layer model, require more epochs to converge effectively, as indicated by their higher loss values.
    - Yet, the fluctuation of model performance observed within the 10 epochs indicates that the training epochs set for the experiment may not be sufficient for all models.

4.2 Model Performance

1. **BLEU**:
    - Across all epochs, the 2-layer model consistently recorded a BLEU score of 0, indicating its inability to generate text with meaningful n-gram overlap.
    - Among the other models, the ranking of BLEU performance was 16 > 8 > 4 > 2, with the 16-layer model achieving the highest BLEU score.
    - Yet, BLEU scores of all models exceeding low (near-zero values), suggesting that BLEU might not be a reliable metric for assessing the quality of responses in this experiment.
2. **ROUGE**:
    - The ranking for ROUGE performance was 8 > 2 > 16 > 4, with the 8-layer model achieving the highest ROUGE score. However, the 2-layer model performed better than deeper models like 16 and 4 layers in ROUGE.
    - Significant fluctuations in ROUGE scores were observed within the first 10 epochs, suggesting instability in capturing sequence-level overlap during training.
    - Similar to BLEU, ROUGE values were all below 0.08, indicating that none of the models could consistently generate meaningful textual content.
3. **METEOR**:
    - The 2-layer model outperformed all other models in METEOR at maximum, followed by 4 > 16 > 8, showcasing its ability to capture similarity better than

Table 1: Performance Metrics by Model Layer Depth

Layer	BLEU Avg	ROUGE Avg	METEOR Avg	Observations
2	0	0.0737	0.0413	Strong semantic similarity and sequence overlap in early training.
4	$4.58e - 157$	0.0330	0.0433	Moderate performance; outperformed by 2-layer in METEOR and ROUGE.
8	$2.55e - 157$	0.0276	0.0427	Best ROUGE performance but struggled in METEOR.
16	$8.90e - 157$	0.0248	0.0434	Best BLEU performance but less semantic relevance.

deeper models in the early stages of training.

- Like BLEU and ROUGE, all METEOR scores were below 0.08, further supporting the conclusion that the generated text lacked meaningful content.

5 Conclusion

This study analyzed the impact of transformer layer depth on model performance and training efficiency for a chatbot model trained on a small dataset. Experiments with models having 2, 4, 8, and 16 layers revealed several key insights:

- The **2-layer model** demonstrated strong semantic similarity and sequence overlap in the early stages of training, as reflected by the highest ROUGE and METEOR scores among shallow models. Its computational efficiency makes it well-suited for low-resource or early-training scenarios.

- The **8-layer model** achieved the best overall balance of performance and efficiency, with strong ROUGE scores and moderate METEOR values. This configuration is recommended for generating more stable results over prolonged training on small datasets.

- The **16-layer model** performed best in terms of BLEU score but showed limited semantic relevance and higher computational costs, making it less practical for small datasets or limited epochs.

- Across all layers, BLEU, ROUGE, and METEOR scores remained below 0.08, indicating the difficulty of generating meaningful text with the given dataset size and experimental setup.

In conclusion, while deeper models such as the 16-layer transformer excel in BLEU, the results highlight the suitability of shallow models (e.g., 2 layers) for efficient and effective training in low-resource scenarios. For more balanced performance, the 8-layer configuration offers the best trade-off between quality and computational cost. Future work could explore techniques like data augmentation or transfer learning to improve the overall performance of transformers on small datasets.

6 Acknowledgements

TBU

7 References

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention is all you need.* Advances in neural information processing systems, 30, 2017.