

Twin Head Transformers Are More General Learners

Mattia Opper¹, Roland Fernandez², N. Siddharth¹, Adam Davies³,
Paul Smolensky², Xiaodong Liu², Jianfeng Gao²,

¹University of Edinburgh, ²Microsoft Research, ³Microsoft,

Correspondence: m.opper@ed.ac.uk

Abstract

Self-Attention is a powerful and flexible primitive, forming the foundation for virtually all modern language models. While undeniably capable, these systems quickly degrade when faced with long context, especially if the length is OOD. We introduce TWIN HEADS, a reformulation of attention that allows transformers to learn selective memories of the sequence at each step. TWIN HEADS introduces a minimal parameter overhead and is compatible with tiled computation resembling flash attention, meaning it can stay memory efficient too. We find that TWIN HEADS greatly enhances generalisation to unseen context lengths, and does so while scaling favourably, showing competitive performance at up to 1.4B scale.

1 Introduction

As progress continues to compound, large language models (LLMs) are being charged with far more complex tasks. While their ability to operate autonomously is improving (METR, 2025), significant challenges remain. In particular long context remains an issue, with models struggling as length increases, and failing catastrophically once it falls out of distribution (Liu et al., 2023b; Zhou et al., 2024; Huang et al., 2025; Hong et al., 2025). This is far from ideal. Truly complex operations require processing and synthesising a lot of information and we’d like models that grow from, rather than drown in, such information.

To understand the issue, as well as potential solutions, we have to consider the trade-off between two classes of architecture: transformers, in particular the decoder-only variant used as language models, (Vaswani et al., 2017) and state space models (SSMs) (Schlag et al., 2021; Gu et al., 2022; Gu and Dao, 2024; Yang et al., 2025b). Transformers excel at *retrieval*, because self-attention lets them pull information from anywhere in the sequence.

However, they struggle with *time*. Temporal ordering is generally provided through positional encodings, which are rigid and become incoherent out of distribution, leading to catastrophic failures within the model. State space models, on the other hand, excel at temporal processing, effortlessly remaining coherent even when faced with out of distribution sequence lengths. They do so through the use of data dependent time decay, which allows the hidden state to forget past information when something new is encountered that renders it no longer salient. This provides SSMs with a flexible, multi-timescale memory. Information can be overwritten when encountering a structural transition point (e.g. a sentential boundary) or for more specific features e.g. when a particular entity takes a new action; because forgetting is data dependent it becomes an expressive and flexible way of propagating information through time. However, SSMs are far inferior to transformers when it comes to retrieval (Waleffe et al., 2024; Jelassi et al., 2024). The entire sequence history is compressed into a single vector, and even if it is a very capable memory this cannot compare to a transformer which can reach back whenever it needs to.

Retrieval and content-based time decay are both crucial if we want models that generalise well to long contexts; but how do we combine them? We introduce TWIN HEADS where attention is derived from the interaction of standard and auxiliary gate heads, the latter of which directly controls when to forget and where to focus, thus enabling and aiding both capabilities. We find that when trained with TWIN HEADS, transformers exhibit significantly enhanced OOD generalisation and long context capabilities. Moreover, TWIN HEADS introduces negligible additional parameters (in fact fewer than most positional encodings) and is compatible with FlashAttention-like online computation. As a result we are able to scale it and show that it can produce highly capable language models.

2 Model

A core strength of the transformer is that it is able to represent many states in parallel. Each column in the network can pull information from different parts of its context, thereby building its own unique set of connections. TWIN HEADS is designed to play to this strength, and enhance self-attention’s expressivity by allowing each *query* to form its own selective memory as well. To do so we begin by splitting each head in self-attention in two paired sets: a standard head (with logits $S = QK^T/\sqrt{d}$) and a gate head (with logits $G = Q'K'^T/\sqrt{d}$).

Selective Memories and Time Decay: one of the two key primitives for effective long context capabilities is a flexible, selective and multi-timescale memory. This is the core capability that enables SSMs to be robust to long and out of distribution context lengths. How do we incorporate it in a transformer? Prior work shows us two pathways to doing so, both of which require a parameterised forget gate. For each input position i , we can compute a decay $\delta_i = \log \sigma(W_f x_i + b) < 0$, where x_i is the input to the layer at position i , $W_f \in \mathbb{R}^{E \times 1}$ is a forget projection weight, b scalar bias, and σ denotes sigmoid. The vector δ contains decay values for each sequence element which can be used as relative positional encodings (Dai et al., 2019) (i.e. biases on the unnormalised attention weights). However, to do so we must find a way of turning δ into a penalty matrix Δ that influences each pairwise interaction, leading to two approaches.

The first approach, resembling ALiBi (Press et al., 2022), treats δ as representing a query’s receptive field (i.e. lookback window) with penalties

$$\Delta_{ij}^c = \sum_{t=j}^i \delta_i = (i - j + 1)\delta_i, \quad j \leq i. \quad (1)$$

Note that each query (or row) in Δ^c computes a suffix sum over the same constant scalar δ_i value (i.e. identical columns), yielding a constant linear decay. This allows flexibility in how far back a query wants to look, but makes the memory agnostic to what it encounters along the way. Where attention should terminate on, say, a sentence boundary, this fails as forgetting is agnostic to the content of the sequence. Contentful forgetting, introduced by the forgetting transformer (FoT) (Lin et al., 2025), can be achieved as

$$\Delta_{ij}^r = \sum_{t=j}^i \delta_t, \quad j \leq i. \quad (2)$$

Here, time decay is **data dependent**. Decay rates change at each time step t , yielding a more flexible piecewise linear form. This means that model can learn to forget the past after encountering e.g. a change in topic or a syntactic boundary. This second form is more flexible and superior to the first, as demonstrated by (Lin et al., 2025). However, it suffers from the limitation that the same decay values are shared across the sequence (identical rows). This both robs each query of the ability to define its own receptive field and restricts memory by making it uncontextual. Each token must predict what will be relevant and what should be forgotten without knowing what lies further ahead. If incorrect, the decision cannot be overwritten. In the absence of a perfect predictor, this is clearly suboptimal. In TWIN HEADS we instead use the gate head G to combine both forms, providing the model with data dependent forgetting that is adaptive to the needs of each time steps. We achieve this by making it **contextual**, taking advantage of G as

$$\Delta_{ij} = \sum_{t=j}^i \delta_i \sigma(G_{it}), \quad j \leq i. \quad (3)$$

TWIN HEADS decay (3) allows each query to define its own lookback window (δ_i), but conditions its transitions on the gate head’s activations. The sigmoid enforces their actions as soft on or off switches deciding when a boundary transition is reached. Crucially, unlike FoT (2), it *does not* impose these decisions on future time steps. This allows each query to define a unique segmentation of the past, capturing different snapshots depending on what is relevant at the time. This is highly advantageous when we want to, say, track the actions of multiple characters in a book, or have each token pay attention to the the specific set of syntactic features that determine its role in a syntactic parse. Such operations are temporally / locality bound while simultaneously being context dependent.

Content Gating: The second role of G is to inhibit S by acting as a content gate. Not all attention operations are positional, and so far we have only used G to determine locality. The model also may want to perform retrieval operations where it simply wants to match a very specific set of key(s) not matter how far away it is. For such an operation attention needs to be highly selective. A very good way of doing this is by distributing the operation across multiple heads, because they can each specialise

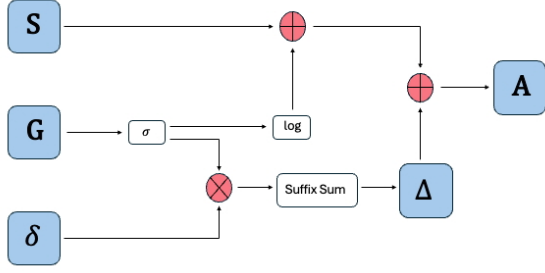


Figure 1: TWIN HEADS attention

for different features which when combined enable a discriminative decision (Shazeer et al., 2020; Ye et al., 2025; Golovneva et al., 2025). Given that we have already defined attention as the interaction between paired heads we can take advantage of G by having it help here to, serving as a content gate. This can be achieved simply as follows:

$$S + \log \sigma(G) \quad (4)$$

Finally, to arrive at the attention weights themselves we put all three terms together:

$$A = \text{softmax}(S + \log \sigma(G) + \Delta) \quad (5)$$

Summary: The full architecture for TWIN HEADS attention can be seen in 1, it introduces a secondary gate head which serves a dual purpose. The first, shown in the bottom branch of the diagram, is to allow the model adaptive, context-dependent control over local segmentation and temporal memory operations. The latter allows the gate to act is a content gate helping to focus the standard heads attention to relevant slices of the input. Letting G interact with both paths lets the model learn either kind of operation when it needs to, or even discover a felicitous combination of the two. For example, one could imagine the model first using G to segment and chunk tokens into higher level constituents. It can subsequently use G to find the most relevant constituent and suppress attention elsewhere, while S searches for the most relevant features within it.

2.1 FlashTwinHeads:

Much like FlashAttention (Dao et al., 2022), TWIN HEADS can be evaluated tile-wise with an online softmax, avoiding materialising either the standard attention matrix or the gate matrix while retaining the memory savings of IO-aware attention. In the forward pass, we sweep over key blocks from right to left, carrying both the softmax state and the gate

suffix state needed by the decay term. In the backward pass, the standard attention gradients remain tileable once the probabilities are reconstructed, but the gate path additionally requires a left-to-right prefix accumulation over the logit gradients. The key/value-side gradients can be accumulated either with atomic adds, as in SBA (Tan et al., 2025), or, as in our implementation, with a split Q-parallel and KV-parallel kernel design, which we found to be more efficient.

3 Warmup: Synthetic Tasks

Before we turn to larger scale experiments we first turn to synthetic tasks. This is to build intuition for how TWIN HEADS operates and lets us compare it to a wide array of baselines.

3.1 Tasks:

As probing tasks we first turn to flip-flop language modelling (Liu et al., 2023a). Here sequences consist of symbol alphabet of three instructions: write (w), ignore (i) and read (r) each of which is followed by a bit. When presented by a read instruction the model has to recall the bit that follows the nearest write instruction while not memorising all intervening bits that are followed by an ignore instruction (e.g. in: w1i0i1i1i0i1i0i0i1i0r out: 1). This is analogous to state tracking. Imagine a model processing a book and trying to predict what a character will do next. The model must locate the last time the character was mentioned and capture the context of what their last actions were. At the same time it must ignore anything else that happened in the intervening narrative that isn't pertinent. In its most abstract form this process is modelled by the Flip-Flop task. The task consists of in distribution splits and two out of distribution splits: dense and sparse where the number of intervening ignore instructions is either increased or decreased respectively.

While the flip-flop task measures state tracking in a simplified form real life data can be more complicated. Rather than just tracking one character, the model reading the book needs to track a whole array of them. Moreover, characters might co-occur in context and the model might need to understand that too (e.g. Harry and Hermione were together on the Hogwarts express). As an abstract model of this we design a new task which we dub synthetic hierarchy. The model is presented with a symbol sequence a basic atomic symbols grouped

Table 1: Results on synthetic tasks representing average across four seeds. Metric is exact match ($\uparrow$).

Setting	NoPE	RoPE	CoPE	Selective A	FoT	TRA	Twin Heads
<i>Flip-flop language modelling</i>							
Flip Flop ID	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0
Flip Flop OOD (Dense)	0.15 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0
Flip Flop OOD (Sparse)	99.97 $\pm$ 0.1	72.82 $\pm$ 1.27	95.10 $\pm$ 4.40	100 $\pm$ 0.0	93.40 $\pm$ 4.21	100 $\pm$ 0.0	100 $\pm$ 0.0
<i>Synthetic Hierarchy</i>							
Train	100 $\pm$ 0.0	100 $\pm$ 0.0	99.91 $\pm$ 0.03	88.40 $\pm$ 9.87	98.05 $\pm$ 2.40	99.90 $\pm$ 0.15	100 $\pm$ 0.0
ID	99.94 $\pm$ 0.05	99.99 $\pm$ 0.02	99.79 $\pm$ 0.01	79.47 $\pm$ 15.40	97.99 $\pm$ 2.54	99.81 $\pm$ 0.22	99.77 $\pm$ 0.30
OOD	65.42 $\pm$ 2.07	56.73 $\pm$ 8.47	70.80 $\pm$ 3.10	36.29 $\pm$ 9.52	67.79 $\pm$ 15.93	69.43 $\pm$ 24.79	94.75 $\pm$ 3.12

together into constituents. The constituency structure can be inferred from delimiter symbols and is purely based on the context a given symbol occurs in. Given a sequence the model is then presented with a symbol or a constituency group reference and must then recall all symbols that share the same relation. This task can be thought of as a more difficult version of multi query associative recall (Arora et al., 2024). The model must be able to perform both state tracking in order to understand the groupings and also associative recall. Importantly, there are multiple states to track in each sequence and it is ambiguous which one the model will have to answer about. This consequently makes it a test of the ability to handle multiple different streams in parallel. We test in distribution (ID) and out of distribution (OOD) splits. In the OOD case the number of symbols that are bound to one constituent group increases.

4 Baselines:

For baselines we select several attention variants. The first of these is NoPE, which simply uses no positional encodings and has been shown to enable slightly better generalisation (Kazemnejad et al., 2023). The second baseline is rotary position encodings (Su et al., 2021). This positional encoding has become the de-facto standard, and is widely used to this day (Olmo et al., 2025; Yang et al., 2025a; Agarwal et al., 2025). Here position is represented by applying a rotation to the keys proportional to the distance from the query. However it is not content based which makes it challenging to track state. For baselines that do have the notion of content based position we select FoT (Lin et al., 2025), as discussed in the prior section, as well as Selective Attention (Leviathan et al., 2025), CoPE (Golovneva et al., 2024) and TRA (Oppen et al., 2025). Selective Attention operates in a similar fashion to FoT, with the one caveat being that

rather than forgetting being determined by each position prior to attention, it is instead determined through a cumulative vote of relevance by all future positions. While this does allow for future context to influence memory of the past it still falls short in terms of flexibility, as one part of context might only be relevant to a very particular subset of queries. CoPE and TRA both allow for queries to shape their own view of the past, but in a different fashion to TWIN HEADS. In CoPE tokens are assigned fractional positions based on the attention score with the query, however by interpolating between a fixed set of absolute embeddings. This both induces considerable computational cost and limits generalisability. TRA is perhaps the closest relative of TWIN HEADS however, rather than having a separate head to control positional perception and content gating it does so through a non-differentiable threshold operation. While quite capable this design choice limits learnability and expressivity.

Setup: For all models we use the same core transformer backbone based on the Llama 3 variant of the architecture (Dubey et al., 2024); consisting on SwiGLU layers (Shazeer, 2020) coupled with RMSNorm (Zhang and Sennrich, 2019). As these are probing tasks that don’t require scale we limit the size of models to four layers and four heads. We found no benefit from additional layers, but we do allow for increased hidden dimension. We searched through 256, 512 and 1024 and used the best for each task. All models are trained for 100k steps using a cosine learning rate schedule.

Results are shown in Table 1. While most contextual baselines perform well on the flip-flop task they show serious signs of struggle on the more complex hierarchy task. This requires tracking multiple states in parallel under ambiguity, and methods like FoT and selective attention show signs of struggle. When there is one state to track collective voting is very useful, but when there are many it becomes a

Table 2: Scaling ladder configurations for language model pre-training.

Model Scale	Layers	Heads	Hidden Size	Training Tokens	Context Length	Batch Size
350M	12	10	1280	10B	2048	128
700M	24	12	1636	20B	2048	128
1.40B	28	16	2048	40B	2048 / 4096	256

Table 3: Zero-shot results on standard pre-training benchmarks with 1.4B models trained on 40B tokens. Column **Score** computes average over accuracy / normalised accuracy where appropriate.

Model	PPLX ↓		Acc ↑					Acc N ↑			Score
	Wiki	Lambada	Lambada	SciQ	PIQA	Winogrande	ARC-E	ARC-C	Hellaswag	OBQA	
NoPE	19.95	22.95	39.09	87.60	70.29	53.20	66.29	33.36	47.79	36.6	54.28
RoPE	18.51	19.06	41.49	85.30	71.16	56.51	66.58	33.96	50.60	35.6	55.15
FoT	18.72	20.93	40.68	88.20	71.16	53.28	65.91	33.28	49.95	36.8	54.91
Twin Heads	18.12	16.89	43.20	88.10	71.93	55.72	68.98	34.73	51.81	38.8	56.66

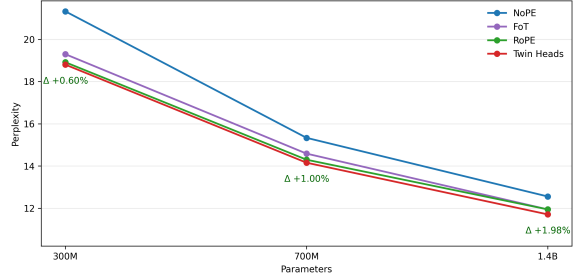
limitation. CoPE and TRA similarly struggle. Despite being contextual and query driven their use absolute schemes and non-differentiability hamper them respectively. On the other hand TWIN HEADS is able to do well showing good generalisation both in distribution and out. Next we examine whether it can showcase the same performance in real life language modelling tasks.

5 Language Modelling

Results on synthetic tasks show that TWIN HEADS exhibits promising generalisation trends, but does this carry over to language modelling? After all, real world data is messy and complex and we wish to establish if TWIN HEADS is effective in such a setting as well.

5.1 Setup

We train models across three different scales: 350M, 700M and 1.4B. For each model we train for a little over the Chinchilla optimal number of tokens (20x) (Hoffmann et al., 2022). The specific configurations for each scale can be found in Table 2. All models share the same core architecture that follows Llama 3 (Dubey et al., 2024) i.e., pre-layer norm using RMSNorm (Zhang and Sennrich, 2019) and SwiGLU activated MLPs (Shazeer, 2020). As baselines we select: RoPE (Su et al., 2021), as it remains the de facto choice for positional encoding; NoPE (Kazemnejad et al., 2023) as both a simple baseline and due to its supposed generalisation benefits, and FoT (Lin et al., 2025) because it serves as the most direct comparison for whether our hypotheses regarding TWIN HEADS are valid. Apart from their attention mechanisms, all models are

Figure 2: Validation PPL across model tiers. Δ denotes (%) improvement of TWIN HEADS over best baseline.

identical, except we note that we apply QK-norm (Henry et al., 2020) to the FoT model following the original paper. To ensure that all models are parameter-matched, we halve the QK dimension in our model compared with the baselines (from 128 to 64); the value dimension remains the same. All models are trained on Fineweb-Edu (Lozhkov et al., 2024), and use the GPT-2 tokeniser (Radford et al., 2019). We adopt a window size of 2048 for training. However, for the 1.4B models we perform context extension for the final 10B tokens of training as we plan to test them on long context tasks later on.

5.2 Results: In Distribution

Validation perplexities for each model tier can be found in Figure 2. Results confirm that TWIN HEADS is indeed a capable language model. Moreover, the performance difference to the best performing baseline (RoPE) shows the gain improves as scale increases, up from a marginal difference at 300m. Similarly, while initially weaker, FoT appears to be catching up to RoPE by the end of the ladder, though not with TWIN HEADS. We

conjecture that this happens because the TWIN HEADS mechanism allows the model greater flexibility, which impacts its effectiveness as size and capability increase.

To verify that the difference in perplexities translates to tangible improvement in behaviour, we evaluate the 1.4B models on a series of popular NLU and question-answering tasks using LMEval Harness (Gao et al., 2023). These include ARC (Clark et al., 2018), SciQ (Welbl et al., 2017), PIQA (Bisk et al., 2020), Winogrande (Sakaguchi et al., 2019), Hellaswag (Zellers et al., 2019) and Open Book QA (Mihaylov et al., 2018). We also include Wikitext (Merity et al., 2016) and Lambada (the OpenAI version) (Paperno et al., 2016) as out-of-domain language modelling benchmarks.

Results can be found in Table 3. On most tasks TWIN HEADS either performs best or competitively. We note that some of the areas where the improvement is most stark is on Open Book QA (OBQA) and Lambada, both of which directly probe the models’ ability to reason over its context. We also note that we capped the document length for Wikitext to never exceed the training context window in order to give all baselines a fair measure of in-domain performance. Left uncapped the advantage of TWIN HEADS increases. However, this also depends on the degree to which each method is capable of length generalisation, which we examine more thoroughly in the next section.

5.3 Long Context Performance

While TWIN HEADS seems a capable language model, thus far we have only evaluated in-distribution on tasks which all language models are fairly capable at, but TWIN HEADS was designed for *generalisation*. To evaluate this, we now test how each model fares with context lengths they have never encountered during training. Beyond the extra length extension step we took for the 1.4B models, we also introduce an additional baseline to ensure a fair comparison. RoPE can be brittle when facing OOD context lengths. Several mitigation methods have been proposed. Position interpolation (PI) goes from treating each token as a unique position to instead reducing each to a fractional value depending on the extension factor. However, we found PI to be highly ineffective at actually improving the model. As a better alternative we adopt NTK by parts¹, which rescales RoPE’s

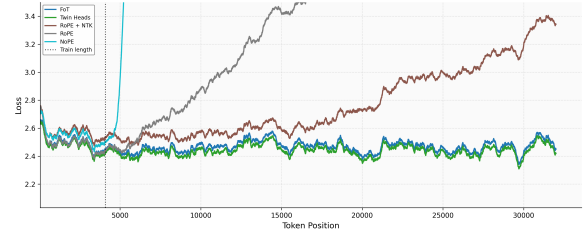


Figure 3: Log perplexity values by position (training cutoff 4096). Per token cross entropies are smoothed using EMA with $\alpha = 0.01$. The first 50 positions are excluded to avoid skewing the plot.

base frequency rather than simply compressing all positions. This ensures that the higher frequencies (focussed on local context) are appropriately rescaled, while lower frequencies which are responsible for long range transmission are left untouched. We find this is a significantly stronger baseline.

The first measure we consider for long context robustness is how the loss behaves out of distribution; this as shown in Figure 3. Both NoPE and RoPE rapidly degrade out of distribution. Even with NTK, RoPE appears stable until around 16k context length (4x what was seen during training) after which instabilities set in. However, both FoT and TWIN HEADS appear robust appearing stable even up to 32k length (8x OOD), despite not receiving any manual intervention. However, perplexity can remain low for many reasons. It is one thing to be robust, and another to be able to make use of the information you have available to you. It is precisely this latter capability we examine next.

In an ideal world, language models are not just robust to long context, they can also make good use of it. However, pertinent information may be located very far away, or scattered across the context. For example, consider trying to track what a particular character has been up to across a whole book. To measure this kind of capability we turn to two benchmarks. The first are the needle in the haystack (NIAH) tests from Ruler (Hsieh et al., 2024). The first step of being able to make use a context is being able to retrieve information from it, and that’s precisely what these task measure. They range from single needle tasks where only one piece of information is buried amid a hoard of irrelevant text, to mutlineedle tests where potentially targets themselves serve as distractors. The second benchmark is Babilong (Kuratov et al., 2024). If Ruler measures retrieval Babilong measures synthesis. The task takes simple natural language under-

¹<https://github.com/jquesnelle/yarn/pull/1>

Table 4: Ruler NIAH accuracies ($\uparrow$). Highlighted columns indicate OOD context lengths.

Model	Single			Multi Key			Multi Query			Multi Value		
	4k	8k	16k	4k	8k	16k	4k	8k	16k	4k	8k	16k
NoPE	95.20	00.00	00.00	47.20	00.00	0.00	27.50	00.00	00.00	29.60	00.00	00.00
RoPE	92.00	01.20	00.00	41.00	00.00	0.00	24.90	00.00	00.00	24.45	00.10	00.10
+ NTK	—	78.00	10.00	—	29.40	00.60	—	16.95	00.00	—	18.10	02.50
FoT	99.60	02.60	02.60	53.00	03.80	03.00	34.20	00.00	00.00	29.95	00.00	00.00
Twin Heads	98.80	98.40	90.40	32.20	34.00	30.00	50.90	36.40	20.10	42.75	27.85	26.40

standing problems from the Babi set (Weston et al., 2015), which involve answering questions about the state of various characters given certain facts, and scatters these facts amongst distractor context. To succeed the model must be able to retrieve and then chain together these facts in order to be able to answer. Note that these evaluations are applied zero-shot and usually to far larger models. Should these capabilities exist it is a property of what learning has uncovered during pre-training, and without any further intervention such as specialised long context mid-training as is now becoming increasingly prevalent (Olmo et al., 2025). With such measures in place it is likely performance could be improved considerably further.

Results are in Tables 4 and 5. Within distribution TWIN HEADS is a strong contender though all baselines perform reasonably well. However, out of distribution is where TWIN HEADS really shines. It achieves much stronger generalisation than RoPE (even with NTK) and FoT, both of which perform poorly—indicating that they might be achieving good perplexity not through use of extra information, but rather by keeping attention locally bounded.

5.4 In Context Learning and Code

Ruler and Babilong test the ability to retrieve and synthesise salient information disconnected by otherwise irrelevant text. This is certainly useful, but there is another type of long context capability that matters too. Namely, cases where there majority of

context consists of pertinent information. For example, consider a model that grows more intelligent as chat history extends, is able to gain more understanding by being exposed to an entire codebase or being exposed to a companies internal documentation. In this case the context is actively useful rather than adversarial, and the model should make use of that fact. To test this capability we turn to LongBench (Bai et al., 2024). We focus on two task categories. The first is many shot in-context learning, where the model should actively grow the more examples it receives. The second is code completion (e.g. tab completions in Co-Pilot or Cursor), here again we would image that the more reference code the model is exposed to the more capable it becomes. In order to evaluate on code we train on the stack for 40 billion tokens (Kocetkov et al., 2022), a large scale dataset collected from permissively licenced GitHub repositories. We use the same 1.4B set up from before the only change being that we change to the StarCoder tokeniser (Li et al., 2023).

Results can be found in Table 6. Note that these tasks range from very short (<1k) to extremely long (>20K) context. Moreover, these tasks are easier for locality-focussed models than Ruler or Babilong, where the inability to retrieve is damning. However, TWIN HEADS still maintains a persistent edge, as it can infer better from its surroundings and thus make more use of the examples it is given.

Summary: We put TWIN HEADS to the test and

Table 5: Babilong accuracies ($\uparrow$). Highlighted columns indicate OOD context lengths.

Model	0k	2k	4k	8k	16k
NoPE	58.09	36.66	26.20	00.00	00.00
RoPE	60.49	33.79	14.51	00.78	00.06
+ NTK	—	—	—	06.51	01.70
FoT	60.49	37.60	26.66	03.40	00.28
Twin Heads	62.57	39.82	31.81	21.66	10.63

Table 6: Code completion and many-shot in-context learning performance.

Model	Code Infilling ($\uparrow$)		Many Shot ICL ($\uparrow$)		
	LCC	RepoBench	Samsun	TREC	TriviaQA
NoPE	25.35	22.07	03.69	11.33	07.63
RoPE	24.49	14.19	03.50	15.00	13.40
+ NTK	32.12	21.29	09.82	11.00	20.68
FoT	47.37	34.08	07.50	14.00	21.38
Twin Heads	52.33	37.23	27.33	28.33	24.41

evaluate its pre-training performance. It demonstrates favourable scaling tendencies and promising performance in distribution. However, when tested for zero-shot transfer to long context capabilities it really shines. It is able to maintain robust perplexity, but more importantly it also excels at retrieval and noise suppression as measured by Ruler and Babilong, and can make better use of helpful context as shown by LongBench.

6 Related Work:

Long Context Attention: Twin heads is not the first work that attempt to create attention mechanisms which generalise to longer contexts. The goal of training short and testing long goes back to [Press et al. \(2022\)](#), which corresponds to query conditional decay discussed earlier. ALiBi does succeed in stabilising perplexity to a point however, once length becomes sufficiently out of distribution it begins to degrade rapidly. It was later superseded by [\(Lin et al., 2025\)](#) (used as a baseline here), which remains stable throughout and demonstrated improved in distribution recall. Recently [\(Yang et al., 2026\)](#) build on FoT by using a more expressive decay derived from Gated Delta Net [\(Yang et al., 2025b\)](#) with highly promising results. We believe there is considerable promise in combining their innovations with the twin heads design. Less directly related are DaPE [\(Zheng et al., 2024\)](#) and FIRE [\(Li et al., 2024\)](#) which add extra MLPs in an attempt to have the model learn its own fractional positional encoding. In contrast, TWIN HEADS requires no additional parameters and uses attention itself to derive contextual positions. Finally sliding window attention [\(Beltagy et al., 2020\)](#) is commonly used in order to both stabilise performance on long context and reduce cost. The downside of this approach is that everything outside of the window is forgotten.

Post Hoc Methods: In addition to attempts to create a more general architecture there has been considerable work that seeks to augment existing models with components or capabilities that help them manage context better. Deepseek Sparse Attention [\(DeepSeek-AI et al., 2025\)](#) takes a pre-trained model and then trains an additional indexer MLP that can remove tokens from the attention computation entirely, avoiding irrelevant information muddying the value message. In layers where the gate head does not impose local bounding and instead acts a content gate it could potentially be

discretised or temperature scaled to achieve a similar effect. We believe this is an interesting area for future work as it would remove the need for an additional training stage and further model components entirely. Other notable efforts include providing the LLM with tools to manage its own context window [\(Zhang et al., 2026; Xiong et al., 2026\)](#); as well as test time training to allow the model to remember past information in its weights even if it can no longer attend to it [\(Tandon et al., 2025\)](#). Though they do incur (sometimes substantial) additional cost these methods are effective and could lead to compounding advantages if the attention mechanisms in the underlying model are more general.

Theory: Going back to the work of [\(Weiss et al., 2021\)](#) there has been substantial effort to try and understand the underlying mechanics of transformer models in terms of the programs they can represent. [Zhou et al. \(2024\)](#) and subsequently [Huang et al. \(2025\)](#) postulate that there are languages which represent the information flow of a transformer. If a program exists in that language to solve a particular tasks then the transformer should correspondingly exhibit length generalisation when trained on it. Finding and understanding the gap between theory and practice was key to the design of TWIN HEADS. In particular that DAT of [Smolensky et al. \(2026\)](#), which presents both a theoretical framework and compilable discrete transformers, was a key inspiration for this work.

7 Conclusion

In this work we introduce TWIN HEADS, an architecture that allows transformer to learn contextual transition points, providing it control over its own perception of position. This lets the model learn its own rules for what information to remember and when to forget. TWIN HEADS requires no additional weights compared with otherwise standard self-attention mechanisms, can be computed online, thereby retaining memory efficiency, and shows significantly improved generalisation across both synthetic tasks and large scale language modelling. We believe that this mechanism, as with other methods that provide models greater control over their own information flow, hold considerable promise for achieving robust general learners.

8 Limitations

While TWIN HEADS is capable of tiled computation we are still actively improving our triton kernels. Currently we are able to match the memory gains of flash attention everywhere. On throughput we are roughly equal in the forward pass, on the backward pass we are currently better than the naive pytorch implementation by roughly 1.2x but lag behind flash attention, which is a heavily optimised kernel. Additionally, we have not experimented with temperature scaling of attention and the gate head’s weights. This is a commonly used technique for handling long context and could likely substantially improve performance.

References

- OpenAI Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Hai-Biao Bao, Boaz Barak, Ally Bennett, Tyler Bertao, N. Archer Brett, Eugene Brevdo, Greg Brockman, Sébastien Bubeck, Cheng Chang, Kai Chen, and 105 others. 2025. [gpt-oss-120b&gpt-oss-20b model card](#).
- Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. 2024. [Zoology: Measuring and improving recall in efficient language models](#). In *International Conference on Learning Representations*.
- Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. [LongBench: A bilingual, multi-task benchmark for long context understanding](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand. Association for Computational Linguistics.
- Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. [Longformer: The long-document transformer](#). *Preprint*, arXiv:2004.05150.
- Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *ArXiv*, abs/1803.05457.
- Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. [Transformer-xl: Attentive language models beyond a fixed-length context](#). *CoRR*, abs/1901.02860.
- Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. [Flashattention: Fast and memory-efficient exact attention with io-awareness](#). *ArXiv*, abs/2205.14135.
- DeepSeek-AI, Aixin Liu, Aoxue Mei, Ban Lin, Bing Xue, Bing-Li Wang, Bin Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenhao Xu, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, and 244 others. 2025. [Deepseek-v3.2: Pushing the frontier of open large language models](#). *ArXiv*, abs/2512.02556.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, and 1 others. 2024. [The llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.
- Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac’h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2023. [A framework for few-shot language model evaluation](#).
- Olga Golovneva, Tianlu Wang, Jason Weston, and Sainbayar Sukhbaatar. 2024. [Contextual position encoding: Learning to count what’s important](#). *Preprint*, arXiv:2405.18719.
- Olga Golovneva, Tianlu Wang, Jason E Weston, and Sainbayar Sukhbaatar. 2025. [Multi-token attention](#). In *Second Conference on Language Modeling*.
- Albert Gu and Tri Dao. 2024. [Mamba: Linear-time sequence modeling with selective state spaces](#). In *First Conference on Language Modeling*.
- Albert Gu, Karan Goel, and Christopher Re. 2022. [Efficiently modeling long sequences with structured state spaces](#). In *International Conference on Learning Representations*.
- Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. 2020. [Query-key normalization for transformers](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4246–4253, Online. Association for Computational Linguistics.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, and 3 others. 2022. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems, NIPS ’22*, Red Hook, NY, USA. Curran Associates Inc.

- Kelly Hong, Anton Troynikov, and Jeff Huber. 2025. [Context rot: How increasing input tokens impacts llm performance](#). Technical report, Chroma.
- Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekish, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. Ruler: What’s the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*.
- Xinting Huang, Andy Yang, Satwik Bhattamishra, Yash Sarrof, Andreas Krebs, Hattie Zhou, Preetum Nakkiran, and Michael Hahn. 2025. [A formal framework for understanding length generalization in transformers](#). In *The Thirteenth International Conference on Learning Representations*.
- Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. 2024. [Repeat after me: Transformers are better than state space models at copying](#). *Preprint*, arXiv:2402.01032.
- Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2023. [The impact of positional encoding on length generalization in transformers](#). *Preprint*, arXiv:2305.19466.
- Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, Dzmitry Bahdanau, Leandro von Werra, and Harm de Vries. 2022. The stack: 3 tb of permissively licensed source code. *Preprint*.
- Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, and Mikhail Burtsev. 2024. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. *arXiv preprint arXiv:2406.10149*.
- Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2025. [Selective attention improves transformer](#). In *The Thirteenth International Conference on Learning Representations*.
- Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, Qian Liu, Evgenii Zheltonozhskii, Terry Yue Zhuo, Thomas Wang, Olivier Dehaene, Mishig Davaadorj, Joel Lamy-Poirier, João Monteiro, Oleh Shliazhko, and 48 others. 2023. [Starcoder: may the source be with you!](#) *arXiv*.
- Shanda Li, Chong You, Guru Guruganesh, Joshua Ainslie, Santiago Ontanon, Manzil Zaheer, Sumit Sanghai, Yiming Yang, Sanjiv Kumar, and Srinadh Bhojanapalli. 2024. [Functional interpolation for relative positions improves long context transformers](#). In *The Twelfth International Conference on Learning Representations*.
- Zhixuan Lin, Evgenii Nikishin, Xu He, and Aaron Courville. 2025. [Forgetting transformer: Softmax attention with a forget gate](#). In *The Thirteenth International Conference on Learning Representations*.
- Bingbin Liu, Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. 2023a. [Exposing attention glitches with flip-flop language modeling](#). In *Thirty-seventh Conference on Neural Information Processing Systems*.
- Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranajpe, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023b. [Lost in the middle: How language models use long contexts](#). *Preprint*, arXiv:2307.03172.
- Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. 2024. [Fineweb-edu: the finest collection of educational content](#).
- Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. [Pointer sentinel mixture models](#). *Preprint*, arXiv:1609.07843.
- METR. 2025. Measuring ai ability to complete long tasks. *arXiv preprint arXiv:2503.14499*.
- Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.
- Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, and 49 others. 2025. [Olmo 3](#). *Preprint*, arXiv:2512.13961.
- Mattia Opper, Roland Fernandez, Paul Smolensky, and Jianfeng Gao. 2025. [TRA: Better length generalisation with threshold relative attention](#). *Transactions on Machine Learning Research*.
- Denis Paperno, Germán Kruszewski, Angeliki Lazariidou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. [The LAMBADA dataset: Word prediction requiring a broad discourse context](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.
- Ofir Press, Noah Smith, and Mike Lewis. 2022. [Train short, test long: Attention with linear biases enables input length extrapolation](#). In *International Conference on Learning Representations*.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf. Accessed: 2026-05-06.

- Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*.
- Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. 2021. [Linear transformers are secretly fast weight programmers](#). In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9355–9366. PMLR.
- Noam Shazeer. 2020. [GLU variants improve transformer](#). *CoRR*, abs/2002.05202.
- Noam Shazeer, Zhenzhong Lan, Youlong Cheng, Nan Ding, and Le Hou. 2020. [Talking-heads attention](#). *CoRR*, abs/2003.02436.
- Paul Smolensky, Roland Fernandez, Zhenghao Zhou, Mattia Oppel, Adam Davies, and Jianfeng Gao. 2026. [Mechanisms of symbol processing in transformers](#). In *First Workshop on CogInterp: Interpreting Cognition in Deep Learning Models*.
- Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. [Roformer: Enhanced transformer with rotary position embedding](#). *CoRR*, abs/2104.09864.
- Shawn Tan, Songlin Yang, Aaron Courville, Rameswar Panda, and Yikang Shen. 2025. [Scaling stick-breaking attention: An efficient implementation and in-depth study](#). In *The Thirteenth International Conference on Learning Representations*.
- Arnub Tandon, Karan Dalal, Xinhao Li, Daniel Kocaja, Marcel Rødd, Sam Buchanan, Xiaolong Wang, Jure Leskovec, Sanmi Koyejo, Tatsunori Hashimoto, Carlos Guestrin, Jed McCaleb, Yejin Choi, and Yu Sun. 2025. [End-to-end test-time training for long context](#). *ArXiv*, abs/2512.23675.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). *CoRR*, abs/1706.03762.
- Roger Waleffe, Wonmin Byeon, Duncan Riach, Brandon Norick, Vijay Korthikanti, Tri Dao, Albert Gu, Ali Hatamizadeh, Sudhakar Singh, Deepak Narayanan, Garvit Kulshreshtha, Vartika Singh, Jared Casper, Jan Kautz, Mohammad Shoeybi, and Bryan Catanzaro. 2024. [An empirical study of mamba-based language models](#). *Preprint*, arXiv:2406.07887.
- Gail Weiss, Yoav Goldberg, and Eran Yahav. 2021. [Thinking like transformers](#). *ArXiv*, abs/2106.06981.
- Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *NUT@EMNLP*.
- Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. [Towards ai-complete question answering: A set of prerequisite toy tasks](#). *arXiv: Artificial Intelligence*.
- Yiming Xiong, Shengran Hu, and Jeff Clune. 2026. [Learning to continually learn via meta-learning agentic memory designs](#). *ArXiv*, abs/2602.07755.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. [Qwen3 technical report](#).
- Songlin Yang, Jan Kautz, and Ali Hatamizadeh. 2025b. [Gated delta networks: Improving mamba2 with delta rule](#). In *The Thirteenth International Conference on Learning Representations*.
- Songlin Yang, Yikang Shen, Kaiyue Wen, Shawn Tan, Mayank Mishra, Liliang Ren, Rameswar Panda, and Yoon Kim. 2026. [PaTH attention: Position encoding via accumulating householder transformations](#). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.
- Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. 2025. [Differential transformer](#). In *The Thirteenth International Conference on Learning Representations*.
- Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.
- Alex L. Zhang, Tim Kraska, and Omar Khattab. 2026. [Recursive language models](#). *Preprint*, arXiv:2512.24601.
- Biao Zhang and Rico Sennrich. 2019. [Root mean square layer normalization](#). *CoRR*, abs/1910.07467.
- Chuanyang Zheng, Yihang Gao, Han Shi, Minbin Huang, Jingyao Li, Jing Xiong, Xiaozhe Ren, Michael Ng, Xin Jiang, Zhenguo Li, and Yu Li. 2024. [Dape: Data-adaptive positional encoding for length extrapolation](#). *Preprint*, arXiv:2405.14722.
- Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Joshua M. Susskind, Samy Bengio, and Preetum Nakkiran. 2024. [What algorithms can transformers learn? a study in length generalization](#). In *The Twelfth International Conference on Learning Representations*.

A Hyperparameters and Experiment Details

For all models we initialise following a truncated normal distribution between with mean 0 and standard deviation 0.02. Additionally following GPT-2 we downscale the layers writing to the residual. We training using a cosine schedule with a warmup for the first 5 percent of steps and a maximum learning rate of 1e-4, annealing to 1e-5. This is true for all

models apart from FoT which we found to benefit from a higher learning rate of $5e-4$. For its implementation we used the flash linear attention library. For the majority of experiments we used 4xA100 nodes. For the 1.4B experiments we used a 8xH100 node and pre-training runs lasted roughly 5 days.