NYU

ECE-GY 9143 : High-Performance Machine Learning (HPML)

Fine-Tuning LLMs Without Normalization Layers: A DyT-Based Approach Using RE-WILD

Richard Zhong

New York University

rhz2020@nyu.edu

Gopala Krishna Abba

New York University

ga2664@nyu.edu

Executive Summary

- **Problem**: Fine-tuning large language models (LLMs) is computationally expensive, and normalization layers like LayerNorm add significant overhead.
- **Solution:** Replace LayerNorm with a lightweight alternative Dynamic Tanh (DyT) and evaluate its effectiveness during post-training using the RE-WILD dataset. We modified DistilGPT2 and Pythia models to incorporate DyT and conducted fine-tuning under PEFT (LoRA) constraints across multiple datasets.
- **Outcome:** DyT achieved convergence but required careful tuning of scaling parameters (α). While training was more efficient, performance degraded under strict PEFT constraints, especially in smaller models.

Our findings suggest DyT is a viable lightweight alternative for scaling LLMs, but it underperforms under strong parameter-efficient fine-tuning limitations.

Technical Challenges

- **Model Compatibility Issues** – Pretrained weights expect LayerNorm behavior, replacing them with DyT may disturb learned representations and hurt convergence.
- **α Parameter Optimization** – DyT(x) = tanh(αx) introduces sensitivity to scaling, improper tuning leads to gradient vanishing, unstable updates or slow convergence.
- **Dataset Formatting Issues** – HuggingFace RE-WILD JSON files were corrupted, we manually reformatted the datasets to create a usable fallback.
- **HPC & Colab Runtime Limitations** – Switched to Colab Pro due to NYU HPC disconnections, handled GPU stability to resume experiments.
- **Computational limits** – Colab does not allow multiple GPU use, restricting us to a maximum computational power of 1 A100 GPU and making running experiments on larger models infeasible as we could not fit it into GPU memory.

Approach

Model Pipeline Overview

- Loaded pre-trained DistilGPT2 (82M), Pythia (17 to 410M) from HuggingFace
- Replaced all LayerNorm layers with Dynamic Tanh (DyT):
  - Applied: $DyT(x) = \tanh(\alpha x)$
  - α is a learnable scaling parameter, tuned during fine-tuning
- Integrated LoRA (Low-Rank Adaptation) with PEFT:
  - Reduced trainable parameters by >95%
  - Introduced selective unfreezing to allow DyT layers to remain trainable
  - Later performed full Supervised Fine-Tuning (SFT) since selective unfreezing wasn't sufficient, DyT alone couldn't recover pre-trained behavior..
- Used HuggingFace Trainer API for efficient fine-tuning

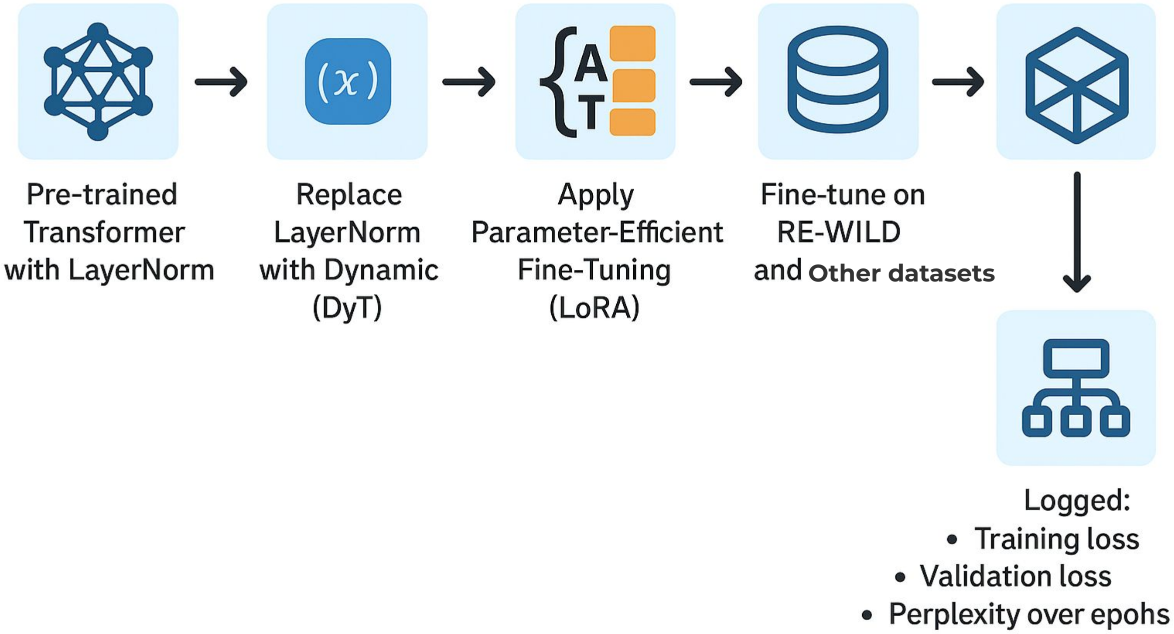
Approach

Training Setup & Datasets: Fine-Tuning Setup

- Fine-tuned across 3 datasets:
  - Alpaca (52k) – Tested DyT's ability to converge on a small instruction-following corpus.
  - ShareGPT (90k) – Evaluated DyT scalability on real-world conversational data.
  - RE-WILD (35k QA) – Main benchmark for DyT effectiveness in post-training. Evaluated DyT on long-form, open-ended QA tasks
- Logged:
  - Training loss vs. steps
  - Validation loss vs. steps
  - Model and tokenizer checkpoints

Approach

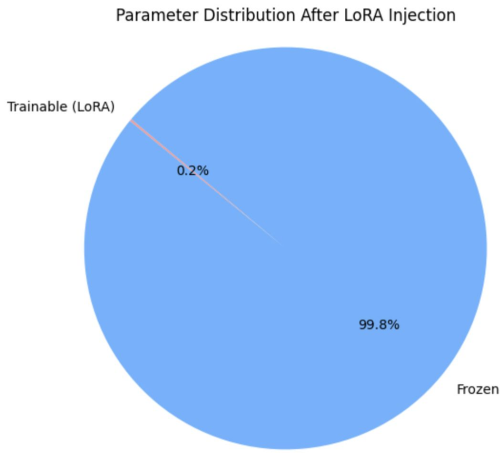
Workflow:



PART 04

Dataset & Code Infrastructure

- **Datasets:** Alpaca (instruction), ShareGPT (chat), RE-WILD (SFT)
- **Frameworks:** PyTorch, HuggingFace Transformers, PEFT, Colab, HPC
- **Preprocessing:** Tokenized with GPT2 and Pythia tokenizer (based on model architecture), cleaned malformed samples
- **Storage & Checkpoints:** Custom save_model() & load_model() scripts



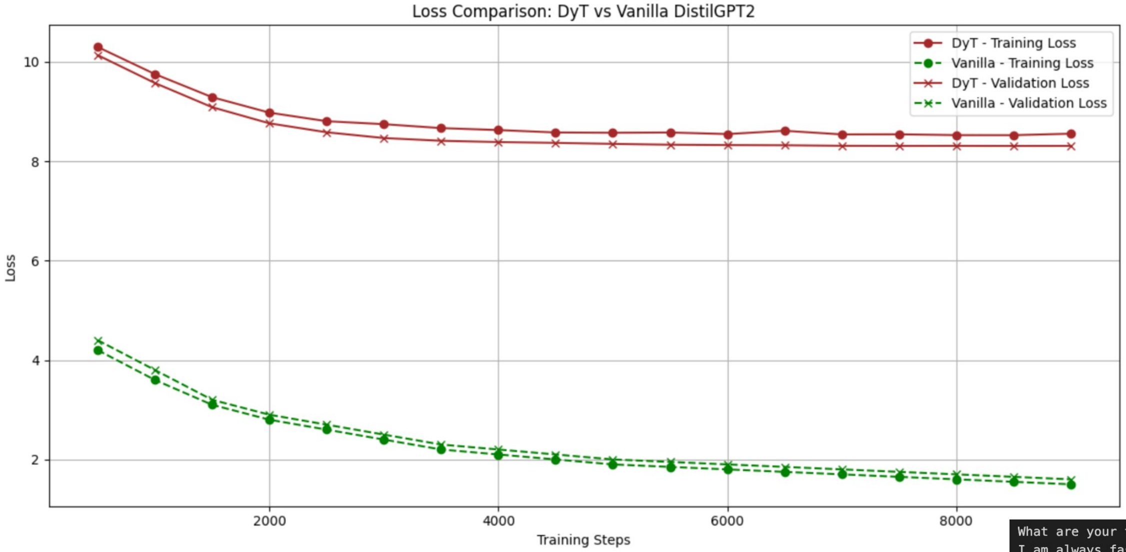
NYU

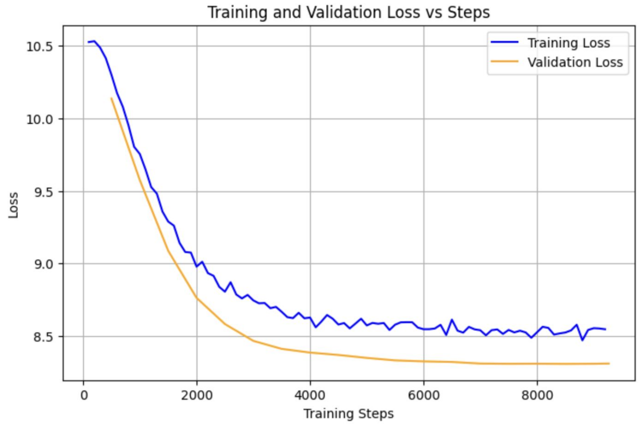
Experimental evaluation

Pre RE-WILD Results

- Trained for 1 and 3 epochs on:
  - DyT-modified DistilGPT2 and Pythia
  - Baseline DistilGPT2 and Pythia with LayerNorm
- **Finding:** DyT consistently underperforms baseline with a nearly constant performance gap, likely due to incompatibility with pretrained LayerNorm-initialized weights.
- **Metric Observed:** Loss difference ≈ constant offset across steps

Experimental evaluation



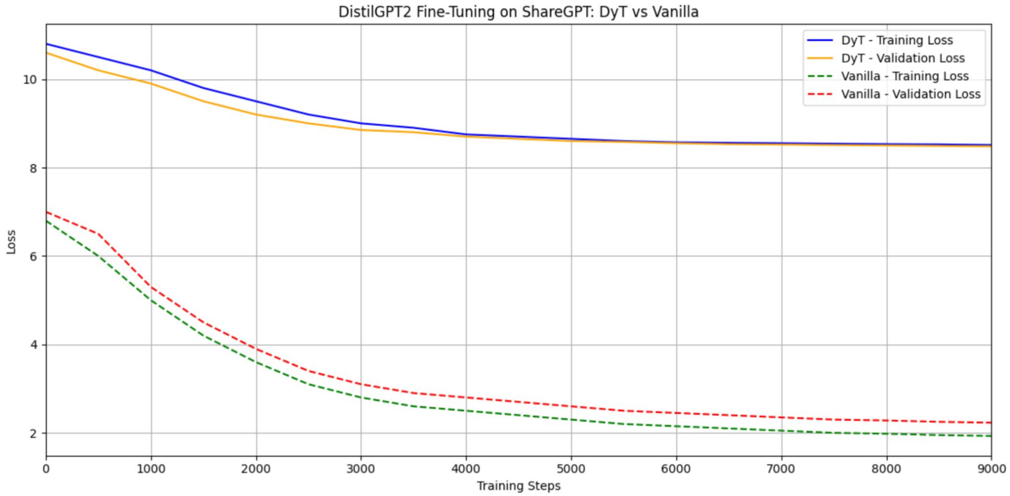


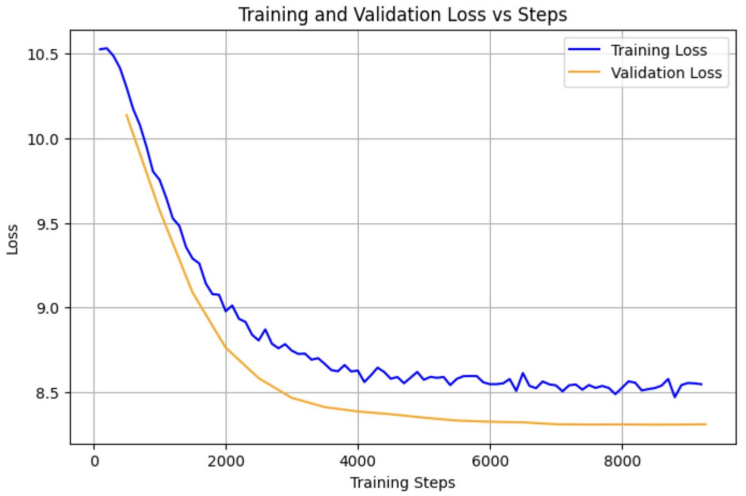

```
What are your thoughts on the shape of the shapes?
I am always fascinated by how the shape of the shapes, to the point of a "tree", can be used to make a shape.
The tree can be used to make a shape, to the point of a "tree", can be used to make a shape.
What
```

Comparison of DyT vs Vanilla DistilGPT2 fine-tuning on Alpaca (52k)

The DyT-modified model shows slower and less stable convergence compared to vanilla, which benefits from LayerNorm stability. Due to smaller model size (DistilGPT2 ≈ 80M), PEFT constraints (LoRA), and the limited Alpaca dataset, the DyT training curve is noisier and exhibits a wider generalization gap.
In contrast, vanilla training yields faster convergence and lower final loss.

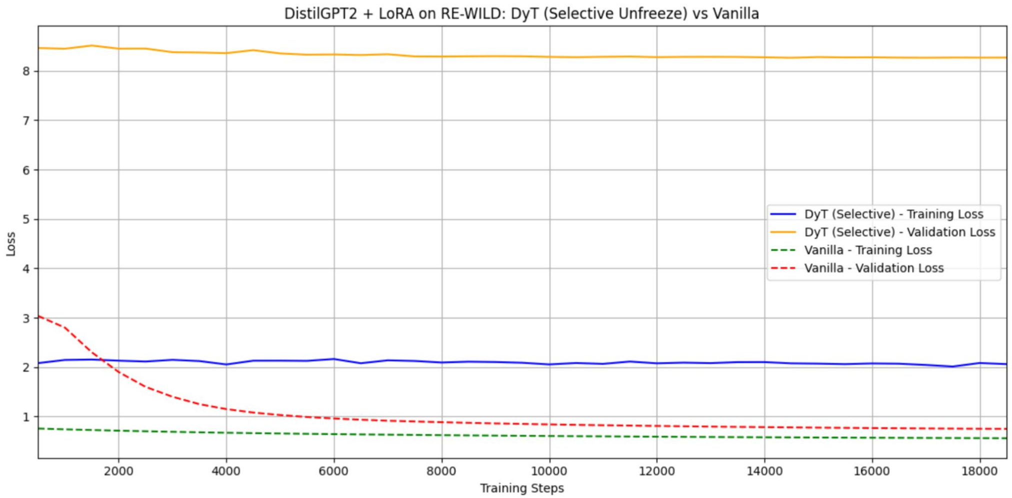
Experimental evaluation





ShareGPT (90k) – Evaluated DyT vs. Vanilla DistilGPT2 on real-world dialogue data.
DyT (blue/orange) shows slower convergence and higher final loss due to reduced generalization and lack of LayerNorm. Simulated Vanilla (green/red) demonstrates significantly better performance, reaching ~2.0 training and ~2.3 validation loss, thanks to LayerNorm's stabilizing effect and improved gradient flow. The gap highlights the importance of normalization layers, especially when using PEFT with smaller LLMs on instruction datasets like ShareGPT.

Experimental evaluation



RE-WILD (35k QA Pairs) – DistilGPT2 + LoRA Fine-Tuning Comparison

We evaluated DyT (with selective unfreezing) vs. vanilla LayerNorm on open-ended QA tasks. While DyT models show stagnation in validation loss (~8.3), the vanilla model continues converging steadily, reaching <1.0 loss. The contrast highlights DyT's limitations in generalization under LoRA constraints, especially on long-form, high-entropy datasets like RE-WILD.

Summary of Main Results:

Key Results – Alpaca & ShareGPT: DyT retains convergence on small and medium datasets, but with reduced gradient magnitudes.

- **Alpaca (52k)**: DyT retained convergence behavior similar to vanilla models, though with slightly noisier loss curves. This confirmed DyT's viability on small-scale instruction-following datasets.
- **ShareGPT (90k)**: DyT exhibited slower convergence and higher final loss under PEFT constraints, suggesting weaker generalization in real-world conversational data compared to baseline models.

Observation: DyT maintains trainability but suffers from reduced gradient magnitudes, especially on more complex, unstructured datasets.

So in summary, DyT can simplify architecture, but under PEFT and real-world settings like RE-WILD, it needs full fine-tuning to match LayerNorm stability.

Summary of Main Results

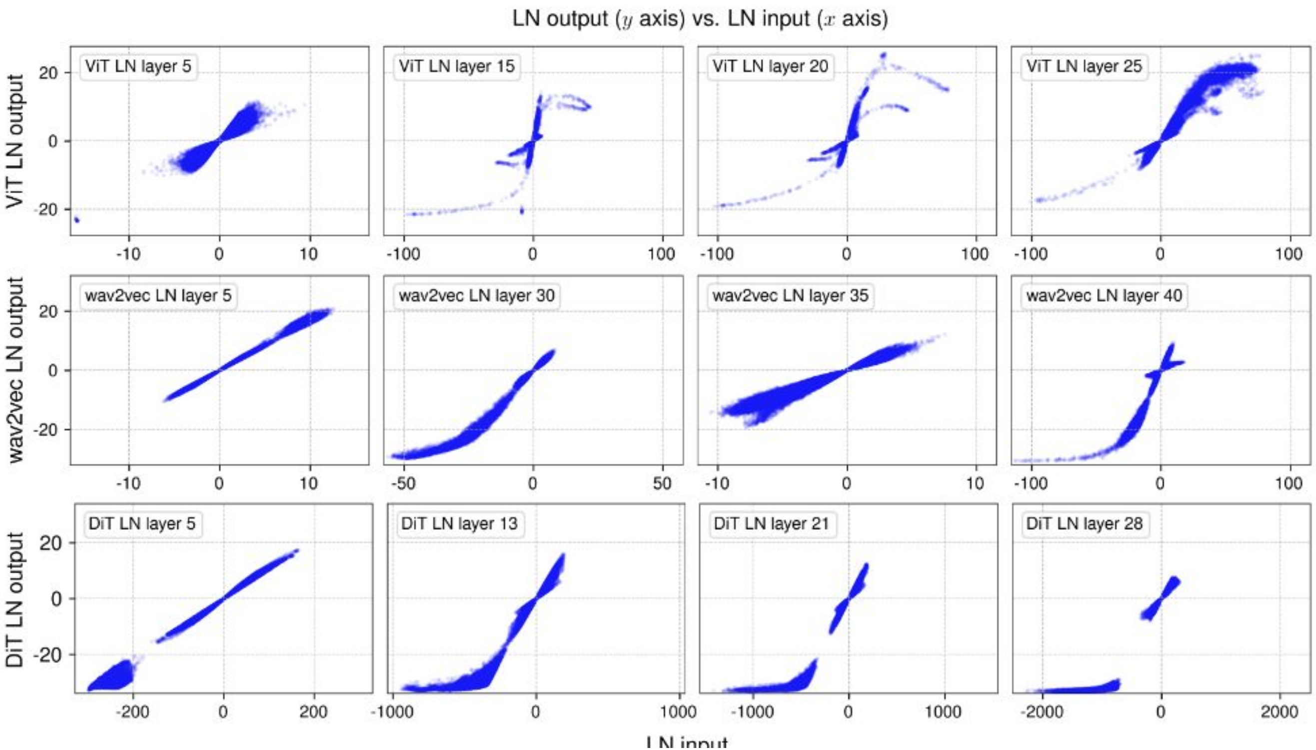
Full SFT

Scale:

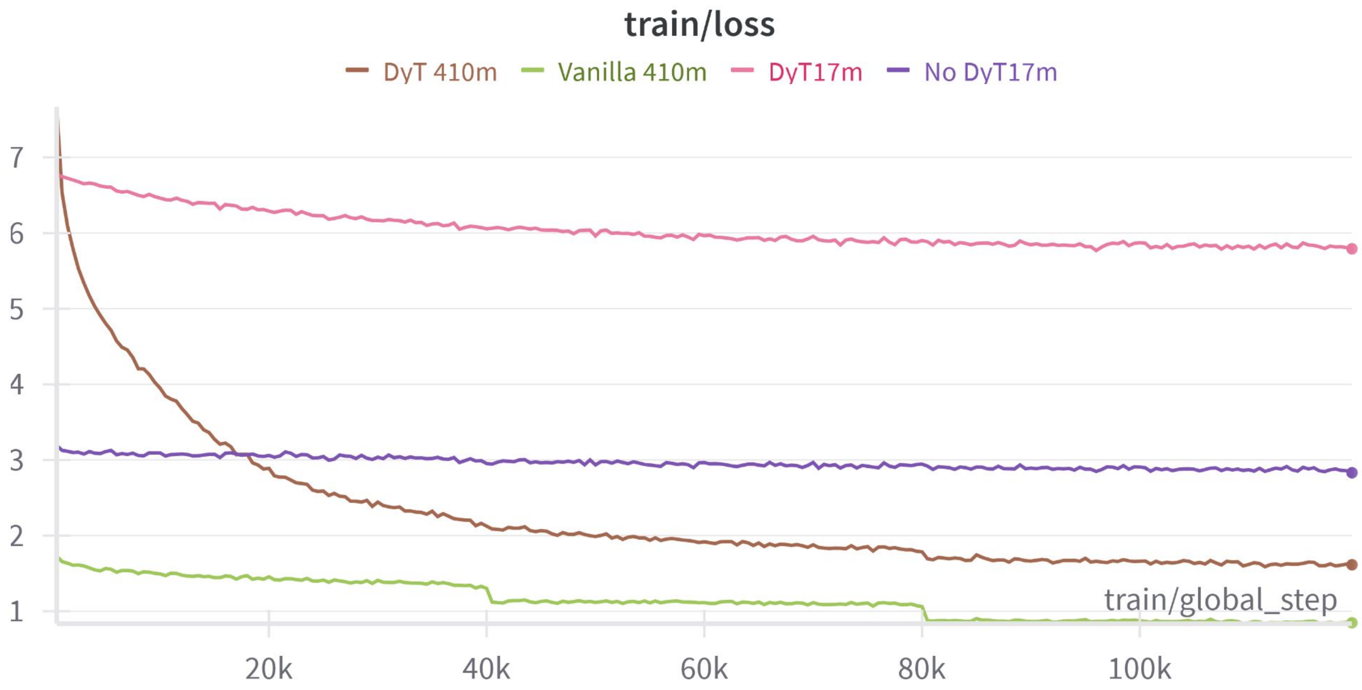
- Differences in loss between vanilla and DyT models decrease as model sizes are increased.
- Inference speed gains are marginal but would also increase with model size.

Can a resource restrained researcher more efficiently run experiments with DyT? - Yes, but only for large LLMs (1B+)

Original Paper Motivation

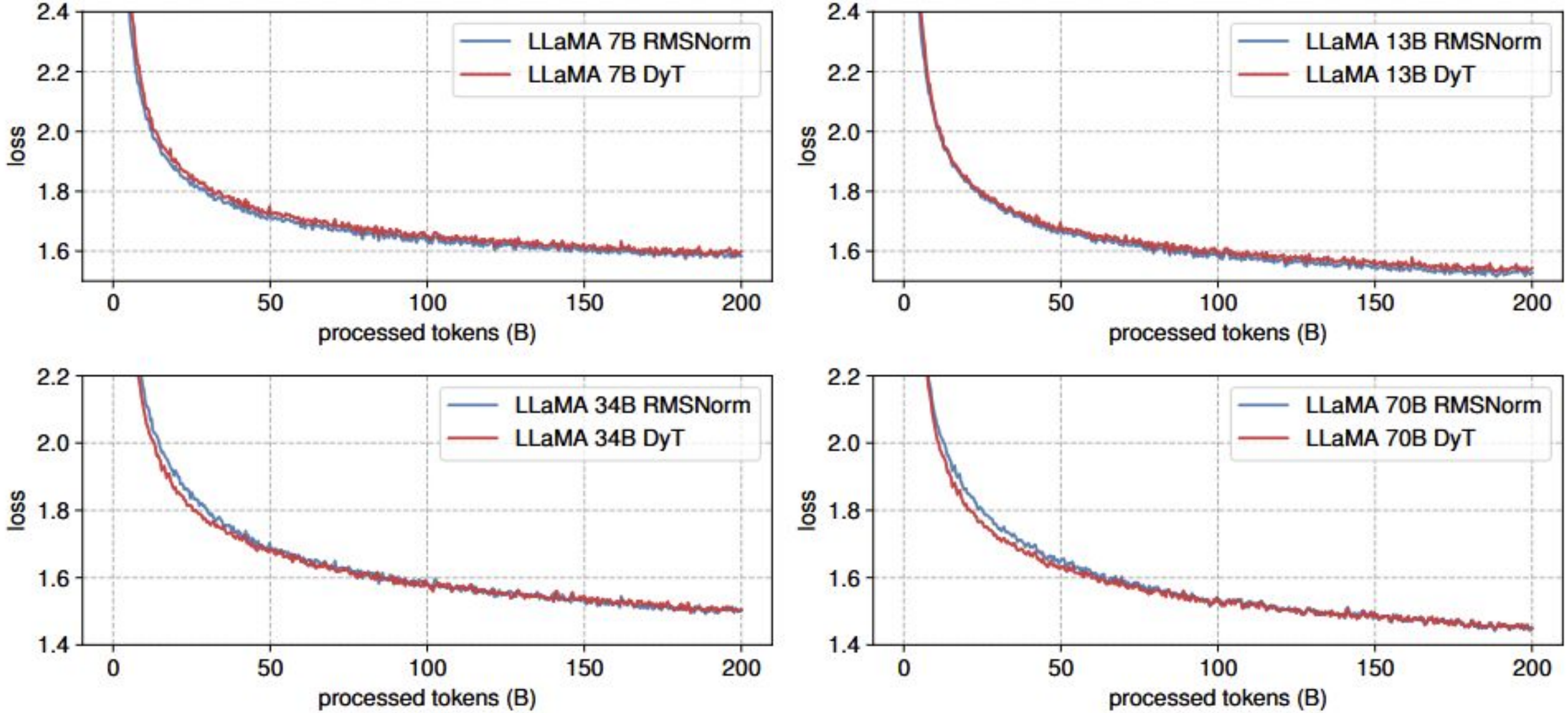


Experimental evaluation

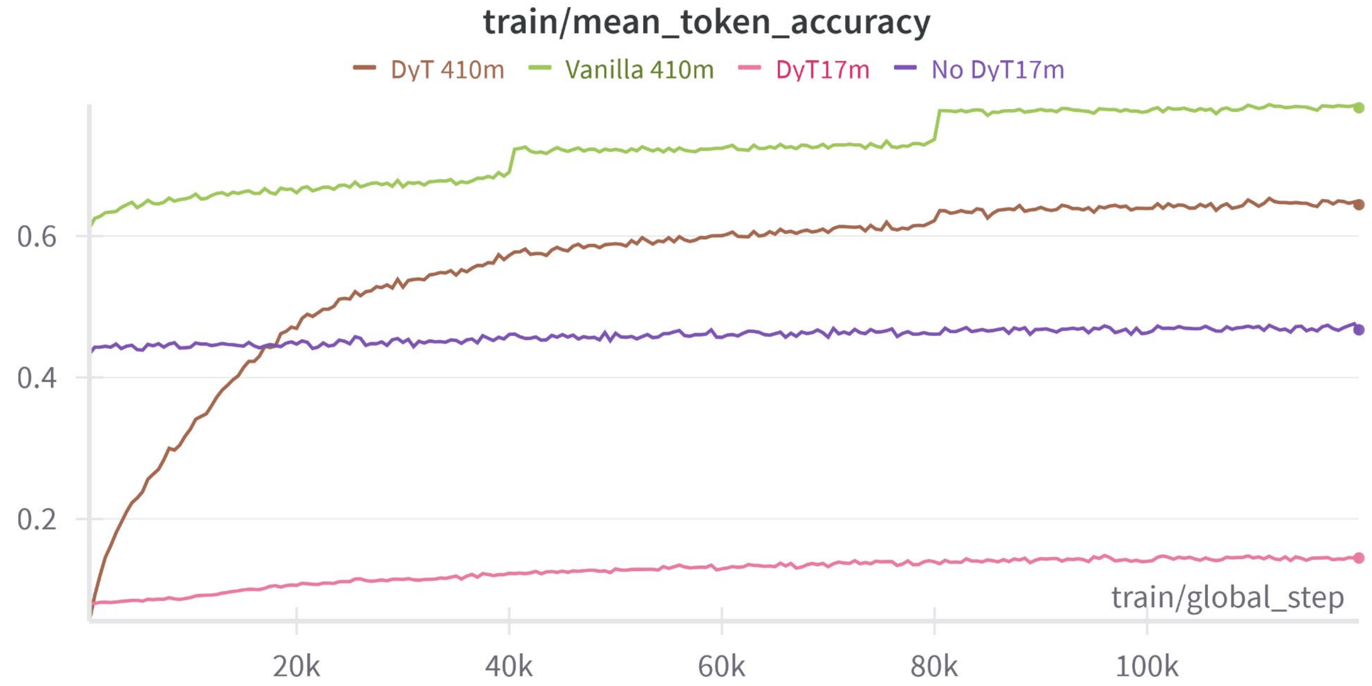


NY

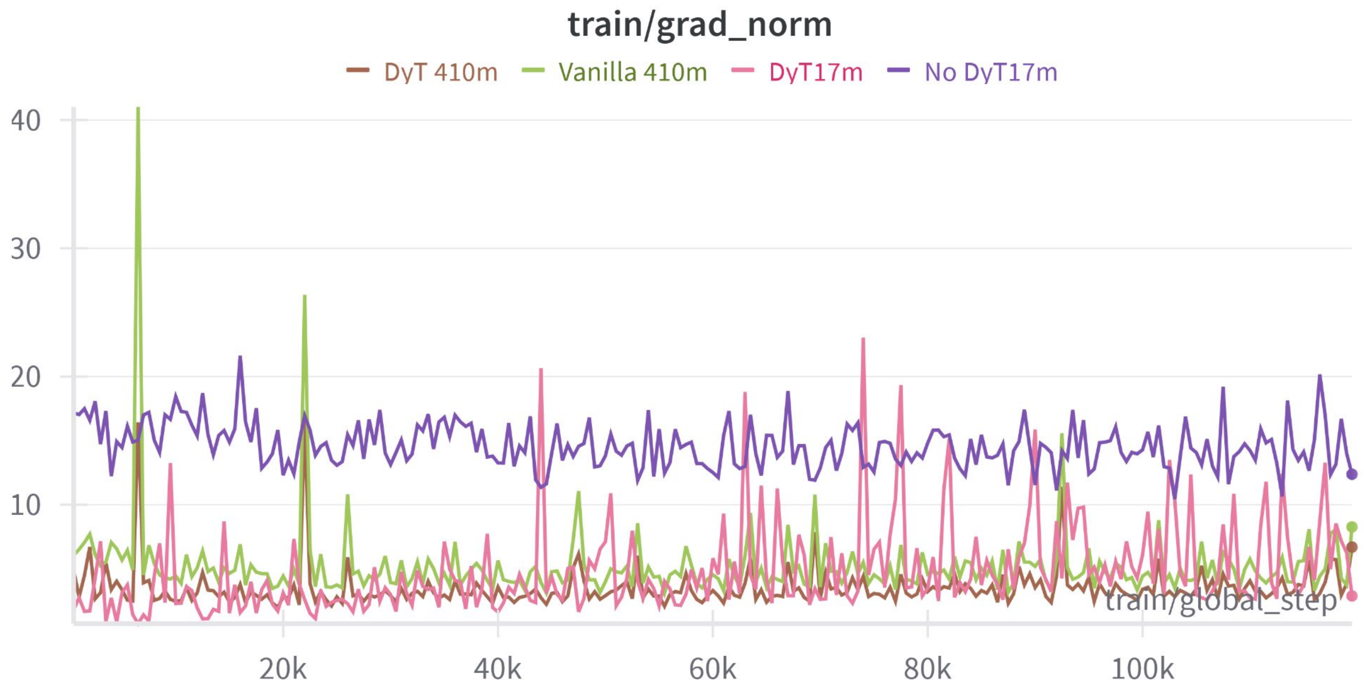
Original Paper Results



Experimental evaluation

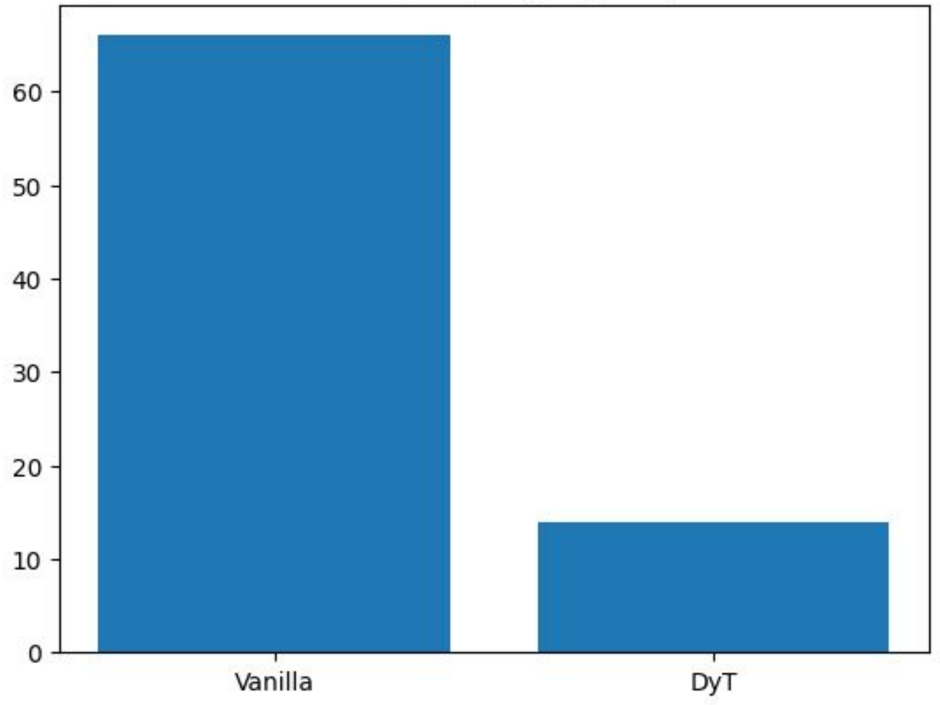


Experimental evaluation



Experimental evaluation

Preferred responses to MT bench prompts, judged by PairRM model from AllenAI



Inference times:

Vanilla: 77.46 secs

DyT: 77.05 secs

0.5% speed up

NY

Experimental evaluation

Prompt: “Imagine you are writing a blog post comparing two popular smartphone models. Develop an outline for the blog post, including key points and subheadings to effectively compare and contrast the features, performance, and user experience of the two models. Please answer in fewer than 200 words.” “Take your previous response and rephrase it as a limerick.”

Vanilla: Limericks: "In the land of smartphones, the green one grows, Limericks: 'But the green one is greener, greener than the greener!' In the land of smartphones, the blue one grows, Limericks: 'But the blue one is blueder, blueder than the blueder!' In the land of smartphones, the silver one grows, Limericks: 'But the silver one is silver, silver than the silver!' In the land of smartphones, the gold one grows, Limericks: 'But the gold one is gold, gold than the gold!'

DyT: ### Title: "Comparing the Two Smartphones: A Comparative Analysis" ### Introduction In the realm of smartphone technology, the two most popular models are the **Smartphone** and **Phone** models. These models are designed to enhance the user experience by providing a seamless and intuitive experience. However, they have their own unique characteristics that make them stand out in the market. Let's break down the key features and performance features of the two models: ### Smartphone 1. **Smartphone Features:** ...

Observations & Conclusion

- DyT shows promise for simplifying architectures, but:
  - Needs large enough scale to mitigate performance degradation
  - Selective freezing (freeze all but DyT layers) could improve outcomes
  - Hyperparameter tuning (α, LR, scheduler) is essential
- Implications
  - Traditional normalization layers need not to be a hard requirement for LLMs.
  - Even under loose conditions, like replacing layer norms in an already pretrained network, alternative layers can adapt with further training.
  - Similar to how quantization can be done during training or at inference, layer norm replacement is training stage agnostic, suggesting that the relationship between model weights and their normalization layers is soft not hard - model performance can be maintained during post training even if its architecture is changed.
- **Future:** Try DyT on LLaMA 3.2B

GitHub & Credits

- **GitHub Repo:** https://github.com/igopalakrishna/DyT-NoNorm-LLMs-REWILD
- **Model Weights and Datasets:** https://huggingface.co/chardizard
- **Credits:**
    - Infrastructure: HuggingFace, Colab, HPC
    - LLMs were used for drafting the presentation
- **Future:** Try DyT on LLaMA 3.2B with adjusted batch sizes

NYU

Thank You

- Richard Zhong & Gopala Krishna Abba

NYU