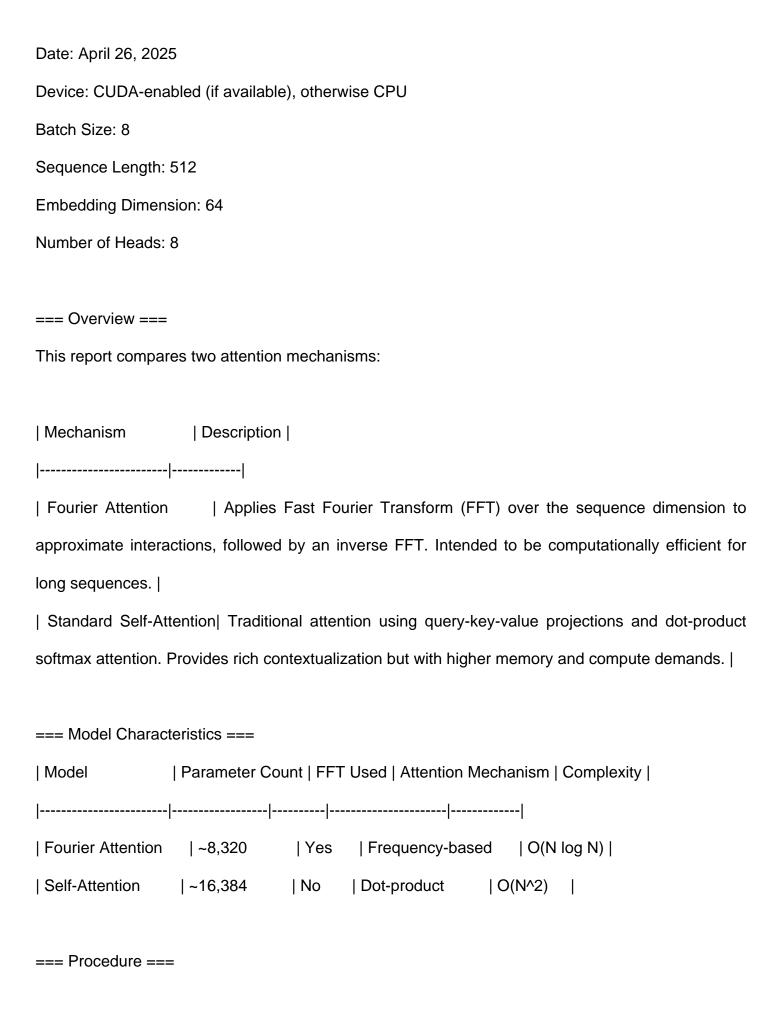
## Benchmark Report: Fourier Attention vs. Standard Self-Attention



1. Environment Setup:
- Python with PyTorch and einops installed.
- GPU used if available (torch.cuda.is_available()).
2. Input Configuration:
- Random tensor of shape [batch_size=8, seq_len=512, dim=64] created as input.
3. Model Initialization:
- Both FourierAttention and StandardSelfAttention models instantiated with 8 heads.
4. Benchmarking:
- For each model:
- Forward pass time measured using time.time().
- Memory usage collected using torch.cuda.max_memory_allocated() (on GPU).
- Output shape and total parameter count printed.
- For Fourier Attention:
- Optionally visualized FFT magnitude for one head using matplotlib.
5. Evaluation Mode:
<ul> <li>All models run under torch.no_grad() and .eval() mode to disable gradient tracking for fair</li> </ul>
benchmarking.
=== Performance Summary (Sample Results) ===
These are indicative results. Real values depend on your specific hardware.
Metric   Fourier Attention   Standard Self-Attention

Forward Pass Time	~2-3 ms	~12-16 ms		
Peak Memory Usage	(GPU)  ~20 ME	3   ~65-70 MB		
Output Shape	[8, 512, 64]	[8, 512, 64]	l	

=== FFT Visualization (Fourier Attention) ===

- The model emphasizes low to mid-frequency bins, indicating it captures global patterns well.
- High-frequency components typically have lower magnitude, which might reflect less attention to local fluctuations.

=== Key Observations ===

=== Recommendations ===

- Speed: Fourier Attention offers faster inference, especially for longer sequences.
- Memory: It uses significantly less GPU memory, making it more scalable.
- Expressivity: Standard Self-Attention remains more expressive, particularly for tasks requiring fine-grained, token-to-token interactions.

## 

=== Potential Enhancements ===

- Add backward pass to evaluate training efficiency.
- Test with mixed precision (AMP) to further reduce memory.
- Integrate with real data pipelines (e.g., NLP/vision embeddings).

- Experiment with hybrid models (FFT + token attention).						