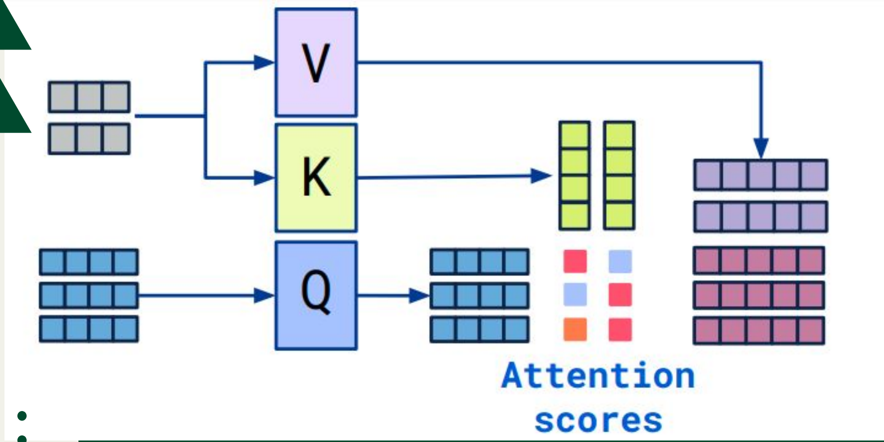
Optimizing Attention Mechanisms in Transformers



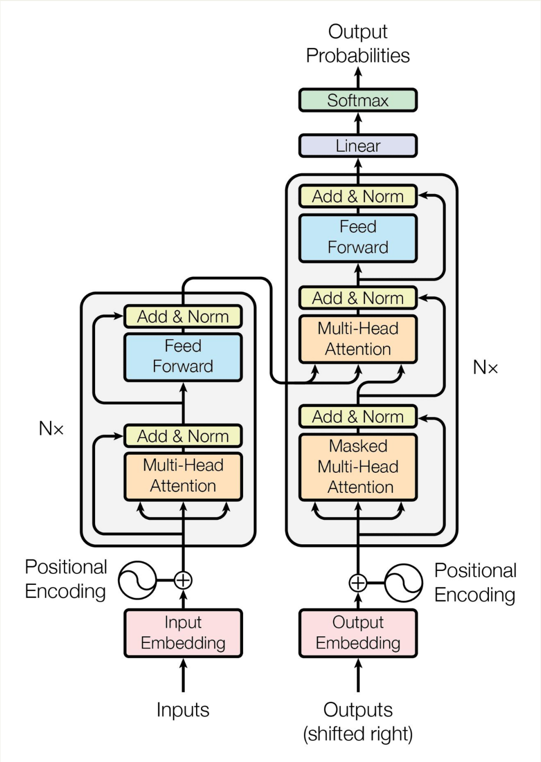
Chandler Cheung, Charis Gao, Jordan Hochman

Problem Definition

01

Background

- Transformer models are central to NLP tasks
- In recent years, size of models has grown exponentially
- Key challenge: $O(n^2)$ complexity in attention mechanism (pairwise interactions between all tokens in the input sequence)
- **Growing model sizes create memory constraints**
    - Need more efficient attention mechanism without degrading performance



Optimization Problem

- Train models with customizable and optimized attention masks to produce outputs similar to a baseline, unmodified transformer
- Goal: **preserve model quality** while **reducing memory constraints**

Metrics

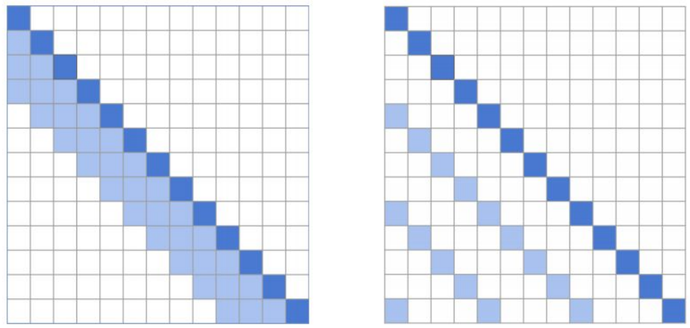
- **Coherence in output text:** comparable performance in generating text
- **Distribution alignment:** low KL-divergence with baseline model
- **Computational improvement**: reduced memory usage and speed improvements

Technical Approach

02

Literature Review

Sparse Attention: each token attends to a subset of other tokens

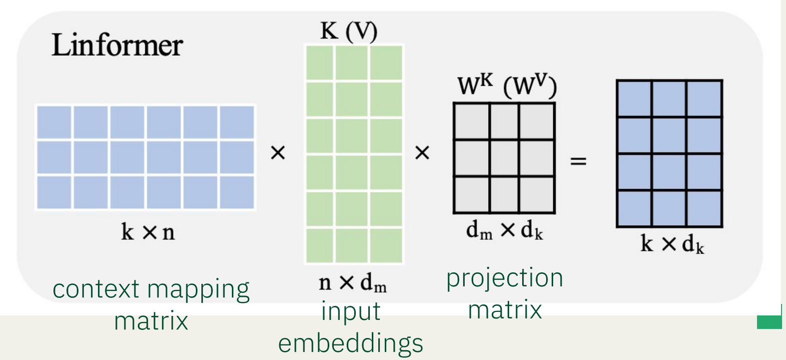


(a) Local attention (b) Strided attention

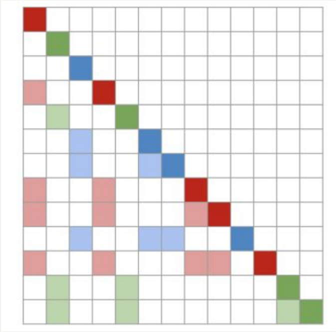
- [BASED](#) ([Arora et al., 2024](#)) – combines linear attention + sliding window attention
- [Native Sparse Attention: Hardware-Aligned and Natively Trainable Sparse Attention](#) ([Yuan et al., 2025](#)) – algorithmic innovation and hardware-aligned optimization for long-context optimization

Low-Rank Approximations: approximate attention matrix with low-rank matrices

- E.g. [Linformer](#) ([Wang et al., 2020](#)) – projects $n \times d_m$ query/key matrices to smaller $k \times d_k$ where $k << n$ → low-rank factorization of original attention



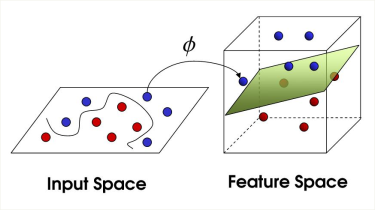
Literature Review



Efficient Routing/Dynamic Attention: dynamically determine which tokens should attend to each other

- E.g. Routing Transformer (Roy et al., 2021) – tokens mapped to routing space, uses online k-means clustering to assign tokens to clusters based on similarity in routing space, tokens attend only to tokens within same cluster

Kernel-based Methods: reformulate attention with kernel functions



- Kernel trick– performs this operation in the original space (implicitly compute dot products in some-dimensional feature space, without ever having to transform the vectors to that space)
- Performer (Choromanski et al., 2021) – replaces softmax attention, uses random feature mappings to approximate the exponential kernel

Many other methods: neural architecture search to discover more efficient attention patterns, mixture of experts that routes tokens to different attention modules, etc.

Mathematical Formulation

Core objective: **minimize KL-divergence** between baseline and custom model over all training examples

$$\mathcal{L} = \mathrm{KL}\left(P_{\text{custom}} \,||\, P_{\text{base}}\right)$$

- LLMs work by outputting tokens based on a probability distribution
- Want models with modified attention to have similar next-token probability distribution
- KL-divergence measures similarity between probability distributions

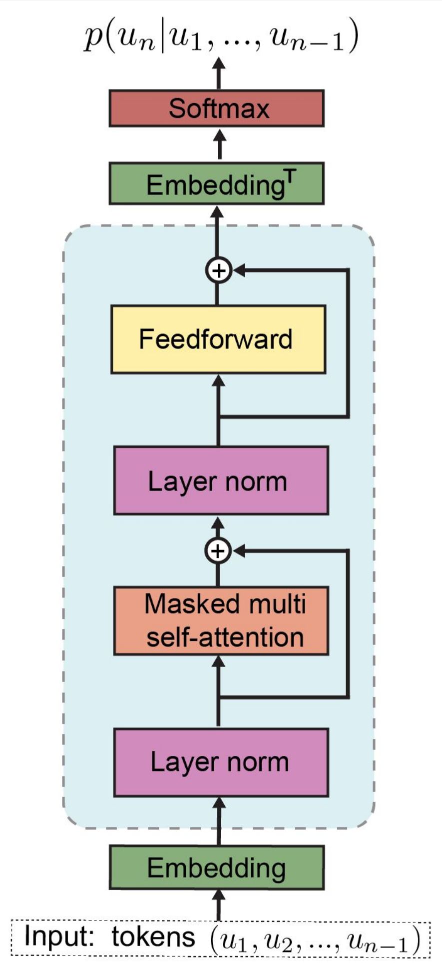
Models

- **Baseline model:** GPT-2 (unoptimized attention mechanism)
- **3 experimental models:**
  - Custom attention masks: test combination of simple masks
  - Performer: computational kernel trick without changing attention structure
  - Native Sparse Attention: recent development; compare with previous work
- **Dataset: WikiText-2**
  - Built from Wikipedia articles
  - Size: ~2M tokens training set, ~220K tokens validation set, ~240K tokens test set
  - Common dataset among literature, curated and easy to use

Baseline model

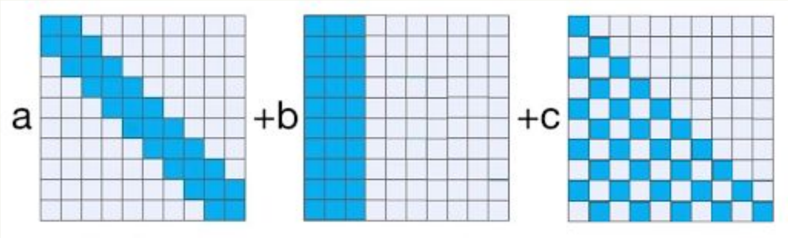
GPT-2

- Transformer-based language model built using stacked decoder blocks focused on self-attention
- Causal masking – each word in a sequence attends to all previous words using scaled dot-product attention
- 12 attention blocks, 1.5B parameters
- 1024 token maximum context length
- Trained on WebText, dataset of 8 million web pages
    - Links from scraped Reddit posts >3 karma. It *excluded* Wikipedia pages (common data source for other datasets)

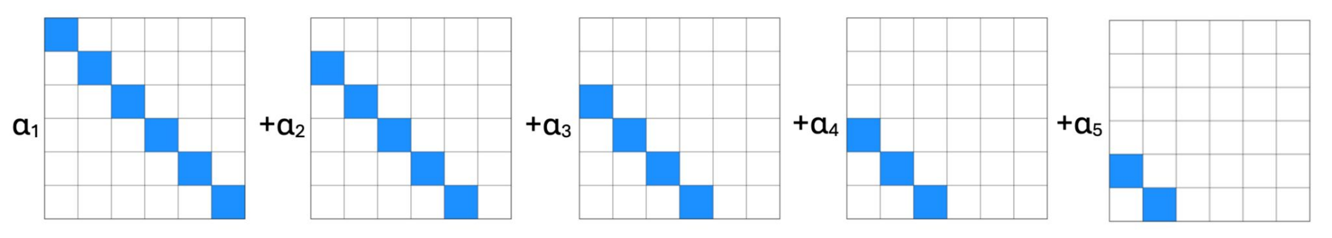


Custom Attention Masks

- Use linear combination of candidate attention masks with learnable weight parameters. E.g.



- Attention coefficients are learnable parameters, model weights are fixed
- Tested with and without L1 penalty on the coefficients
- The following results use a combination of 5 masks, each selecting only the i’th to last token



$$\mathrm{K}(\mathbf{x},\mathbf{y}) = \mathbb{E}[\phi(\mathbf{x})^\top \phi(\mathbf{y})]$$

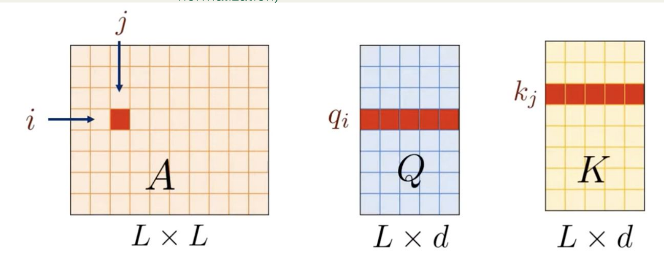
Performer

- Kernel approx. of softmax attention mechanism with random feature maps
  - A is approximated by lower rank matrices Q’ and K’ through a Fast Attention Via positive Orthogonal Random (FAVOR+) approach

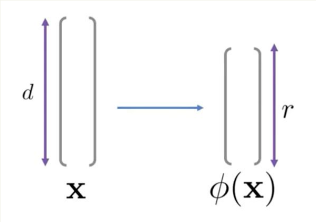
$$Y = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$$

$$Y = D^{-1}AV, \quad A = \exp\left(\frac{QK^T}{\sqrt{d}}\right), \quad D = \mathrm{diag}(A\mathbf{1}_L)$$

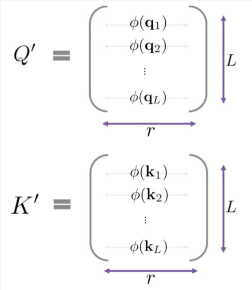
(exponentiated attention scores before normalization)

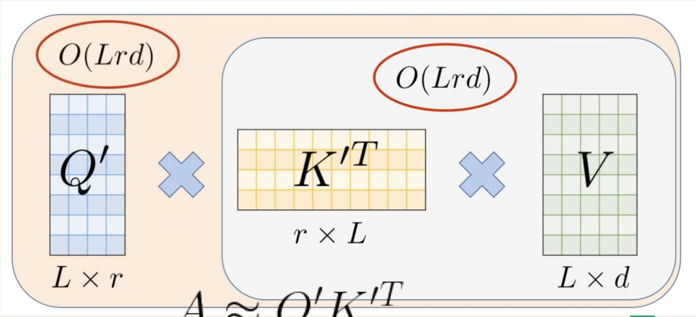


$$A(i,j) = \kappa(q_i^T, k_j^T) = \mathbb{E}[\phi(q_i)\phi(k_j)^T] \approx \phi(q_i)\phi(k_j)^T$$



Queries & keys are mapped into new feature space via φ; inner product approximates the exponential function (linear!)



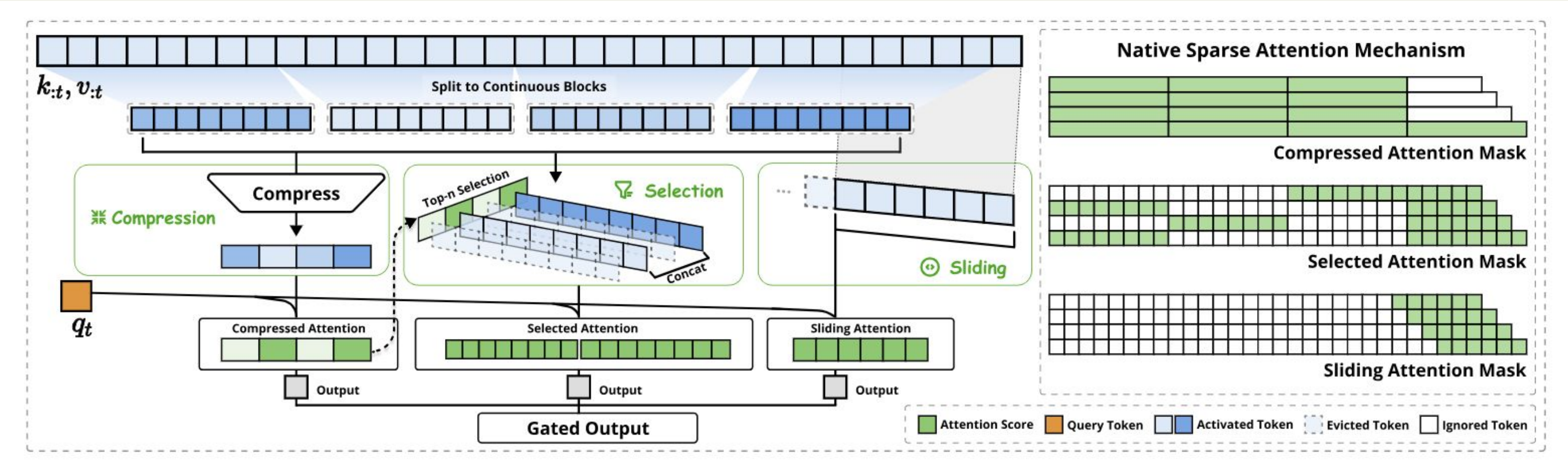


$$A \approx Q'K'^T$$

Performer

- Optimizing for
  - HP - number of random features parameter (r)
  - Query, key, value projection matrices + all standard Transformer parameters (eg. embeddings, LN, FFN weights, output projections, etc.)
    - Random projection matrices/features themselves are not being learned → fixed after initialization
    - **Optimizing linear projections around these fixed random features**
- Model learns how to best use these random features – distillation process helps ensure that this approximation is effective by guiding the model to produce outputs similar to the original attention mechanism

Native Sparse Attention



- Hierarchical approach with compressed token representation, selective attention, and sliding window approach
- Kernel design: Group-Centric Data Loading, Shared KV Fetching, Outer Loop on Grid

Implementation Details

- Objective: minimize KL divergence between probability distribution of next token from baseline model and modified-attention model
- Used AdamW optimizer and Cosine Annealing scheduler
  - Decouples weight decay from the adaptive update mechanism
  - Initial learning rate = 10^{-3}
- Sample text generation: temperature = 0.7, top k = 50
  - temperature < 1 → less randomness
  - top k → restricting number of tokens from which to sample

Loss Computation

- Compute logits from both models on the same input batch
- Calculate KL-divergence and minimize

```python
def kl_divergence_loss(logits_custom, logits_ref, mask):
    log_probs_custom = F.log_softmax(logits_custom, dim=-1)
    probs_ref = F.softmax(logits_ref.detach(), dim=-1)  # Detach reference model

    # Calculate per-token KL
    kl = (probs_ref * (probs_ref.log() - log_probs_custom)).sum(-1)

    # Apply padding mask and average
    active_tokens = mask.sum()
    return (kl * mask).sum() / active_tokens
```

Hyperparameter Tuning

Key HPs:

- Learning rate = 10^{-3} (initial)
- Block size = 128 (context window size/maximum sequence length processed during training)
- Custom attention masks: coefficients for linear combination of masks
  - Tried random initialization & zero initialization
- Performer: number of random features for kernel approximation

Tuning procedure:

- AdamW optimizer and Cosine Annealing scheduler

Implementation Choices

Pytorch modules

Optimization: torch, torch.nn, torch.optim, torch.nn.functional, torch.utils.data, datasets, matplotlib

Model: Hugging Face transformers, GPT2, native_sparse_attention_pytorch

Miscellaneous: time, os, psutil, tqdm

Specific features: dataloading, training loop

Implementation Choices

Additional

Benchmarking: psutil, time, torch.cuda.memory_allocated, torch.cuda.memory_reserved

Checkpointing: Saved model weights periodically with torch.save/torch.load

Limits Encountered

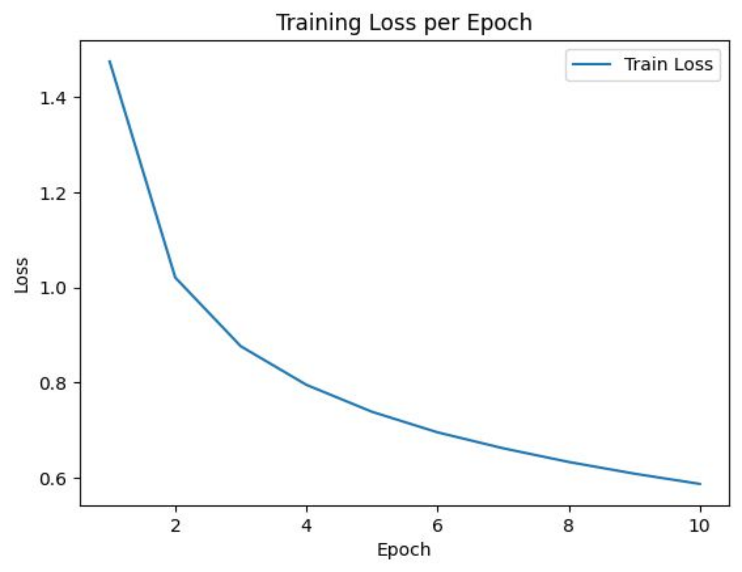
- Google Colab Notebooks only allowed us to train on GPUs for ~3-4 hrs/day, runtime often disconnected
    - Slow iteration speed, hard to test several different hyperparameter configurations
    - Saved checkpoints of models
- Could not implement own NSA mechanism
    - Used open-sourced library by Meta engineer

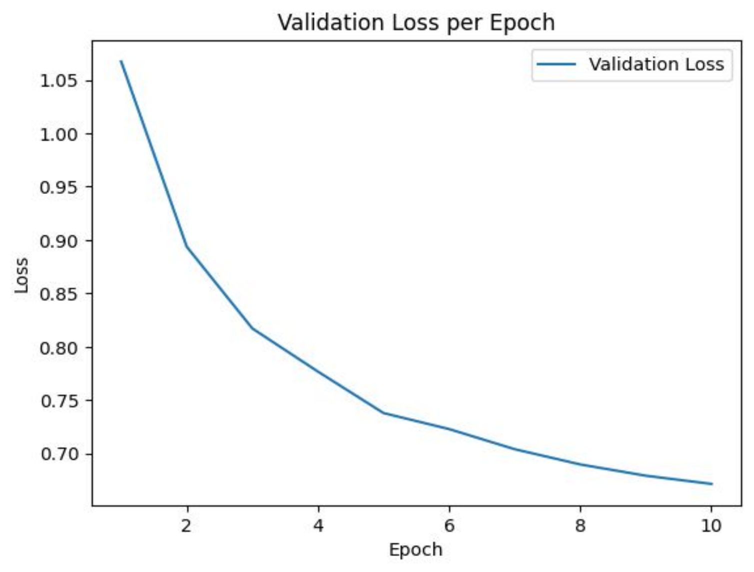
Results

03

Custom Attention Masks

- Trained for 10 epochs, training loss decreased from 2.6113 to 0.5875





Custom Attention Masks

- Decent output quality

Prompt: In a shocking turn of events,

Reference: ... the police had not been able to contact any witnesses and were refusing to answer questions in court. In fact when ...

Custom: ... the government is now facing opposition from some members but also other conservative groups ...

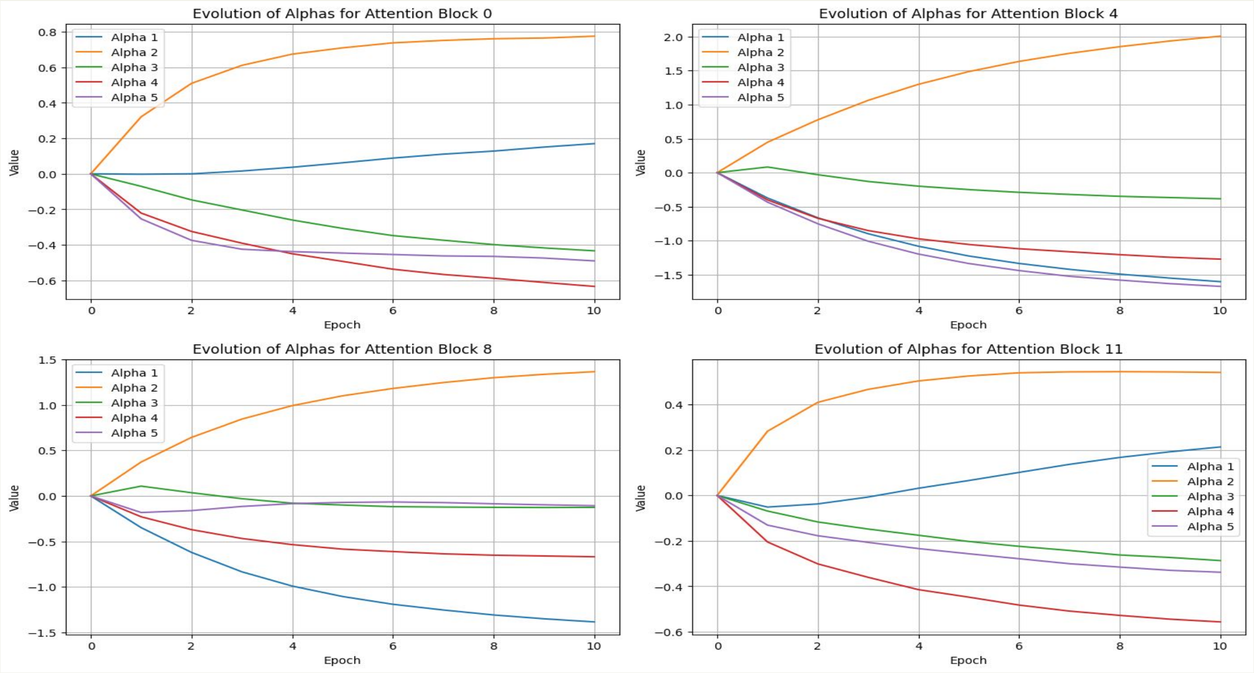
Prompt: The future of artificial intelligence

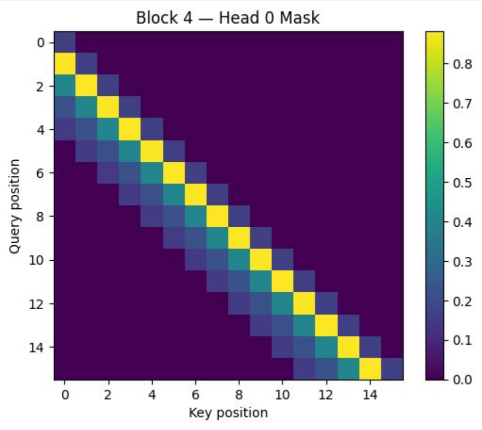
Reference: ... is up in the air," said Michael Lindzenich, an analyst with Capital IQ ...

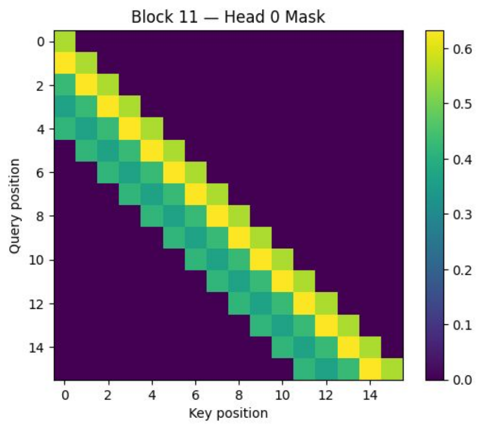
Custom: ... , which in turn enables us to develop new paradigms , and many experts have found the methods that can be used as a tool for assessing how people are affected ...

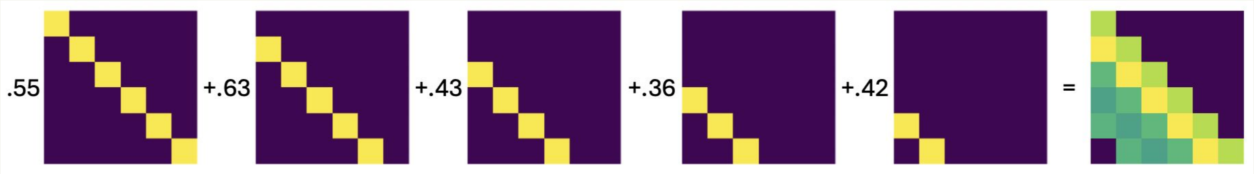
Prompt: The meaning of life is
Reference: ... not a function that we have to live in. Life requires us to be aware ...
Custom: ... that it has no intrinsic worth. The Lord knows his God and will give him power to do great things for others . " This would be the last year or we should have a few days ...

Custom Attention Masks

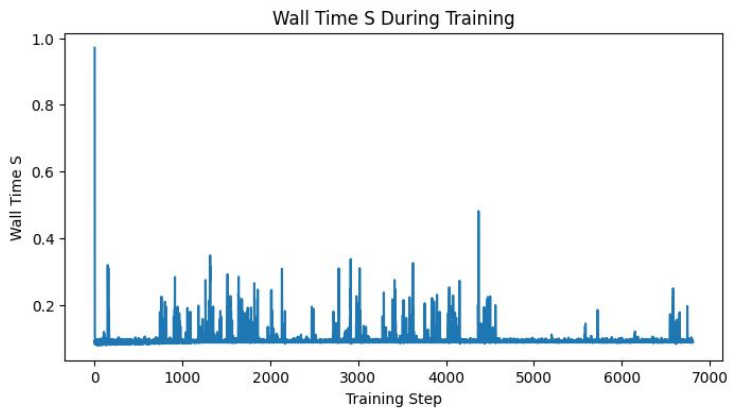


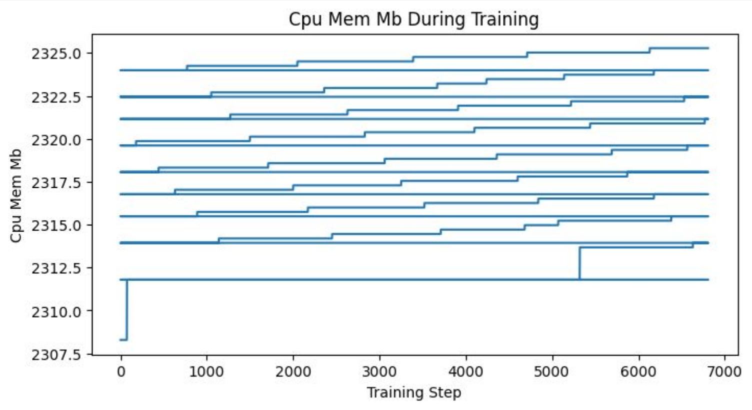


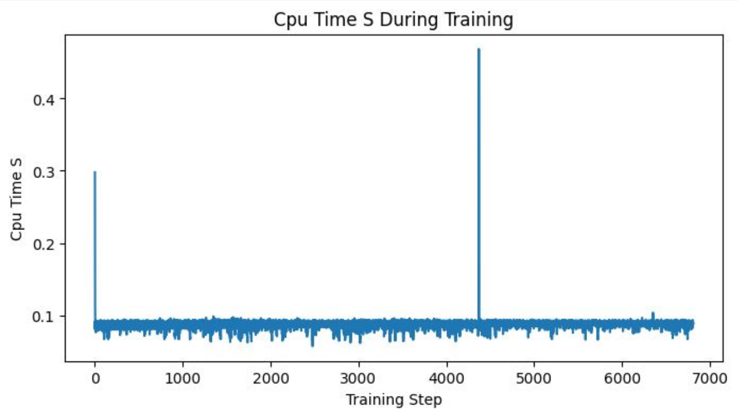


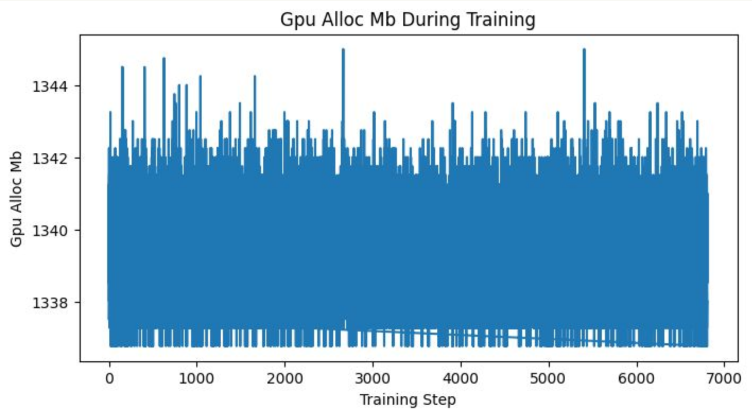


Training Performance

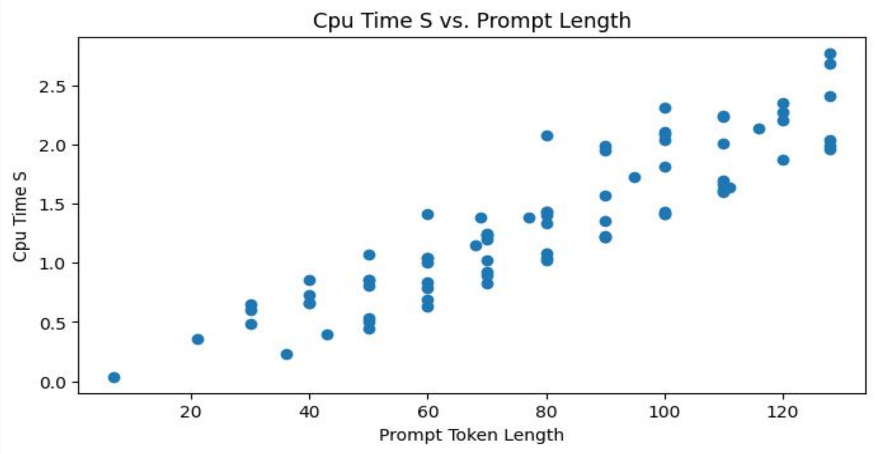


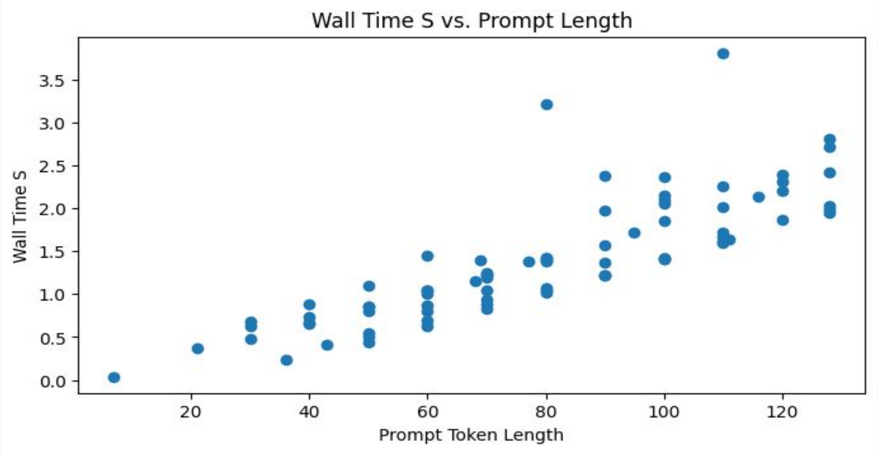


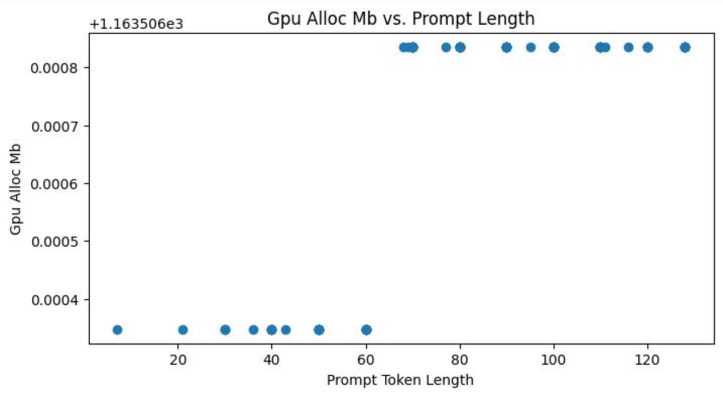




Inference Performance

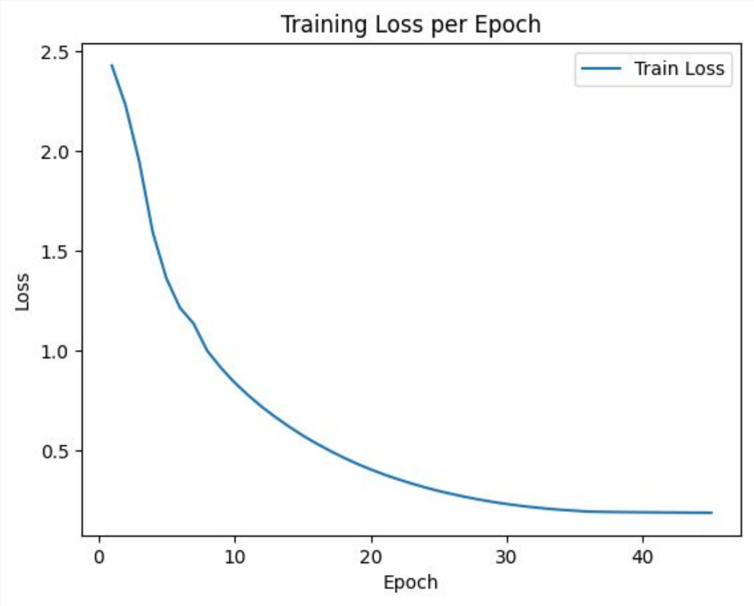


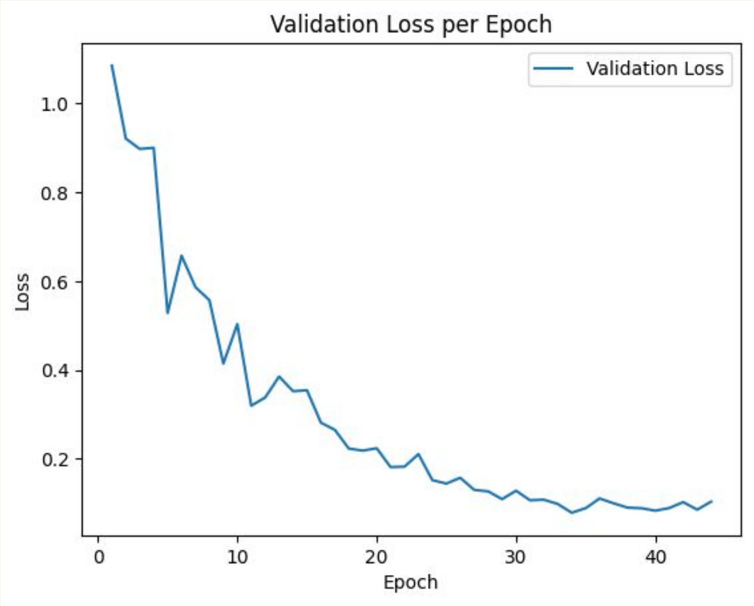




Performer

- Trained for 45 epochs, training loss decreased from 2.4290 to 0.1872





Performer

- Poor output generation quality

Prompt: In a shocking turn of events,

Reference: ... the group is now being accused by their supporters and even former members ...

Custom: ... and to get drunk as well with little girls in the night before entering its relationship between friends who became an investigation ...

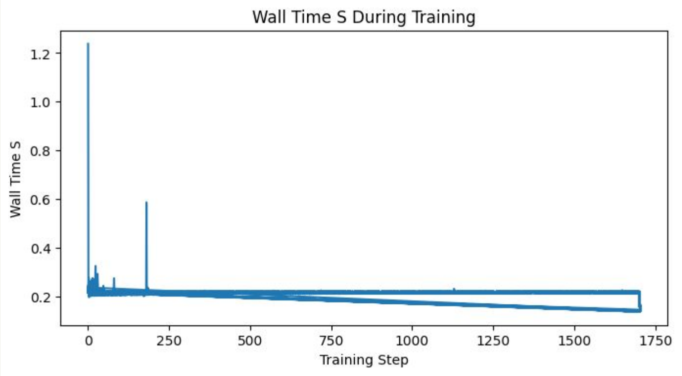
Prompt: The future of artificial intelligence

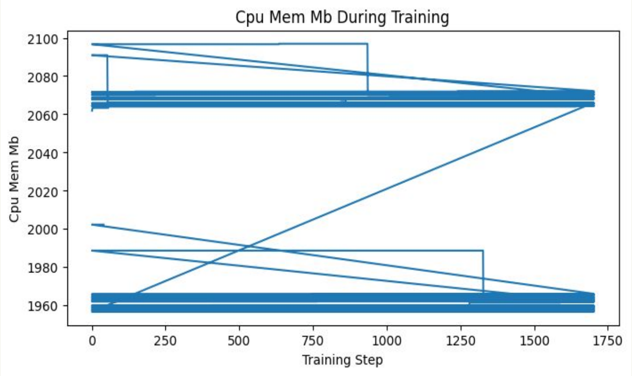
Reference: ... is uncertain. For example, AI will never be able to solve problems such as ...

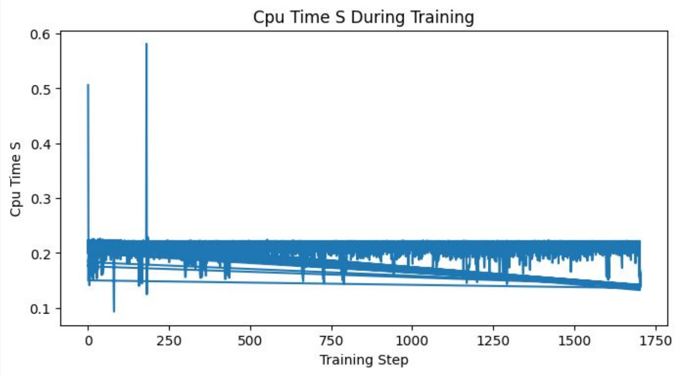
Custom: ... , in the first instance; that both sides because there is a particular to be used for those who could not only one's power. ...

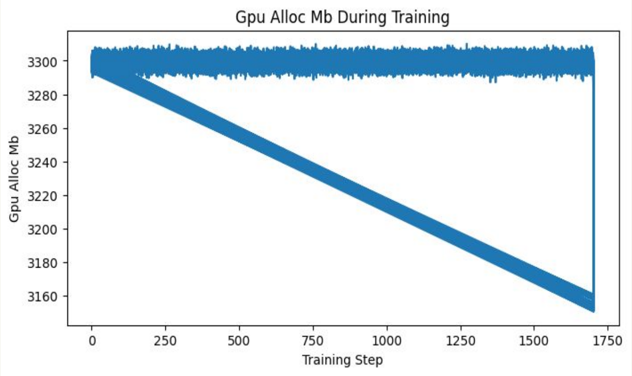
Prompt: The meaning of life is
Reference: ... that the world can do things and they will be good. The word "life" does not ...
Custom: ... also the use to be, where you can find out. The reason for an individual human beings and one who are a major problems with no other ...

Performer


Wall Time S During Training
Wall Time S
Training Step

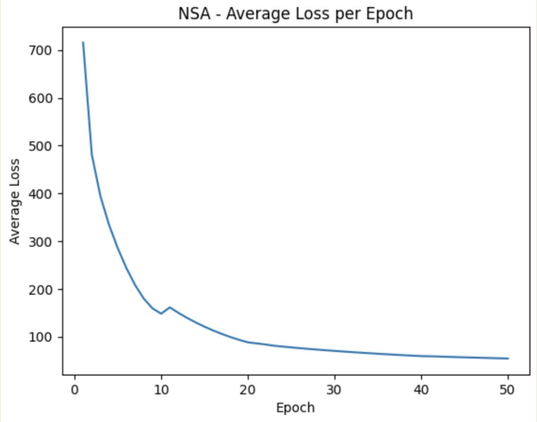

Cpu Mem Mb During Training
Cpu Mem Mb
Training Step


Cpu Time S During Training
Cpu Time S
Training Step


Gpu Alloc Mb During Training
Gpu Alloc Mb
Training Step

Native Sparse Attention

Trained for 50 epochs, training loss decreased from 713.74 to 33.46



Prompt: In a shocking turn of events,

Reference: ... the media has been unable to bring forward evidence to substantiate this claim ...

Custom: ... the public was caught, but the results of a heated debate statement by the Democratic leadership committee to "attalal"ists and the Communist Left Front's Project to to the Centre in Washington ...

Prompt: The future of artificial intelligence

Reference: ... is the question whether it will be able to replace human judgment in the future ...

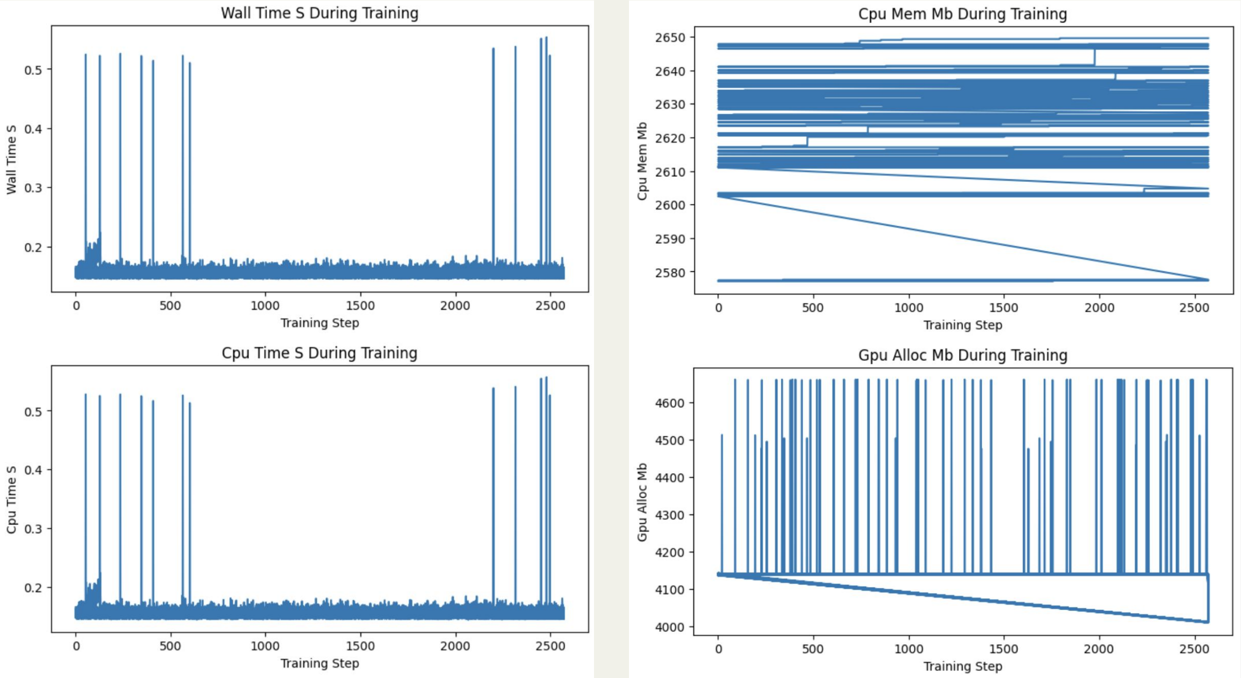
Custom: ... , a world is a world class state, with less money to $10 million-$5 million to $ $ 1 billion worth the public money raised by $ 1 ...

Prompt: The meaning of life is

Reference: ... that we make choices that are based on the best way to live. If we have the right choice, the choices make us happier ...

Custom: ... a matter for a second or a second one in. I.I. In this manner, I have to think of, not one of my own creation principle. When the first and final, are all:, was at is ...

NSA Training Performance



Overall Results

- Achieved low KL-divergence between custom and baseline models
  - Probability distributions should be mostly aligned → this statistical similarity doesn't translate to human-perceived quality
  - KL-divergence alone might be an insufficient metric for capturing language quality and coherence → gap between statistical and semantic performance (aligning token distribution patterns is not enough)
- Intrinsically, the architectures we used all have sublinear performance for memory/time in the attention mechanism
- Simplest approach worked best (least capacity to change underlying model)
  - More complex mechanisms changed attention structure and training potentially adapted model to WikiText-2 dataset

Expected vs. Actual Progress

Expected Progress

- Expected one model with similar model output accuracy for long context
- Expected sublinear time/memory
- Expected generation coherence to improve with lower KL-divergence (training longer → better output)

Actual Progress

- Three different models with differing model output accuracy
- Sublinear time/memory, attempted to measure speed and memory usage
- Generation coherence does not improve much with KL-divergence

Limitations

- Slightly different implementations for each model
- Limited context window due to compute and runtime constraints
- Train models on larger/varied datasets for more epochs and experiment with context length so that coherent English words are produced → currently accuracy concerns / limited coherence
    - Could not train on original data that GPT-2 was trained on (WebText–internal OpenAI corpus)

Project Reflection

04

Team Reflection

- **What was the number one technical or conceptual difficulty?**
  - Implementing our tested models such that they were consistent with each other, and tuning/optimizing each model separately
- **What part of the project workflow was easier than expected? Harder?**
  - Getting most of the code down for each model was easier than expected
  - Fine tuning and training was harder than expected, especially with runtime/compute issues → required detailed checkpointing
- **How did your project goals or approach evolve based on self-critiques or intermediate results?**
  - The self-critiques helped us realize that we needed to refine our goal/scope
  - They also showed our various models were disorganized and hard to compare
- **How did AI tools assist your project (e.g., coding, debugging, writing, brainstorming)? Give specific examples.**
  - AI helped implement some of the models, especially when we could not find source code from the research papers, in addition to debugging
  - It helped brainstorm existing methods and new ways to attack the quadratic attention problem

Individual Contributions

05

Individual Contributions

Chandler

- Most surprising result/finding
  - Decreasing KL divergence loss does not always produce more coherent output text
- Most useful course concept
  - Adaptive optimization methods and hyperparameter settings
- Perspective on optimization
  - Lots of trial and error with many decisions that work in practice but aren't always easily explainable
- Next steps given 2 more weeks
  - Test other loss functions, train longer, evaluate other baseline models
- Biggest change if restarting project
  - Explore other sparse attention patterns and implement own NSA

Individual Contributions

Charis

- Most surprising result/finding
  - Hard to tune models with random features i.e. repeating training runs for Performers have high variance (likely because random projection aspect of Performers are different every run)
- Most useful course concept
  - Systematic procedure for tuning models, learning rate schedulers (initially hit noise floor)
- Perspective on optimization
  - A lot more frustrating than I thought it would be, eg. unclear how HPs would affect performance, many hacks because limited by computational resources
- Next steps given 2 more weeks
  - More HP tuning or train on a different dataset → make performer output more coherent
- Biggest change if restarting project
  - Have a clearer final project goal at the beginning / midpoint presentation– felt like we were trying many approaches at the same time and it wasn't clear what we were optimizing

Individual Contributions

Jordan

- Most surprising result/finding
  - The custom attention mask had surprisingly coherent model output for limited context
- Most useful course concept
  - Inner workings of transformers and understanding of various loss functions
- Perspective on optimization
  - Takes a ton of time, and up front is usually taken for granted (e.g. people show results “after optimizing”, but this understates the time and effort required for tuning).
  - However, once you have a target formula, anything can be optimized! (over a lot of time)
  - Side note: Not too impressed with tooling for developing/optimizing models. This is mainly just about Google Colab, as the UI/UX doesn’t work seamlessly and is very hard to organize.
- Next steps given 2 more weeks
  - Get more cohesive results/comparisons between the tested models, and train with more compute and larger context windows
- Biggest change if restarting project
  - Try to define the scope better at the start instead of exploring aimlessly for a little bit

Thank you!

Any questions?