CS 6120: nGPT, BART and PEGASUS: A Comparative Study

Sanjiv Shrirang Joshi, Shishir Kallapur and **Pratheesh**
Khoury College of Computer Sciences
Northeastern University
Boston MA USA

Abstract

This study evaluates and compares the performance of nGPT, Nvidia's model, against two state-of-the-art abstractive text summarization models: PEGASUS (Google) and BART (Facebook). The primary objective is to assess nGPT's claims of achieving 4-20x faster training times and improved stability for Large Language Models (LLMs) in the context of abstractive text summarization. Key steps include training base models of PEGASUS and BART on standardized datasets, adapting nGPT for abstractive summarization to optimize its performance, and conducting a comprehensive comparison of the models in terms of performance, training speed, and stability. The findings aim to validate nGPT's efficiency and stability improvements.

1 Introduction

This project **critically evaluates Nvidia's claims** regarding the **nGPT** model's training speed and stability within the context of **Large Language Models (LLMs)**. The goal is to verify whether nGPT achieves the claimed **4-20x faster training** and compare its performance to state-of-the-art models like **PEGASUS** (Google) and **BART** (Facebook).

The **interesting aspect of this project is** evaluating **nGPT**—a model not originally designed for text summarization—on this task. This unconventional approach allows us to explore **training efficiency** in LLMs and assess whether faster training can be achieved without sacrificing model performance. By comparing nGPT to PEGASUS and BART, we aim to understand whether **efficiency gains** can be achieved without compromising quality.

This project is motivated by the need for **efficient, cost-effective LLM training**, which is crucial for making AI technologies more accessible while reducing computational costs. Our goal is to **understand the trade-offs** between training speed and performance in real-world scenarios.

2 Background/Related Work

Since our focus is on proving the efficacy of nGPT against BART and PEGASUS we referred to the following papers for in-depth understanding. We also tried implementing the model in pytorch and had to refer some of these papers for how it usually is.

nGPT: Normalized Transformer with Representation Learning on the Hypersphere (Loshchilov et al., 2024) paper by Nvidia introduces the nGPT model, which claims significantly faster training speeds. This paper is central to our project as we aim to evaluate the validity of these claims and assess nGPT's performance in comparison to established models.

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Devlin et al., 2018) introduced the transformer architecture that has since set new standards for a variety of tasks. BERT's impact on the field laid the foundation for subsequent advancements like PEGASUS and BART, which are central to our work.

PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization (Zhang et al., 2020a) presents a model optimized for abstractive summarization by focusing on extracting gap-sentences for pre-training. This model serves as one of our primary baselines for comparing nGPT's summarization capabilities.

BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension (Lewis et al., 2019) introduced a denoising approach to pre-training for sequence-to-sequence models. Its design for text generation and summarization tasks makes it another relevant baseline for our evaluation of nGPT in summarization.

ROUGE: A Package for Automatic Evaluation of Summaries (Lin, 2004) is a widely used

metric for evaluating the quality of generated summaries. ROUGE is essential in our project as we use it to measure and compare the performance of PEGASUS, BART, and nGPT.

Efficient Transformers: A Survey (Tay et al., 2020a) provides an overview of various approaches to making transformer models more efficient. This paper is key for understanding the strategies that could be applied to nGPT to improve its training efficiency and stability.

Transformers for Text Summarization: A Comprehensive Review (Zhang et al., 2020b) reviews various transformer models applied to text summarization. This paper is useful for understanding the state of the art in text summarization and positioning our results against existing work like PEGASUS and BART.

Efficient Attention: A Survey (Tay et al., 2020b) reviews improvements in the transformer attention mechanism, a critical component for model efficiency. Understanding these advancements is important as nGPT's speed improvements are partly attributed to enhancements in the attention mechanism.

3 Data

We used the CNN/DailyMail dataset, a benchmark for abstractive text summarization, containing **over 300,000 article-summary pairs** from CNN and DailyMail. Each entry consists of an article, a corresponding highlight (summary), and a unique identifier (ID). The highlights are human-written summaries of **50-100 words**. Articles themselves are on an average 700 words in length. The dataset is divided into **287,113 training pairs**, **13,368 validation pairs**, and **11,490 test pairs**.

We trained the models using the articles as input and targeting the highlights as the model's summaries. The dataset requires no preprocessing. It is specifically designed for abstractive summarization, where the goal is to generate concise summaries rather than extract sentences verbatim.

4 Methods

The models we use, their implementation details in brief and source code:

nGPT - nGPT-pytorch

nGPT is a normalized Transformer designed to improve training stability and efficiency. Its key mechanism involves normalizing values onto a hypersphere, ensuring all embeddings are scaled between -1 and 1. Nvidia modifies the standard Transformer by removing normalization layers (e.g., RMSNorm, LayerNorm) and normalizing matrices such as $W_q, W_k, W_v, W_o, W_u, W_\nu$ after each training step. Additional scaling adjustments in attention mechanisms, MLP blocks, and logits enhance model stability and convergence. These innovations make nGPT particularly suitable for efficient and robust LLM training.

BART - facebook/bart-base

BART(Bidirectional and Autoregressive Transformer) is a denoising autoencoder developed by Facebook. It features a bidirectional encoder like BERT and a left to right decoder like GPT which makes it useful for natural language processing tasks. Its key mechanism involves corrupting input text using an arbitrary noising function and learning to reconstruct the original text.

PEGASUS - google/pegasus-large

Pegasus (Pre-training with Extracted Gap-sentences for Abstractive Summarization) is a model developed by Google. It removes crucial sentences from the text and trains the model to reconstruct them, simulating a summarization task. This enables Pegasus to focus on key information, enhancing the summarization capabilities of its encoder-decoder architecture.

Approaches to Unknown Words, Training, and Decoding

In nGPT, unknown words are managed inherently through its character-level training approach. This method enables the model to learn character sequences and generate valid words, although it may also produce non-existent words. We use a patience=3 to prevent overfitting. The dataset is preprocessed into fixed-length sequences to match nGPT's input requirements. In contrast, PEGASUS employs SentencePiece subword tokenization strategy and BART uses Byte-Pair Encoding tokenization strategy that breaks down words into recognizable subunits, allowing for more flexible handling of unknown words and improving the model's ability to generate coherent text from unseen vocabulary.

All models share similar training processes in terms of epochs and general objectives, but they diverge significantly in their decoding techniques.

nGPT generates output character by character, which must be decoded from ASCII format to form coherent text. On the other hand, PEGASUS and BART use beam search decoding strategy.

During the training, validation, and generation phases, which occur every two epochs for both models, metrics such as training and validation losses, inference times, memory usage, and ROUGE scores are monitored to gauge the models' performance. All models conduct validation experiments to test various context lengths and their impact on loss, with perplexity scores providing additional insights into each model's learning efficiency and capability to handle diverse text complexities.

5 Experiments

Our own implementation

We initially developed nGPT from scratch in Python, following the research paper. Early results were promising, but scaling the model proved challenging within our project timeline. We shifted to an open-source implementation to adapt to the dataset, learning valuable lessons about transformer architectures, scalability, and practical application of research in real-world scenarios.

Training and Hyperparameters

The models were trained on 15,000 samples from the CNN/DailyMail dataset, with an additional 1,500 samples for validation. The architecture utilized 100M parameters, but we did not get there directly.

We experimented with various hyperparameter combinations to determine the optimal settings. Initially, we started with smaller dimensions (eg. `dim=256`), reduced depth (eg. `depth=6`), and smaller head size (eg. `dim_head=64`) and other model specific parameters, but these configurations led to premature convergence and limited model capacity. Gradually, we increased the dimensions to higher values, depths, and head size, which improved the models' learning and generalization capabilities.

The optimizer used was Adam, with suggested initial learning rates from the models' papers. The sequence length was set to 512, and gradient accumulation was configured to 2 steps to manage training on limited hardware resources.

This experimentation process highlights the iterative tuning of hyperparameters to strike a balance between model capacity and convergence efficiency.

6 Results

Consistency with Data and Models

The results for nGPT were consistent with the provided data and model configuration. By training on a sample of 15,000 examples from the CNN/DailyMail dataset (selected due to resource constraints), 1500 samples for validation, a few from the test set for human annotation, the model demonstrated effective learning and generalization, aligning with our primary goal of evaluating efficiency rather than achieving full-scale training.

Evaluation Metrics

The BART model was trained on Nvidia GTX 1660 Ti, PEGASUS on NVIDIA A100, and nGPT on Nvidia A10Gx4. We evaluated the models using training and validation loss, ROUGE scores, memory usage, inference times and total training times providing a comprehensive analysis. As shown in Figure 3, nGPT achieves a ROUGE score nearly twice that of BART and PEGASUS, illustrating superior performance given the same dataset and model size.

Loss Trends Across Epochs

Figure 1 and Figure 2 shows the training and validation loss trends for nGPT, PEGASUS, and BART. nGPT displays superior initial convergence, starting training at significantly lower loss values around 2.0 compared to BART and PEGASUS around 10.0. Throughout the training, nGPT maintains lower loss values with minimal oscillation, highlighting its robust generalization capabilities.

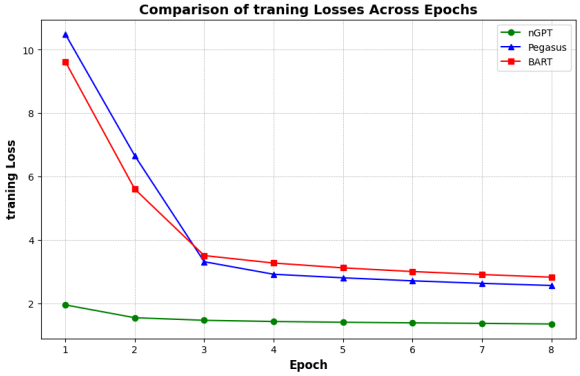

Figure 1: Comparison of Training Losses Across Epochs

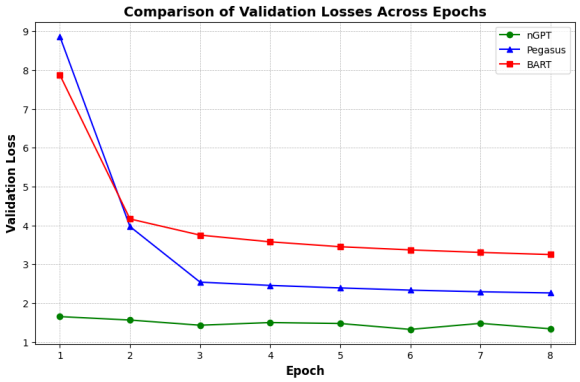

Figure 2: Comparison of Validation Losses Across Epochs

Performance Comparison

From Figure 4 we can observe that nGPT has used much lower memory on average compared to BART and PEGASUS validating the claim of improved stability. Even the training and inference times for nGPT are considerably better than BART and PEGASUS.

Model	ROUGE-1	ROUGE-2	ROUGE-L
nGPT	0.289	0.038	0.136
PEGASUS	0.125	0.0125	0.095
BART	0.122	0.013	0.101

Figure 3: ROUGE scores

Model	Avg Memory Used(MB)	Avg Inference Time(ms)	Total Training Time(s)
nGPT	187	273	4296
PEGASUS	2100	353	19936
BART	1750	303	16967.5

Figure 4: Memory, inference and training times

Figure 5 displays comparative analysis of normalized inference times per sample for nGPT, Pegasus, and BART across different training epochs. It is evident that nGPT consistently achieves lower inference times compared to Pegasus and BART, indicating its efficiency in model deployment.

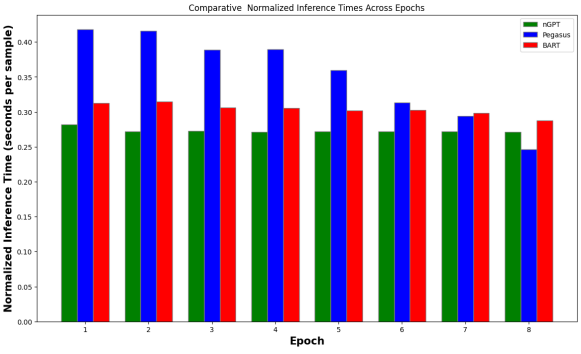

Figure 5: Comparative Normalized Inference Times Across Epochs

Unexpected Results and Observations

The nGPT model exceeded our expectations (we didn't think it would) in efficiency metrics such as memory usage and inference speed, which validated its design for resource efficiency. While its ROUGE scores were the highest, the difference was modest, suggesting potential trade-offs between efficiency and performance in summarization tasks. With fairly low training time, the model was able to produce meaningful English sentences with a low accuracy grammar.

7 Conclusions

Our project comparing nGPT, BART, and PEGASUS provided a comprehensive learning experience, from building models from scratch to utilizing tools like Hugging Face's transformers. We gained insights into key metrics such as inference times, memory usage, and ROUGE scores, understanding how to interpret and optimize them. The study also highlighted the benefits of training models on GPUs and the advantages of using fp16 precision.

We confirmed Nvidia's claim that nGPT trains faster and stabilizes sooner, achieving speed improvements of at least 4x, aligning with Nvidia's reported range of 4-20x. This allowed us to delve deeper into the inner workings of BART and PEGASUS, observing their summaries at various development stages. The bigger and scaled versions of the model produce better summaries in case of BART and PEGASUS, we hope to see better summaries with nGPT too someday. Some summaries are part of our appendix and we decided it was not worth it as a report since it was too early and gibberish. Appendix A

For future work, we aim to fully train nGPT on the entire CNN DailyMail dataset, expecting improved ROUGE scores and enhanced model stability. Additionally, we plan to complete our nGPT implementation and compare its correctness with Nvidia's official release.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv*.

Mike Lewis, Yinhan Liu, Naman Goyal, and et al. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *ACL*.

Ilya Loshchilov, Cheng-Ping Hsieh, Simeng Sun, and Boris Ginsburg. 2024. ngpt: Normalized transformer with representation learning on the hypersphere. *arXiv*.

Yi Tay, Mostafa Dehghani, Alexander Vasilenko, et al. 2020a. Efficient transformers: A survey. *arXiv*.

Yi Tay, Mostafa Dehghani, Ashish Vaswani, et al. 2020b. Efficient attention: A survey. *arXiv*.

Jingqing Zhang, Mostafa Dehghani, Yao Wang, and et al. 2020a. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv*.

Wei Zhang, Chang Zhang, Xiaohui Liu, and et al. 2020b. Transformers for text summarization: A comprehensive review. *arXiv*.

APPENDIX

Generated Summaries

A 15K SAMPLES, 100M PARAMETERS

nGPT: Hours the preceding locations along with measurements and the intracrants thereof can tend to be less decreased in perfection, perfection and perfect

BART: Discovery takes place on the summer's leg leg leg with a leg leg . Says it can help help help get back to get away and get back "We're in your way to make you you you like you you," she says .At least one of the world's best way to be able to get back back

PEGASUS: CNN's new album is a new album of the best in the world . He says you can be able to be a good for you't be able . She says you know you can't know what you know what it't make it

B Fine-tuned BART

Lazing in a hammock is one of the best ways to spend a summer evening. The best way to catch fireflies is with womanly wiles. Running on the beach can get you into great shape.

C Fine-tuned PEGASUS

Real Simple offers five great ways to enjoy your summer .<n>Best way to catch fireflies – How? With womanly wiles .<n>The best way to get in and out of a hammock – Everyone looks good in a hammock .<n>The best way to tie espadrilles – Apply this lace-up logic .

D Code

GitHub