FNet + BART – Is **Fourier Transform** all you need ?

Intro to Natural Language Processing (Spring 2025)

September 30, 2025

Final Report

NYPD

Aniruth Suresh (2022102055)
Aryan Garg (2022102074)
Samkit Jain (2022102062)

Github Link

1 Introduction

As a team of three undergraduate students in Electrical and Computer Engineering, our natural interest was inclined toward Fourier Transform . This project explores its powerful role in NLP by investigating whether Fourier-based transformations can replace the self-attention mechanism in Transformer models. While attention operations scale quadratically with input length $\mathcal{O}(n^2)$, the Fourier Transform offers a significantly more efficient alternative with a computational complexity of $\mathcal{O}(n \log n)$. Inspired by recent works such as FNet [1], which replace attention in the encoder with a Fourier layer, we began by implementing FNet from scratch and adapting the idea to the BERT architecture. However, our core contribution is novel: we extend this **Fourier-based replacement to the BART architecture, which uses both encoder and decoder attention**. We experiment with **replacing attention in the encoder only, decoder only, and both**, and test these variants across multiple multimodal datasets (image , audio and multiple text datasets). Finally, we evaluate the performance of these models on the **JEDI [2] task, which involves generating dialogues based on context**. Our results indicate that the Fourier Transform can serve as an efficient and viable alternative to attention in certain crucial tasks.

2 Project Phases

We divided the entire pipeline of the project into three phases:

- **Phase 1:** Implementing FNet from scratch and benchmarking it, as well as testing FFTNet [3], a recent and interesting paper that builds on similar ideas.
- **Phase 2:** Incorporating the Fourier Transform into the encoder, decoder, and both components of the BART architecture, and testing these variants on multiple datasets.
- **Phase 3:** We first conduct experiments to analyze the contribution of the complex-valued component in the Fourier Transform . After this ablation, we integrate our best-performing models and evaluate them on the JEDI task.

3 Phase 1: FNet Benchmarking (BERT) + FFTNet

3.1 FNet Benchmarking

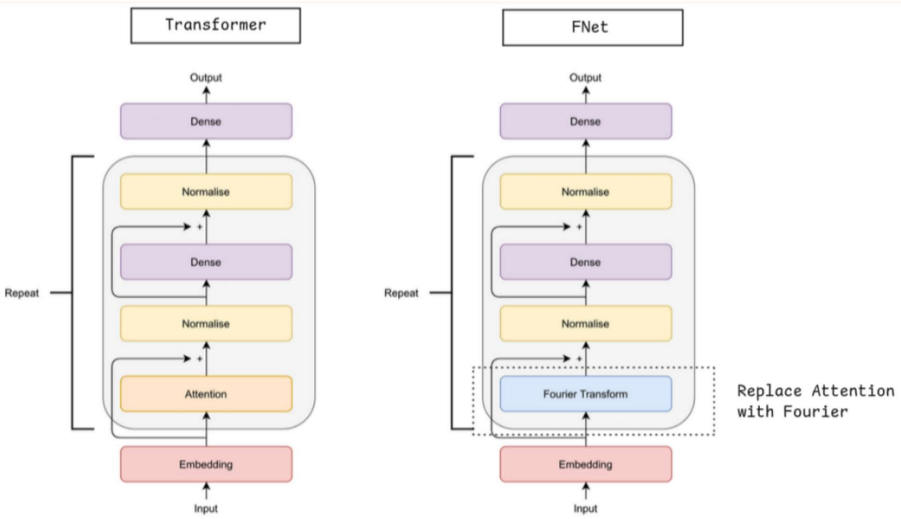

Figure 1: FNet Architecture: Replacing self-attention with Fourier Transform.

FNet replaces the computationally heavy self-attention mechanism in the standard Transformer model with a more efficient operation based on the Discrete Fourier Transform (DFT). In traditional self-attention, the complexity of computing pairwise interactions between all tokens scales quadratically with

the input length ($\mathcal{O}(n^2)$). In contrast, FNet uses the Fourier Transform to mix the input information in a way that reduces the computational complexity to $\mathcal{O}(n \log n)$, which is significantly more efficient, especially for long sequences.

The core idea behind this is that the Fourier Transform (FT) provides a frequency-based representation of the input sequence, allowing it to **capture global dependencies efficiently without explicitly computing all pairwise interactions**. Mathematically, the Fourier Transform of a signal $x(t)$ is given by:

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-2\pi i f t} dt$$

The intuition here is that by shifting the information from the time domain to the frequency domain, we can efficiently capture global patterns without directly attending to every token pair. This approach works because the Fourier Transform captures the essential frequency components of the sequence, which are often sufficient to model long-range dependencies in many NLP tasks.

3.1.1 SWAG Benchmarking

In our first attempt, we focused on replacing the attention mechanism with the Fourier Transform, as mentioned in the previous section, and benchmarked it against the *SWAG dataset* [4]. We ran three models for comparison:

- **BART with Standard Attention:** The original BART model with the traditional self-attention mechanism.
- **BERT + FNet:** A model where the attention mechanism in BERT is replaced by the Fourier Transform (FT), following the approach of FNet.
- **BERT + Random Encoder:** A model where the attention mechanism is replaced by a **random encoder**. This model serves as a baseline to show that the Fourier Transform is not just a random substitution, but rather an effective transformation that contributes positively to the model's performance.

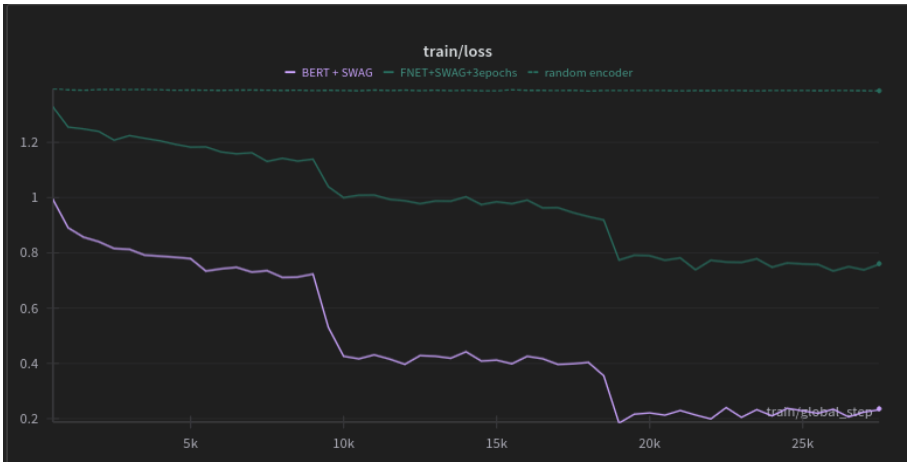

Figure 2: Training loss for all three models on the SWAG dataset.

As observed in Figure 2, the model with the random encoder does not learn effectively, as indicated by its almost constant training loss. This emphasizes that **replacing the attention mechanism with a random encoder is not a viable approach**. On the other hand, the Fourier-based FNet model, while having a slightly higher loss compared to the standard attention model, demonstrates much faster training times. Specifically, the runtime for the FNet model was *46 minutes and 11 seconds*, whereas the standard BART model took *2 hours, 4 minutes, and 15 seconds*. This shows a clear trade-off: while FNet incurs a small increase in loss, it offers significant improvements in computational efficiency and this is because **FT has no learnable parameters** .

Context:
The weather was getting colder and the leaves were falling from the trees.

Choices:
1. She decided to wear a light summer dress.
2. He put on a heavy winter coat.
3. They went to the beach to enjoy the sun.
4. The sun was shining brightly in the sky.

Figure 3: Context and Choices for the Model's Prediction.

The model predicts: "He put on a heavy winter coat" with an accuracy of 0.7800 and an F1 score of 0.7799, as shown below.

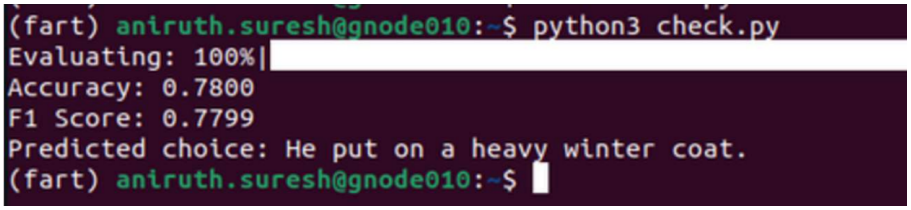

Figure 4: Prediction of the finetuned FNet model

The results highlight that the model can achieve good performance and efficiently predict in much less runtime when the attention mechanism is replaced by the Fourier Transform.

3.2 FFTNet

FFTNet [3] is a recent paper that builds on similar ideas, proposing an efficient method to replace attention with Fourier-based operations. It makes use of an an adaptive spectral filtering framework designed to address the quadratic complexity of traditional self-attention mechanisms in Transformer.I n the frequency domain, **orthogonal frequency components inherently encode long-range dependencies**, and **Parseval's theorem** ensures the preservation of signal energy, contributing to stable representations.

A key innovation is the introduction of a learnable spectral filter and a **modReLU** activation function, which dynamically emphasize salient frequency components based on a global context vector, providing an adaptive alternative to fixed spectral transforms. The FFTNet method involves 4 main steps:

1. **Fourier Transform**

   First the Discrete Fourier Transform is applied along the token dimension:

   $$F = \mathrm{FFT}(X) \in \mathbb{C}^{n \times d}$$

   This operation represents each embedding across orthogonal frequency components, enabling global interactions without explicit pairwise comparisons.

2. **Adaptive Spectral Filtering**

   To selectively emphasize important frequencies, we use a learnable filter. First, compute a global context vector:

   $$c = \frac{1}{n} \sum_{i=1}^{n} X_i$$

   and pass it through a multi-layer perceptron (MLP) to obtain a modulation tensor:

   $$\Delta W = \mathrm{MLP}(c) \in \mathbb{R}^{n \times d}$$

   The final filter can be written as:

$$W = W_{\text{base}} + \Delta W$$

where W_{base} is a fixed base filter (often initialized to all ones). The adaptive filtering step is then

$$\tilde{F} = F \odot W$$

which reweights the Fourier coefficients element-wise according to the global context.

3. **modReLU**

The apply non-linearity in the complex domain of the frequencies, modReLU activation is used which is defined for a complex number $z = re^{i\theta}$ as:

$$\text{modReLU}(z) = \begin{cases} (r+b)\, e^{i\theta}, & \text{if } r + b > 0, \\ 0, & \text{otherwise,} \end{cases}$$

where b is a learnable bias. So,

$$\tilde{F} = \text{modReLU}(\tilde{F})$$

3. **Inverse FFT**

Finally, return to the token domain via the inverse Fourier transform, retaining the real component of the reconstructed signal:

$$Y = \text{IFFT}(\tilde{F}) \in \mathbb{R}^{n \times d}$$

This yields a globally mixed representation that incorporates adaptive filtering and nonlinear transformations in the frequency domain.

3.2.1 Benchmarking on CIFAR - 10

We benchmarked the above-mentioned FFTNet model against ViT and FNet-Base (which replaces encoder attention in BERT with a Fourier Transform). The figure summarizes the key trends. Observe how ViT takes 32 seconds per epoch while FFTNet only takes 14 seconds, achieving nearly similar performance with just a ∼1% drop — highlighting the significant speed advantage of Fourier-based models with minimal loss in accuracy.

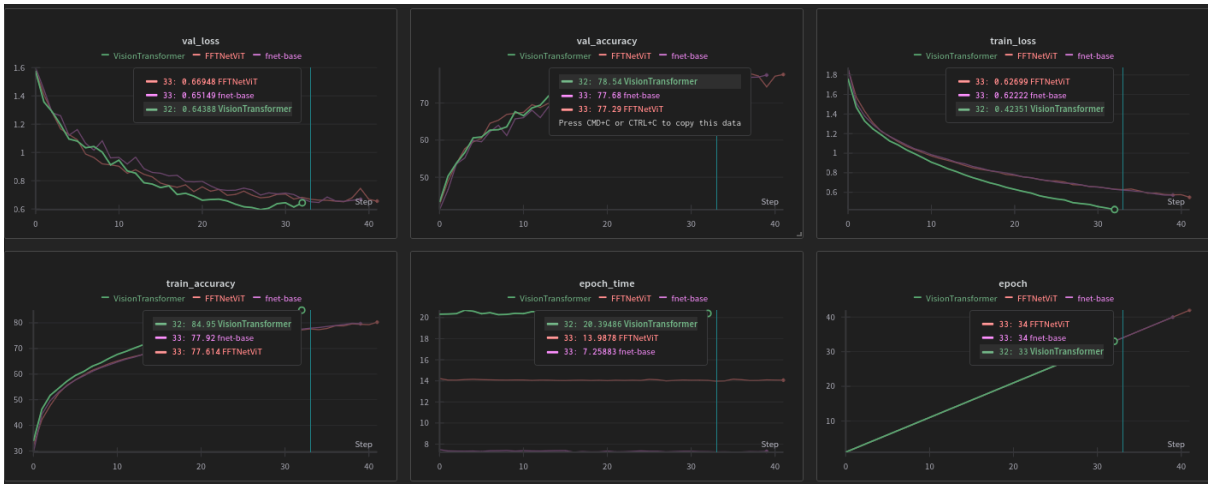

Figure 5: Loss logs on both training and validation data

Performance Order: ViT > FFTNet > FNet-Base
Training Time per Epoch: ViT: **32s** > FFTNet: **14s** > FNet-Base: **7s**

Figure 6: Model comparison in terms of performance and training efficiency.

4 Phase 2: Fourier Transform on BART — Encoder, Decoder, and Both

4.1 Encoder Only

In this setup, we replace the self-attention mechanism in the encoder of the BART model with a Fourier Transform, while keeping the **decoder architecture unchanged**. The goal is to evaluate how much the encoder's attention mechanism contributes to the model's performance and whether it can be effectively substituted by the Fourier Transform. The simplified architecture is shown below:

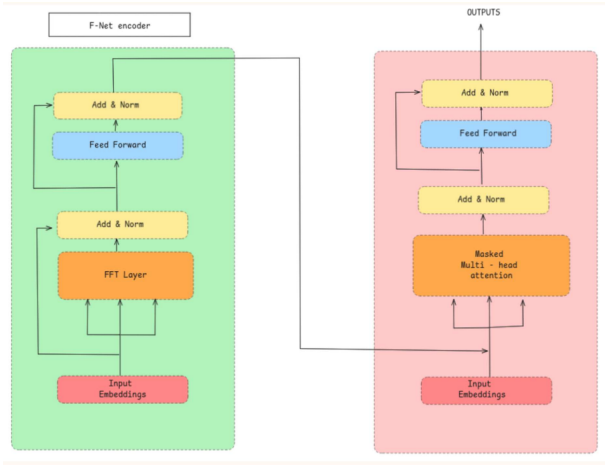

Figure 7: BART with Fourier Transform replacing encoder attention

We benchmarked this encoder-modified BART model on two datasets: SST-2 and SWAG.

4.1.1 Benchmarking on SST-2

SST-2 is a sentiment analysis dataset where the task is essentially binary classification. We fine-tuned both the base BART and the modified BART with Fourier-based encoder (BART+FNet-Encoder). The training loss curve and evaluation metrics are summarized below.

Model	Eval Accuracy (%)	Eval Loss	Training Time
Base BART	92.23	0.401	22m 52s
BART + FNet Encoder	83.37	0.559	13m 33s

Table 1: Performance comparison on SST-2 dataset

Despite a drop in accuracy, the Fourier-based encoder model significantly reduces training time, indicating a tradeoff between performance and computational efficiency. This suggests that even partial replacement of attention mechanisms with Fourier operations can yield time gains, albeit at some cost to accuracy.

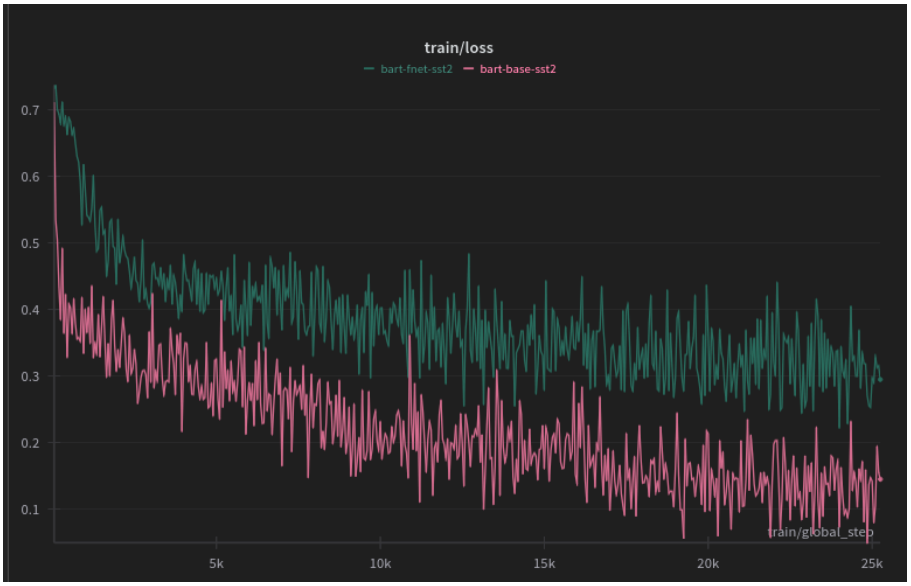

Figure 8: Training logs where pink corresponds to the base BART and green corresponds to our model with Fourier-based encoder.

4.1.2 Benchmarking on SWAG

Extending the same type of analysis to a harder dataset proved to be challenging. SWAG is a more complex dataset compared to SST-2, and hence the model did not perform as expected. In fact, it suggests that further adaptations might be needed, as shown in the training loss graph below.

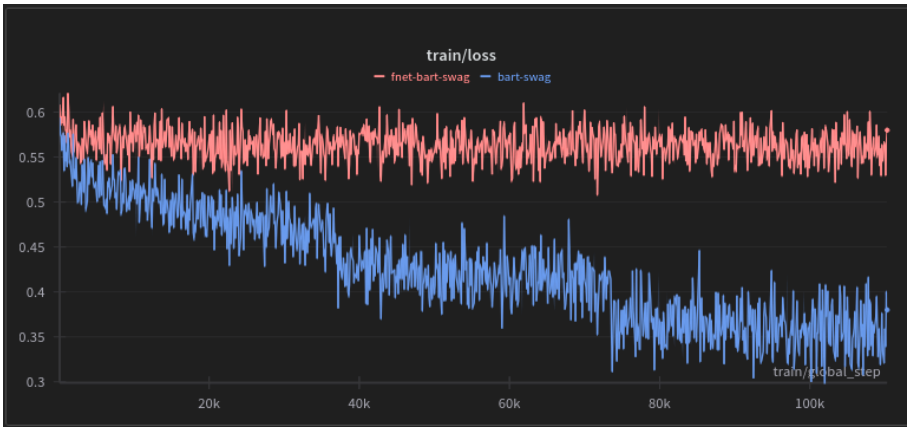

Figure 9: Training logs where blue corresponds to the base BART and red corresponds to our model with Fourier-based encoder.

The reason for this is that SWAG **heavily depends on understanding context and making inferences**, which often benefits from the attention mechanism. By replacing attention with the Fourier Transform, the model may be **losing important semantic information** crucial for such tasks.

4.2 Decoder Only

In this case, we replace both the self-attention and cross-attention mechanisms in the decoder. This setup is **more complex** than the encoder-only modification because we have cross-attention, which can be formed from **different modalities, and there is no natural ordering relationship between the encoder and decoder tokens**.

Our first attempt (on SST-2 dataset) to directly replace the attention mechanism with the Fourier Transform and apply a simple learning rate schedule failed miserably, as shown below.

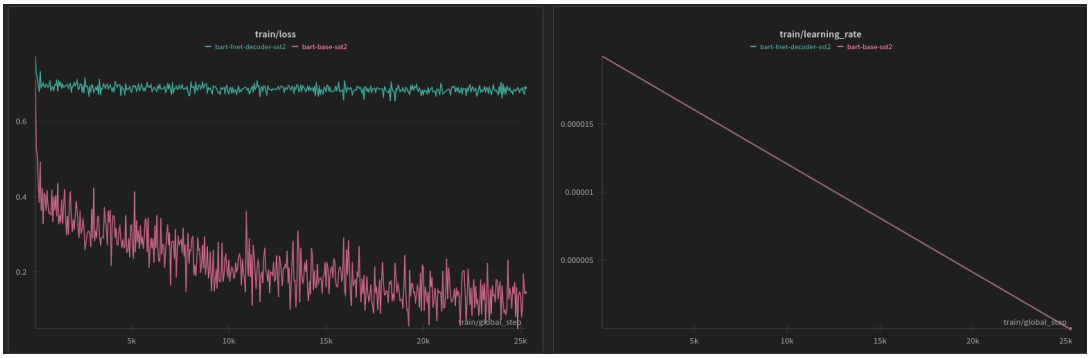

Figure 10: Training loss with decoder-only Fourier Transform replacement (failed attempt).

Upon debugging, we realized that the gradient was approaching zero during the first epoch itself, leading to a **vanishing gradient problem**. To overcome this, we adopted two major approaches:

1. Add **residual connections** at multiple points within the decoder architecture.
2. Implement a **learning rate scheduler**.

We also structured the training process in two stages:

- **Stage 1:** Freeze the encoder and train the decoder independently.
- **Stage 2:** Unfreeze both the encoder and decoder, and start training both together.

These modifications helped stabilize training and enabled us to extract the best performance from the decoder. The loss logs for both stages can be understood from the figure below:

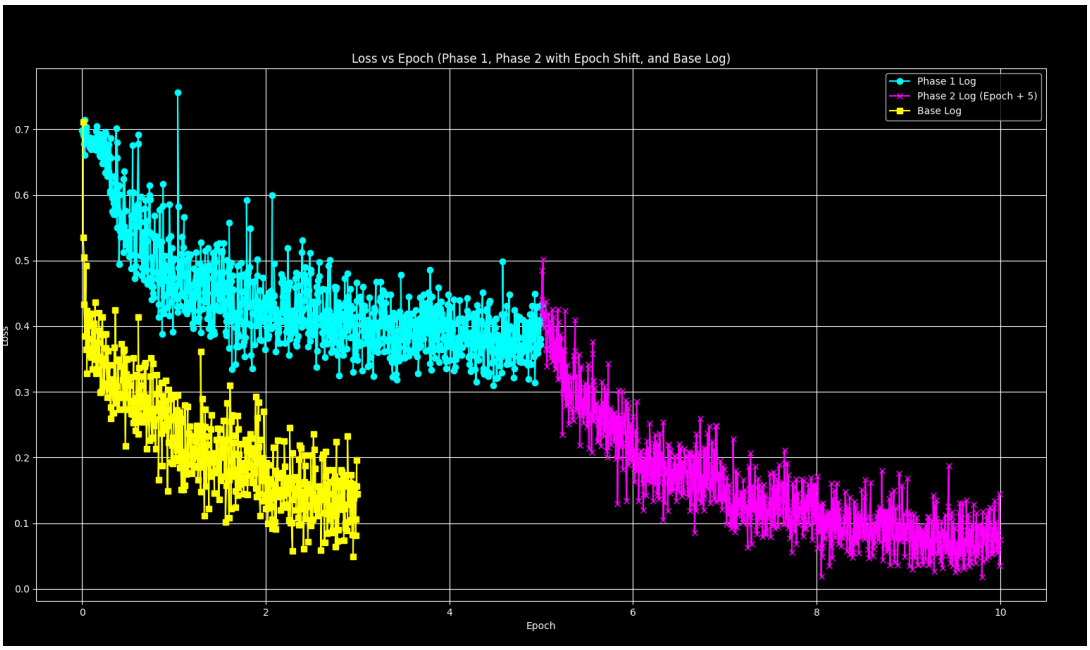

Figure 11: Training loss with decoder-only Fourier Transform replacement in a two-stage attempt. Yellow corresponds to the base model, Blue to Stage 1, and Pink to Stage 2.

Due to the complexity introduced by cross-attention, directly replacing attention with Fourier Transform led to a significant loss of information. To compensate for this, we ran the training for more epochs and this multi-epoch training allowed the decoder to recover the lost information and enabled us to extract the best performance. Specifically, we ran the model for 5 epochs in each stage, whereas the base model was trained for only 3 epochs.

4.3 Encoder + Decoder

In this experiment, we replaced all instances of self-attention and cross-attention in both the encoder and decoder with the Fourier Transform (FT) and benchmarked the performance on the SST-2 dataset.

We explored two different training strategies:

- **Strategy 1:** Epochs = 5, Batch Size = 16, Gradient Accumulation = 2
- **Strategy 2:** Epochs = 3, Batch Size = 1, Gradient Accumulation = 1

The training loss along with the learning rate schedule for these strategies is shown below:

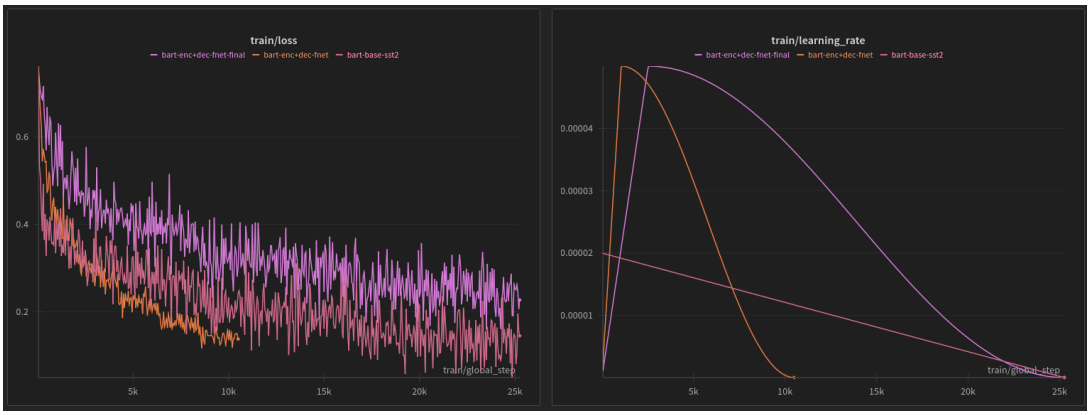

Figure 12: Training loss along with learning rate across training steps for the encoder + decoder Fourier replacement. Red corresponds to the base model, orange to Strategy 1 (Epochs = 5, BS = 16, GA = 2), and purple to Strategy 2 (Epochs = 3, BS = 1, GA = 1).

We observe that the gradient accumulation strategy used in **Strategy 1** performs better than **Strategy 2**. This is likely due to the **effective batch size being larger in Strategy 1, allowing for more stable gradient updates**. Although both strategies yield slightly lower performance compared to the original base model, the results are still remarkable given that all attention mechanisms in both the encoder and decoder have been replaced with the Fourier Transform.

Combining all the above experiments ,

Model Setting	Eval Loss	Eval Accuracy	Time per Epoch (s)
Base (BART)	0.3546	92.78%	17m 13s
Decoder-Only* (FT)	0.3090	92.20%	12m 33s
Encoder-Only (FT)	0.5593	83.37%	14m 17s
Encoder + Decoder (FT)	0.3939	**83.49%**	**9m 17s**

Table 2: Evaluation metrics for different Fourier Transformer configurations.

From the above table, we observe a trade-off between accuracy and training time. One particularly remarkable result is seen in the **Encoder + Decoder** replacement model, where the total evaluation time is nearly halved (from approximately 17 minutes to 9 minutes) compared to the base model, with only an 8% drop in accuracy (from 92% to 84%).

Additionally, the superior performance of the **Decoder-only** model can be attributed to the **two-stage training strategy it underwent**.

5 Phase 3: Complex Information + JEDI

5.1 Complex Information

We came across a very unique type of implementation across all the papers, like FNet, FFTNet, or even FourierTransformer [5]. All of them use only the **real part of the Fourier transform**. We tried to

incorporate the complex part as well and tested it on various multimodal datasets, and the performance was quite amazing.

Essentially, we constructed a **small MLP overhead that takes in both the real and complex parts of the Fourier transform** and projects it onto a normal d-dimensional space from a $2 \times d$-dimensional space, while the rest of the architecture remains the same.

5.1.1 Image - CIFAR-10

We fine-tuned both the real-only FNet encoder and the real + complex FNet encoder BART models for 10 epochs. The training and validation metrics are shown below.

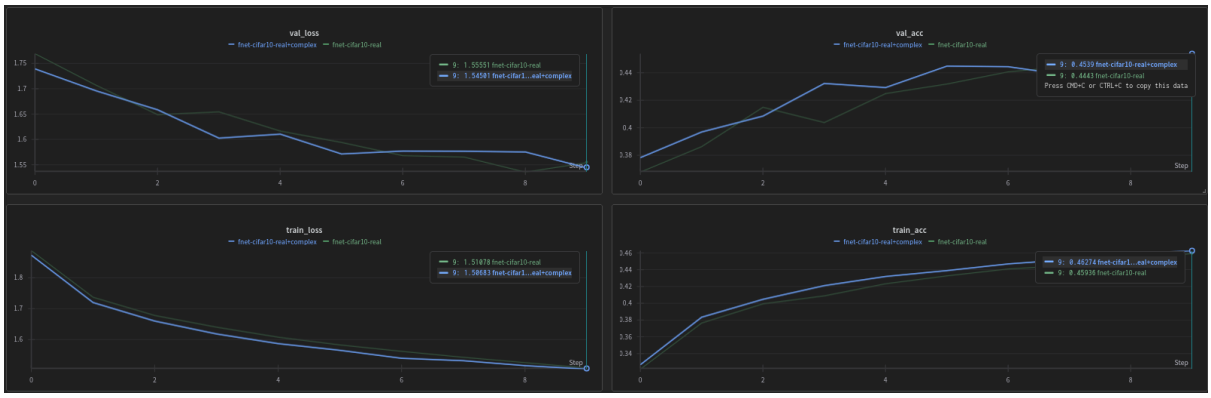

Figure 13: Training and validation loss and accuracy on CIFAR-10.

We observe that the model performance improves by 1.03% in validation accuracy when the complex part is incorporated.

This is a decent result. Although we don't see a major boost in accuracy, it suggests that **image-based datasets might not heavily benefit from complex-valued features**. To confirm this, we next evaluate performance on an audio dataset.

5.1.2 Audio - SpeechCommands

To further investigate the importance of complex-valued information, we used the SpeechCommands dataset but restricted it to six classes: *down, left, no, right, up, yes.*

We extracted audio samples and computed **Mel-frequency features**, which were then passed into the model.

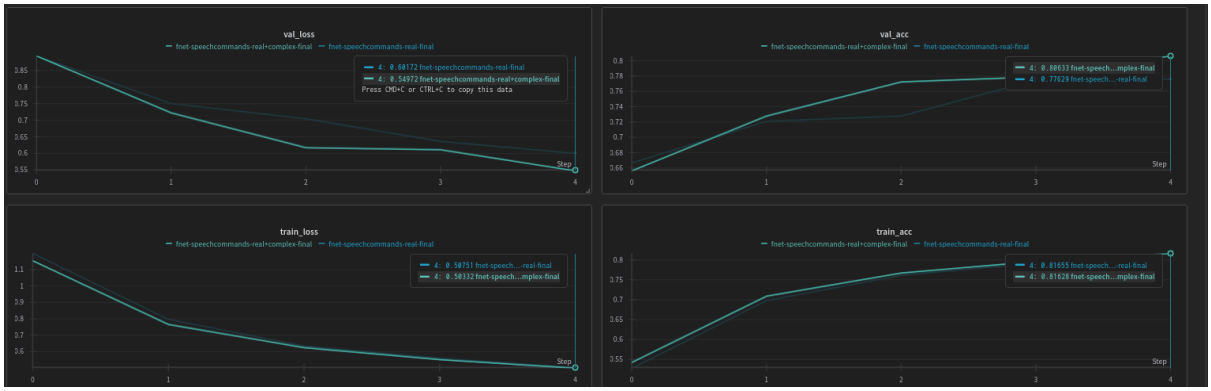

Figure 14: Training and validation loss and accuracy on SpeechCommands.

As shown above, we observe a clear 3% improvement in validation accuracy when complex information is incorporated. This suggests that complex-valued representations play a crucial role in audio datasets, likely due to the **phase and frequency components preserved in the complex domain.**

5.2 JEDI (Justifiable End-dialogue Driven Interaction)

We explored a very interesting custom project from **Stanford University** known as **JEDI**, which investigates the potential of large language models to revolutionize role-playing game (RPG) dialogue generation. The case study used in this project is the game *Star Wars: Knights of the Old Republic (KOTOR)*.

Traditional RPGs typically suffer from rigid, **one-to-one dialogue structures** that are heavily dependent on pre-scripted trees. These dialogues are static and influenced by a combination of **player choices, character stats, and game states**, which can limit replayability and immersion.

The key idea introduced by JEDI is to **represent "dialogues as graphs"**, **allowing dynamic and context-aware dialogue generation**. This structure enables the system to adapt to various in-game factors, supporting more interactive and personalized narrative experiences.

5.2.1 Dataset Creation + Understanding

In the JEDI project, the dialogue system is represented as a graph, where:

- **Nodes** correspond to individual dialogue utterances.
- **Edges** represent the transitions between utterances, which are determined by the game state.

To group similar dialogue nodes, a clustering algorithm based on a basic threshold F1 score is used. This algorithm helps categorize dialogue into meaningful clusters. Once the clusters are identified, the graph is **linearized** to facilitate efficient training.

During training, one utterance is randomly masked in a given sequence, and the model is tasked with predicting the masked dialogue line based on the remaining lines within the cluster and the current game state.

The reason for using attention mechanisms in this model can be understood as follows:

- $\mathbf{Q}$ = current dialogue input (query).
- $\mathbf{K}, \mathbf{V}$ = representations of the game state (keys and values).

The attention mechanism computes the dot product between the query and the key (QK^T) to determine the similarity between the current dialogue input and the game state. This allows the model to weigh the relevance of different parts of the dialogue in relation to the current game context, facilitating more accurate and dynamic dialogue generation.

5.2.2 Comparing Our Best Models

In this section, we compare the performance of all four models: the base BART, encoder-only FNet, decoder-only FNet, and encoder + decoder FNet. The training loss logs for each model are shown below:

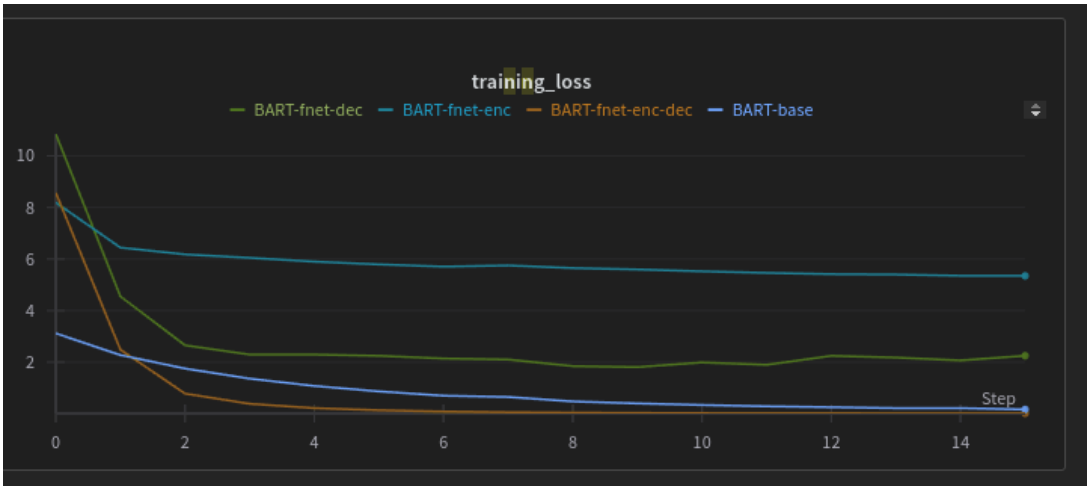

Figure 15: Training loss logs for the different models: Base BART, Encoder-Only FNet, Decoder-Only FNet, and Encoder + Decoder FNet.

Model	Precision	Recall	F1 Score	DialogueRPT
Base BART	0.86	0.86	0.86	0.50
Encoder + Decoder FNet	0.78	0.84	0.81	0.50
Encoder Only FNet	0.63	0.65	0.64	0.41
Decoder Only FNet	0.71	0.74	0.72	0.46

Table 3: Final Results on JEDI

We observe that the Encoder+Decoder FNet model performs similarly to the Base BART model used in the actual paper. The better performance of the decoder model can be attributed to the fact that the task is a text generation task which is heavily dependent on the performance of the decoder on with encoder embedding outputs. A future study could include understanding why the Encoder+Decoder FNet BART model overfits on the dataset. This could result in a big boost in performance.

6 Further Notable attempts

6.1 Benchmarking on SWAG Dataset

We attempted to apply the encoder-only, decoder-only, and encoder + decoder FNet models to the SWAG dataset, similar to how we did for the SST-2 dataset. However, the models did not perform as well, as shown below.

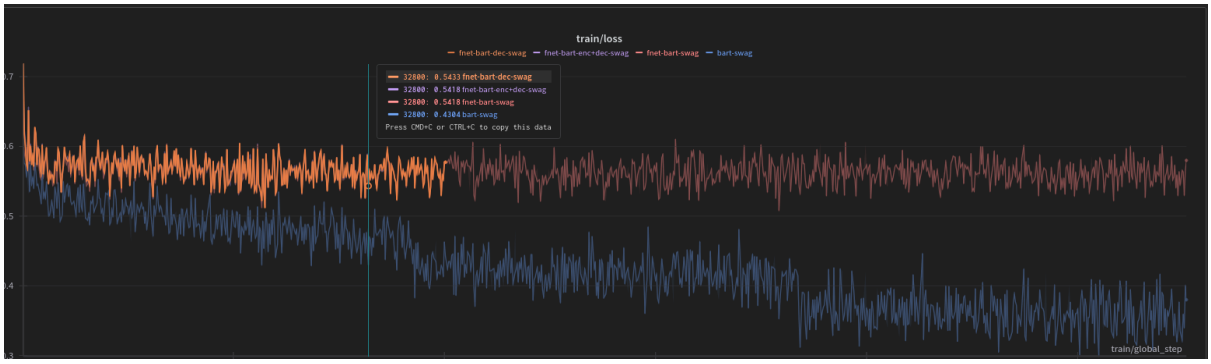

Figure 16: Performance of the encoder-only, decoder-only, and encoder + decoder FNet models on the SWAG dataset.

Clearly, all models failed to produce satisfactory results, emphasizing that **attention cannot be entirely replaced with Fourier transforms in all cases**. For tasks like SWAG, which require contextual understanding, attention mechanisms are essential, even if they are computationally expensive.

6.2 Extending FFTNet to SST2

We tried implementing the spectral adaptive filter, which was discussed in Section 3.2, to the existing FNet model and benchmarked it on the SST-2 dataset, hoping it would perform better than the base FNet.

But to our surprise, even after incorporating the filter, the model performance was almost the same as the base model , as shown below.

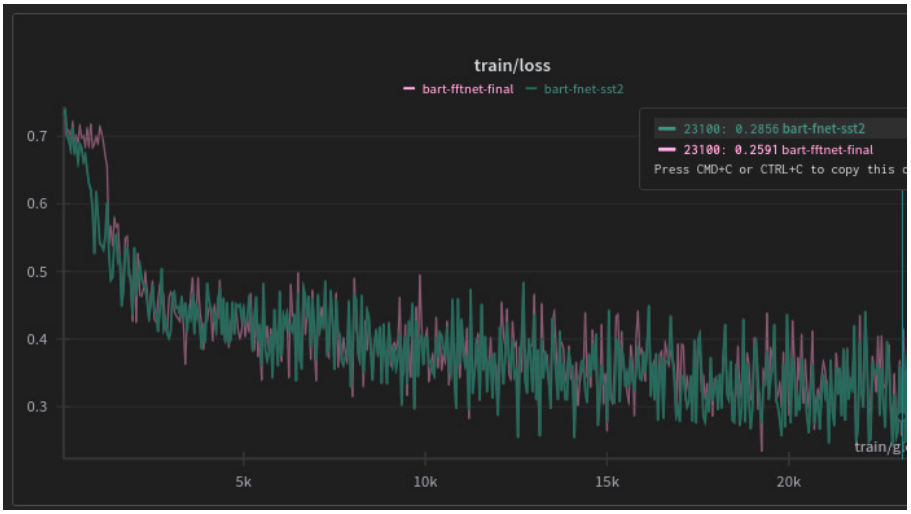

Figure 17: Performance of the adaptive filter on sst-2

This could be because SST-2 is a sentiment analysis dataset, which may not benefit as much from the adaptive filtering technique .

References

[1] J. Lee-Thorp, J. Ainslie, I. Eckstein, and S. Ontañón, *FNet: Mixing Tokens with Fourier Transforms*, arXiv preprint arXiv:2105.03824, 2021. [Online]. Available: https://arxiv.org/abs/2105.03824

[2] W. Chan, O. Abul-Hassan, and S. Sun, *JEDI: Justifiable End-dialogue Driven Interaction for NPC Entities in Role-Playing Games*, Stanford CS224N Final Project Report, 2025. [Online]. Available: https://web.stanford.edu/class/cs224n/final-reports/256911920.pdf

[3] J. Fein-Ashley, N. Gupta, R. Kannan, and V. Prasanna, *The FFT Strikes Again: A Plug and Play Efficient Alternative to Self-Attention*, arXiv preprint arXiv:2502.18394, 2025. [Online]. Available: https://arxiv.org/pdf/2502.18394v1

[4] Allen Institute for AI, *SWAG Dataset*, 2025. [Online]. Available: https://huggingface.co/datasets/allenai/swag

[5] Z. He, M. Yang, M. Feng, J. Yin, X. Wang, J. Leng, and Z. Lin, *Fourier Transformer: Fast Long Range Modeling by Removing Sequence Redundancy with FFT Operator*, arXiv preprint arXiv:2305.15099, 2023. [Online]. Available: https://arxiv.org/pdf/2305.15099