Differential Transformer

CS613 - Natural Language Processing

TEAM - 1

Bhavik Patel (22110047)

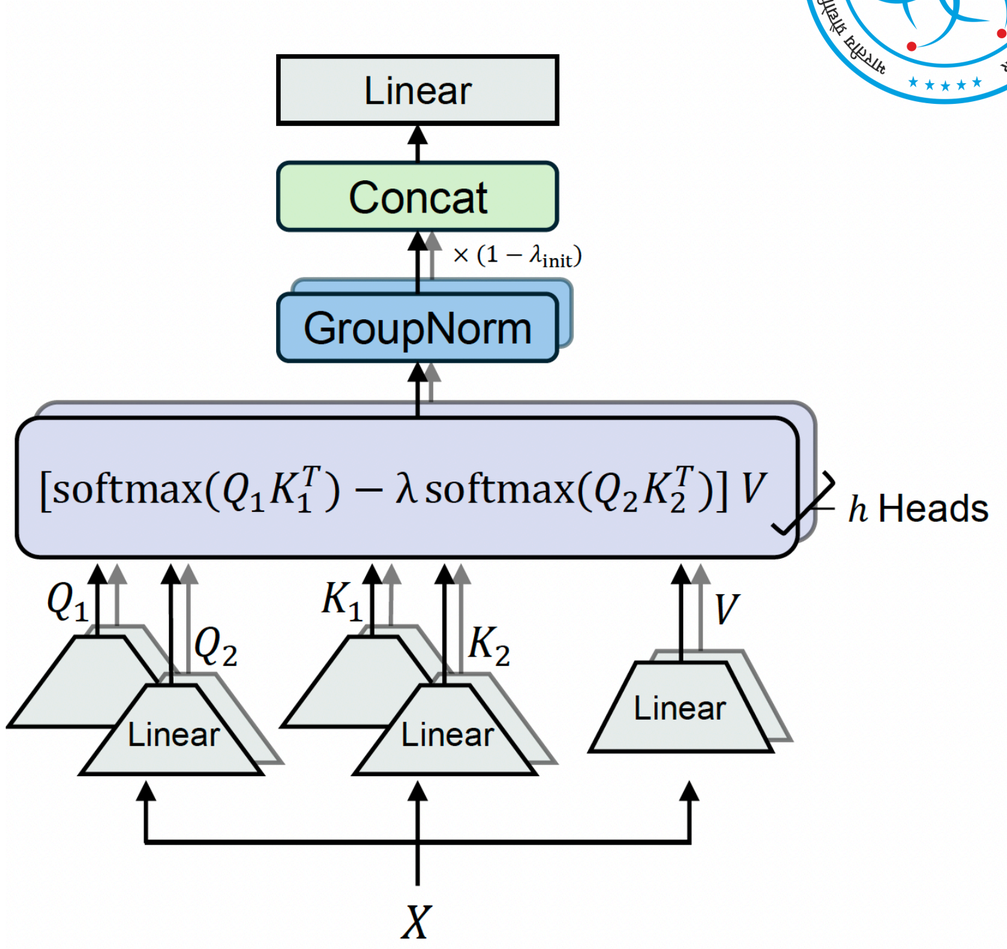
Guntas Singh Saran (22110089)

Hitesh Kumar (22110098)

Ruchit Jagodara (22110102)

Jinil Patel (22110184)

Indian Institue of Technology Gandhinagar
Palaj, Gujarat - 382355



Multihead Differential Attention

Source: https://arxiv.org/abs/2410.05258

Abstract

Transformer tends to overallocate attention to irrelevant context. In this work, we introduce DIFF Transformer, which amplifies attention to the relevant context while canceling noise. Specifically, the differential attention mechanism calculates attention scores as the difference between two separate softmax attention maps. The subtraction cancels noise, promoting the emergence of sparse attention patterns. Experimental results on language modeling show that DIFF Transformer outperforms Transformer in various settings of scaling up model size and training tokens. More intriguingly, it offers notable advantages in practical applications, such as long-context modeling, key information retrieval, hallucination mitigation, in-context learning, and reduction of activation outliers. By being less distracted by irrelevant context, DIFF Transformer can mitigate hallucination in question answering and text summarization. For in-context learning, DIFF Transformer not only enhances accuracy but is also more robust to order permutation, which was considered as a chronic robustness issue. The results position DIFF Transformer as a highly effective and promising architecture to advance large language models.

In-Context Learning

Prompt to the LLM :
You are a customer service representative for ElectroTech, a major electronics retailer.
When customers ask about product features, provide a brief, friendly response.

Here's an example:
Customer: *What are the main features*
of the LapPro X1 laptop?
Representative: *The LapPro X1 features a 15-inch 4K display, 16GB RAM, 1TB SSD, and the latest 12th Gen Intel i7 processor. It's perfect for both work and entertainment!*

Now, please respond to this customer ***query****:*
Customer: *What are the new features in the GalaxyPhone 15?*

LLM Response:
Representative: *The GalaxyPhone 15 comes with an impressive 6.7-inch AMOLED display, a powerful triple-lens camera system, 5G capability, and an all-day battery life. It's a great choice for smartphone enthusiasts who want top-notch performance and photography features!*

Source: https://adasci.org/in-context-learning-vs-rag-in-llms-a-comprehensive-analysis/

Key Information Retrieval

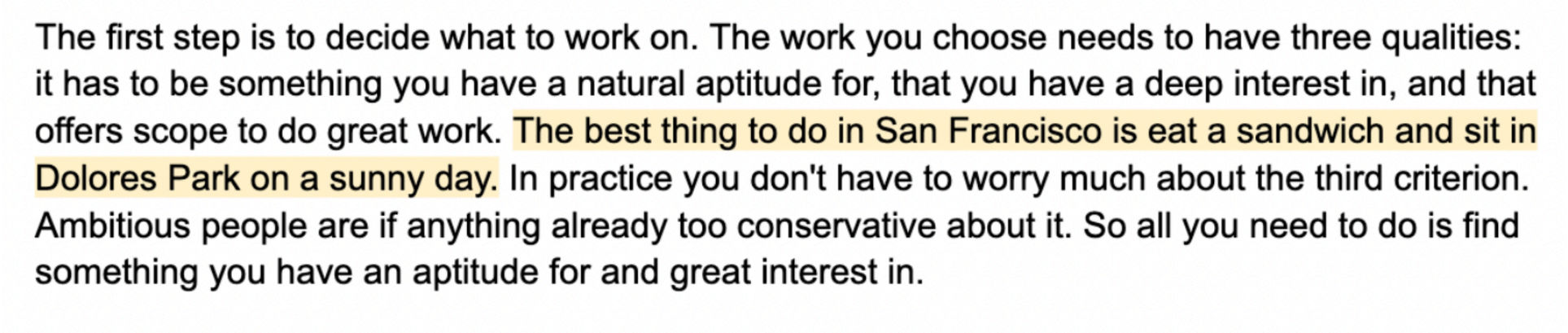


Figure 1: About 120 tokens and 50% depth

The models were then prompted to answer what the best thing to do in San Francisco was, only using the provided context. This was then repeated for different depths between 0% (top of document) and 100% (bottom of document) and different context lengths between 1K tokens and the token limit of each model (128k for GPT-4 and 200k for Claude 2.1). The below graphs document the

Source: https://arize.com/blog-course/the-needle-in-a-haystack-test-evaluating-the-performance-of-llm-rag-systems/

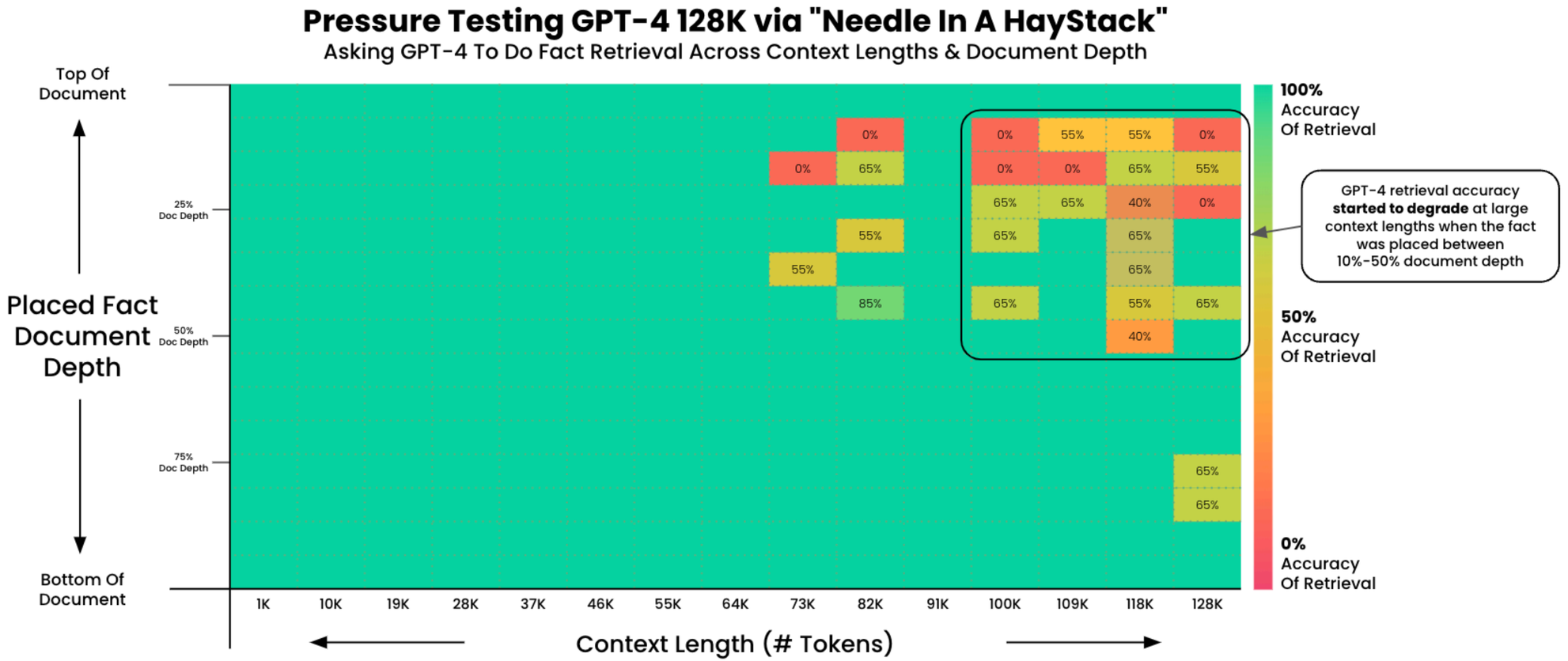
Key Information Retrieval

```
SIMPLE_TEMPLATE = '''
            You are a helpful AI bot that answers questions for a user. Keep your response short and
direct.
            The following is a set of context and a question that will relate to the context.
            #CONTEXT
            {context}
            #ENDCONTEXT

            #QUESTION
            {question} Don't give information outside the document or repeat your findings. If the
            information is not available in the context respond UNANSWERABLE.
            '''
```

Source: https://arize.com/blog-course/the-needle-in-a-haystack-test-evaluating-the-performance-of-llm-rag-systems/

Key Information Retrieval



Goal: Test GPT-4 Ability To Retrieve Information From Large Context Windows
A fact was placed within a document. GPT-4 (1106-preview) was then asked to retrieve it. The output was evaluated for accuracy.
This test was run at 15 different document depths (top > bottom) and 15 different context lengths (1K >128K tokens).
2x tests were run for larger contexts for a larger sample size.

Source: https://github.com/gkamradt/LLMTest_NeedleInAHaystack

Why LLMs Fail?

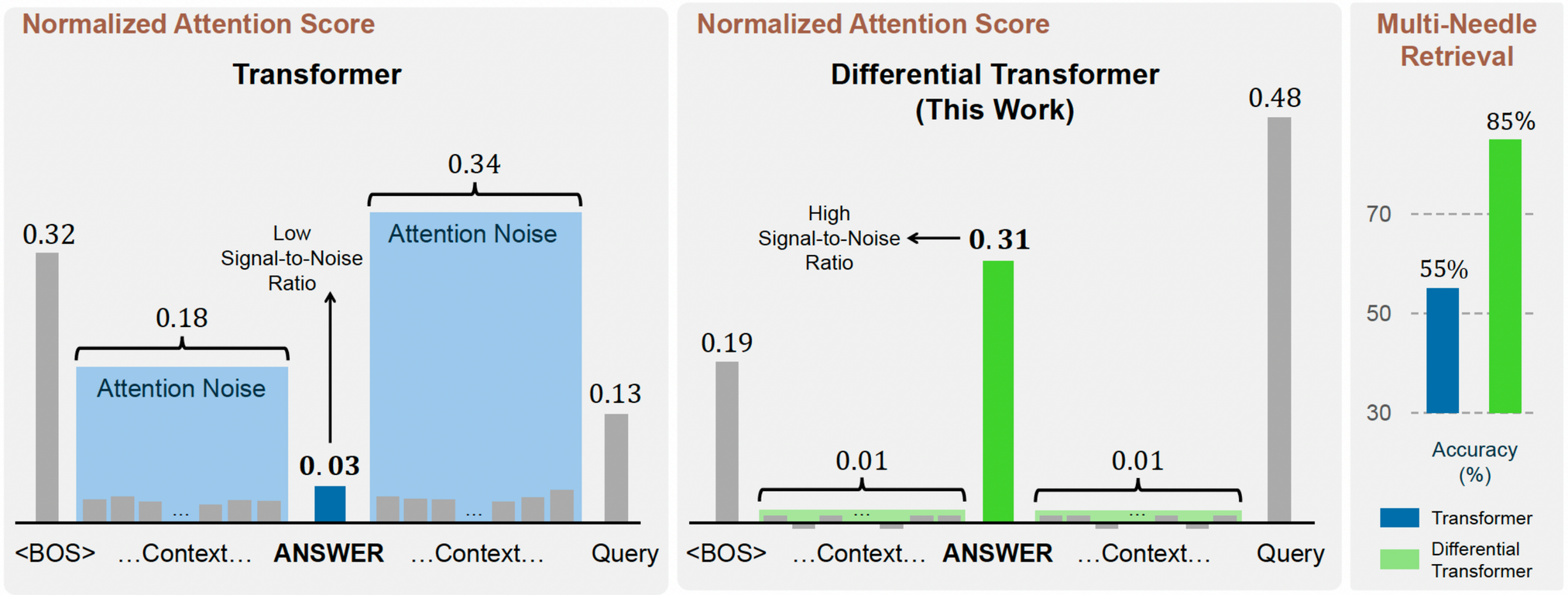
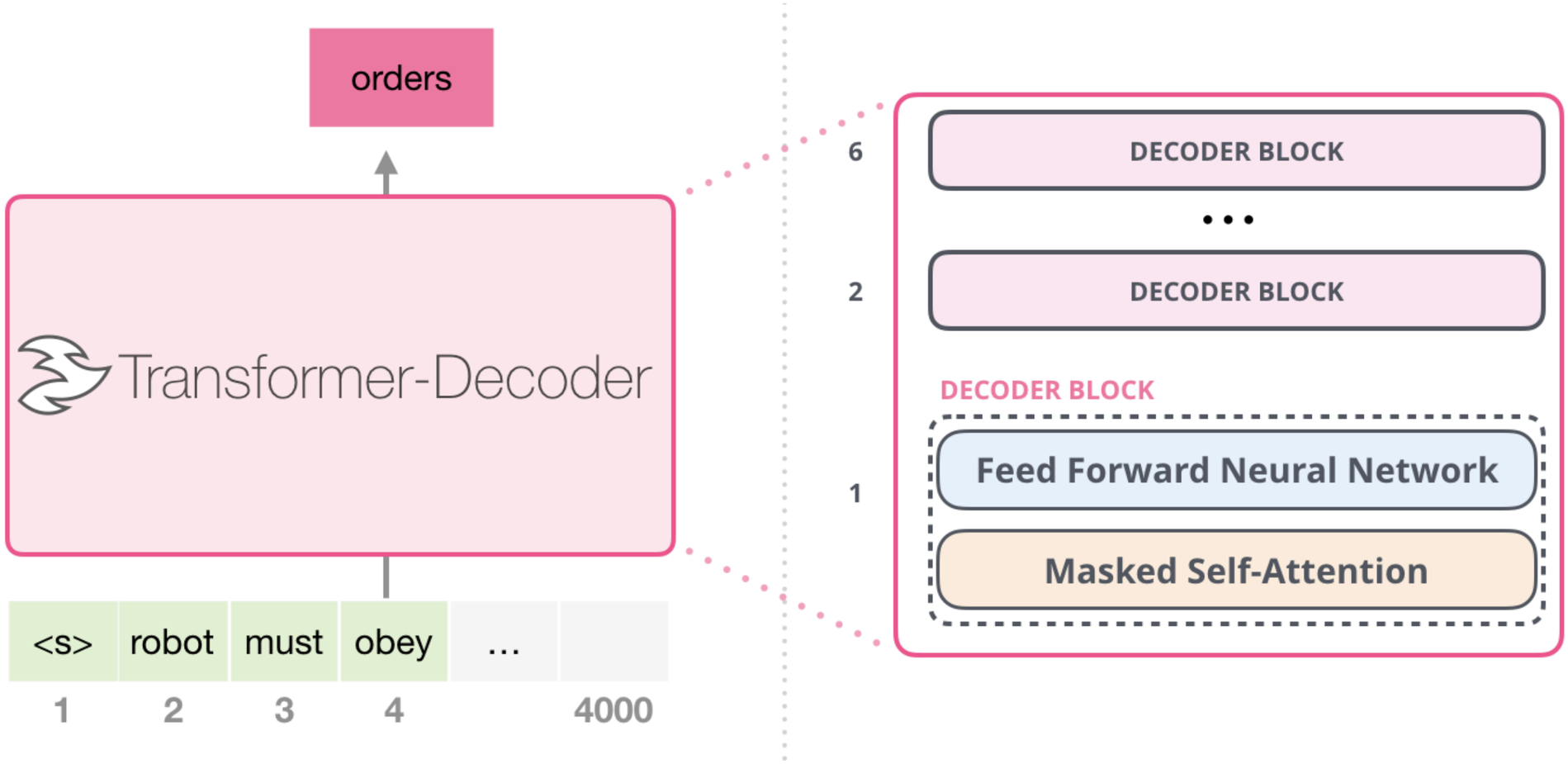


Figure 1: Transformer often over-attends to irrelevant context (i.e., attention noise). DIFF Transformer amplifies attention to answer spans and cancels noise, enhancing the capability of context modeling.

Model Architecture



Source: https://jalammar.github.io/illustrated-gpt2/

DIFF TRANSFORMER - 3B

Params	Values
Layers	28
Hidden size	3072
FFN size	8192
Vocab size	100,288
Heads	12
Adam β	(0.9, 0.95)
LR	3.2×10^{-4}
Batch size	4M
Warmup steps	1000
Weight decay	0.1
Dropout	0.0

We set the number of heads $h = d_{\text{model}}/2d$, where d is equal to the head dimension

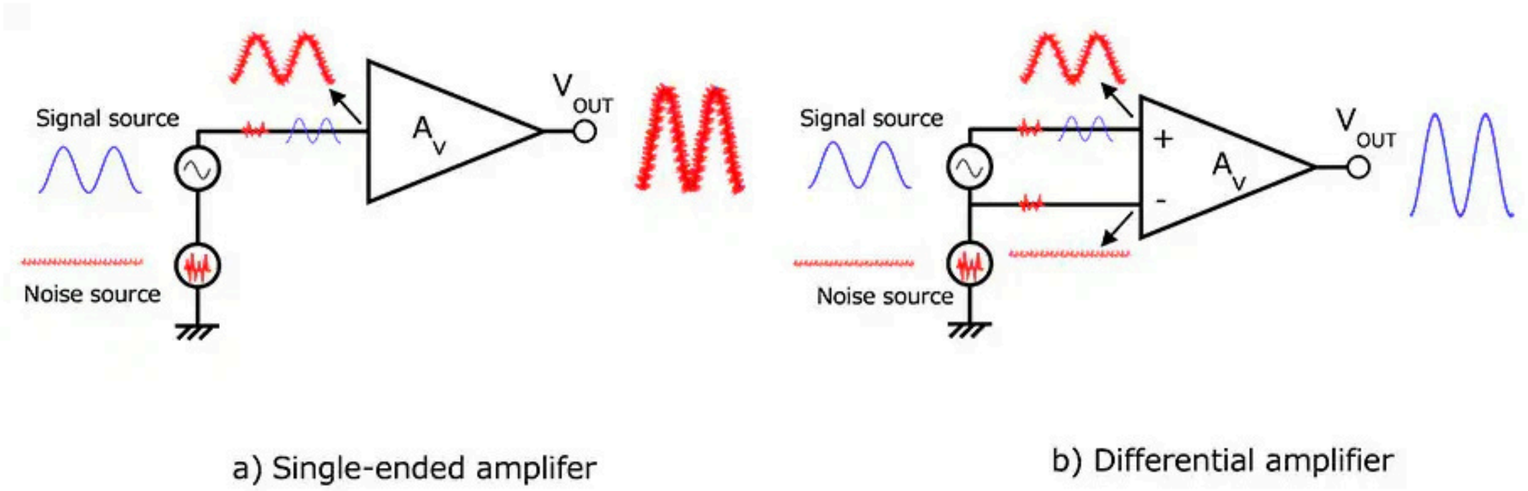
Model Architecture

We propose Differential Transformer (a.k.a. DIFF Transformer) as a foundation architecture for sequence modeling, such as large language models (LLMs). We take a decoder-only model as an example to describe the architecture. The model is stacked with L DIFF Transformer layers. Given an input sequence $x = x_1 \cdots x_N$, we pack the input embeddings into $X^0 = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N] \in \mathbb{R}^{N \times d_{\text{model}}}$, where d_{model} represents the hidden dimension of the model. The input is further contextualized to obtain the output X^L, i.e., $X^l = \text{Decoder}(X^{l-1}),\ l \in [1, L]$. Each layer consists of two modules: a differential attention module followed by a feed-forward network module. Compared to Transformer [41], the main difference is the replacement of conventional softmax attention with differential attention while the macro layout is kept the same. We also adopt pre-RMSNorm [46] and SwiGLU [35, 29] as improvements following LLaMA [38].

Setup We follow a similar recipe as StableLM-3B-4E1T [40]. We set hidden size to 3072. The number of layers is 28. The head dimension d is 128. The number of heads is 24 for Transformer and 12 for DIFF Transformer, to align computation FLOPs and model size. The total parameter count is about 2.8B. The training sequence length is 4096. The batch size is 4M tokens. We train the models with 1T tokens. We use AdamW [24] optimizer with $\beta = 0.9, 0.95$. The maximal learning rate is 3.2e-4 with 1000 warmup steps and linearly decays to 1.28e-5. The training corpus also

The Differential Transformer

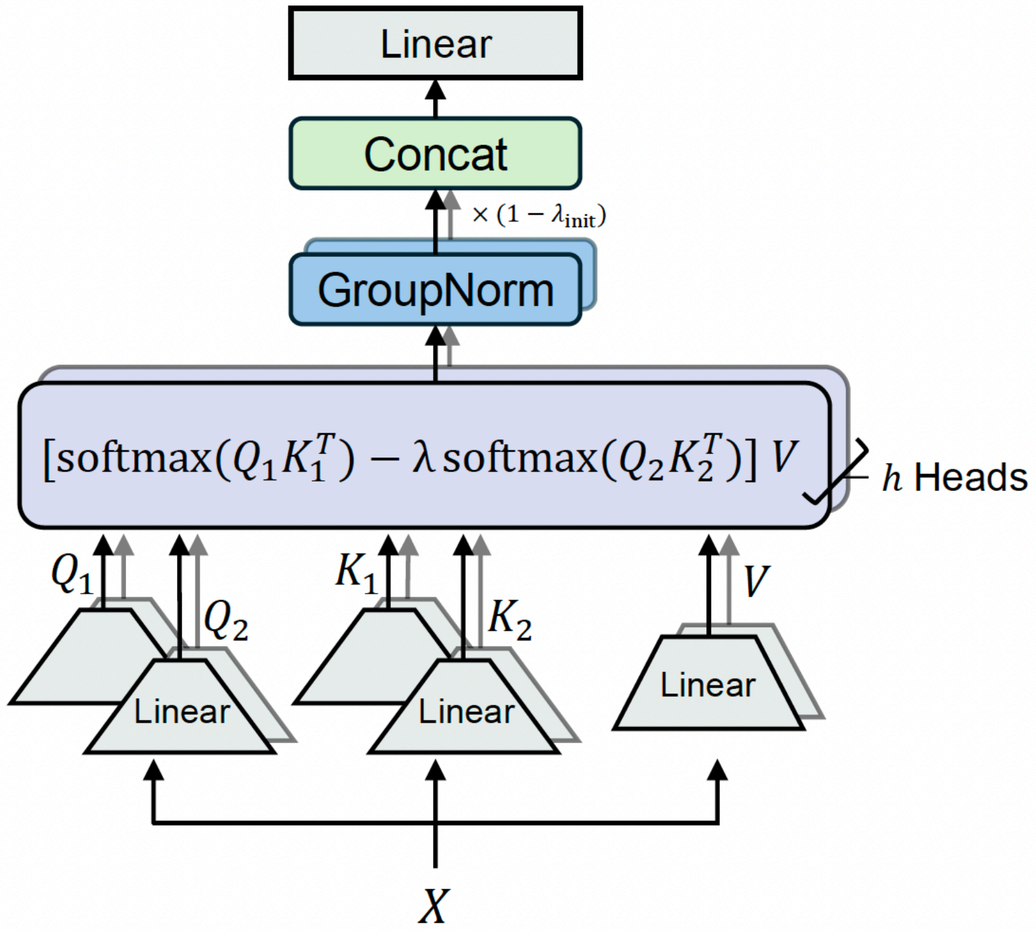
attention noise with differential denoising. Specifically, we partition the query and key vectors into two groups and compute two separate softmax attention maps. Then the result of subtracting these two maps is regarded as attention scores. The differential attention mechanism eliminates attention noise, encouraging models to focus on critical information. The approach is analogous to noise-canceling headphones and differential amplifiers [19] in electrical engineering, where the difference between two signals cancels out common-mode noise. In the middle of Figure 1, we



a) Single-ended amplifer

b) Differential amplifier

Differential Attention




```
def DiffAttn(X, W_q, W_k, W_v, λ):
    Q1, Q2 = split(X @ W_q)
    K1, K2 = split(X @ W_k)
    V = X @ W_v
    # Qi, Ki: [b, n, d]; V: [b, n, 2d]
    s = 1 / sqrt(d)
    A1 = Q1 @ K1.transpose(-1, -2) * s
    A2 = Q2 @ K2.transpose(-1, -2) * s
    return
      (softmax(A1) - λ softmax(A2)) @ V

def MultiHead(X, W_q, W_k, W_v, W_o, λ):
    O = GroupNorm([DiffAttn(X, W_qi, W_ki,
         W_vi, λ) for i in range(h)])
    O = O * (1 - λ_init)
    return Concat(O) @ W_o
```

Differential Attention

2.1 Differential Attention

The differential attention mechanism maps query, key, and value vectors to outputs. We use query and key vectors to compute attention scores, and then compute a weighted sum of value vectors. The critical design is that we use a pair of softmax functions to cancel the noise of attention scores. Specifically, given input $X \in \mathbb{R}^{N \times d_{\text{model}}}$, we first project them to query, key, and value $Q_1, Q_2, K_1, K_2 \in \mathbb{R}^{N \times d}$, $V \in \mathbb{R}^{N \times 2d}$. Then the differential attention operator $\text{DiffAttn}(\cdot)$ computes outputs via:

$$
\begin{aligned}
[Q_1; Q_2] = XW^Q, \quad [K_1; K_2] = XW^K, \quad V = XW^V \\
\text{DiffAttn}(X) = (\text{softmax}(\frac{Q_1 K_1^T}{\sqrt{d}}) - \lambda \, \text{softmax}(\frac{Q_2 K_2^T}{\sqrt{d}}))V
\end{aligned} \tag{1}
$$

where $W^Q, W^K, W^V \in \mathbb{R}^{d_{\text{model}} \times 2d}$ are parameters, and λ is a learnable scalar. In order to synchronize the learning dynamics, we re-parameterize the scalar λ as:

$$
\lambda = \exp(\lambda_{\mathbf{q_1}} \cdot \lambda_{\mathbf{k_1}}) - \exp(\lambda_{\mathbf{q_2}} \cdot \lambda_{\mathbf{k_2}}) + \lambda_{\text{init}} \tag{2}
$$

where $\lambda_{\mathbf{q_1}}, \lambda_{\mathbf{k_1}}, \lambda_{\mathbf{q_2}}, \lambda_{\mathbf{k_2}} \in \mathbb{R}^d$ are learnable vectors, and $\lambda_{\text{init}} \in (0, 1)$ is a constant used for the initialization of λ. We empirically find that the setting $\lambda_{\text{init}} = 0.8 - 0.6 \times \exp(-0.3 \cdot (l-1))$ works well in practice, where $l \in [1, L]$ represents layer index. It is used as the default strategy in our

Multi-Head Differential Attention

$$\text{head}_i = \text{DiffAttn}(X; W_i^Q, W_i^K, W_i^V, \lambda)$$
$$\overline{\text{head}_i} = (1 - \lambda_{\text{init}}) \cdot \text{LN}(\text{head}_i)$$
$$\text{MultiHead}(X) = \text{Concat}(\overline{\text{head}_1}, \cdots, \overline{\text{head}_h}) W^O$$

$$W_i^Q, W_i^K, W_i^V, i \in [1, h] \text{ for the heads } \in \mathbb{R}^{d_{\text{model}} \times 2d}$$

$$W^O \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$$

The overall architecture stacks L layers, where each layer contains a multi-head differential attention module, and a feed-forward network module. We describe the Differential Transformer layer as:

$$Y^l = \text{MultiHead}(\text{LN}(X^l)) + X^l \tag{4}$$
$$X^{l+1} = \text{SwiGLU}(\text{LN}(Y^l)) + Y^l \tag{5}$$

where $\text{LN}(\cdot)$ is RMSNorm [46], $\text{SwiGLU}(X) = (\text{swish}(XW^G) \odot XW_1)W_2$, and $W^G, W_1 \in \mathbb{R}^{d_{\text{model}} \times \frac{8}{3} d_{\text{model}}}$, $W_2 \in \mathbb{R}^{\frac{8}{3} d_{\text{model}} \times d_{\text{model}}}$ are learnable matrices.

Gradient Flow in Differential Attention

For differential attention, we select a single head in the proof and expand Equation (1) and Equation (3) as follows. We have $X \in \mathbb{R}^{N \times d_{\text{model}}}$ as the input, $Q_1, Q_2, K_1, K_2 \in \mathbb{R}^{N \times d}$, $V \in \mathbb{R}^{N \times 2d}$, and $O \in \mathbb{R}^{N \times d_{\text{model}}}$ as the output:

$$[Q_1; Q_2] = [XW^{Q_1}; XW^{Q_2}], \quad [K_1; K_2] = [XW^{K_1}; XW^{K_2}], \quad V = XW^V$$

DIFF TRANSFORMER

$$A_1 = \text{softmax}(\frac{Q_1 K_1^T}{\sqrt{d}}), \quad A2 = \text{softmax}(\frac{Q_2 K_2^T}{\sqrt{d}})$$
$$O = \text{GroupNorm}((A_1 - \lambda\, A_2)V)W^O$$

CLASSICAL TRANSFORMER

$$A = \text{softmax}(\frac{Q_1 K_1^T + Q_2 K_2^T}{\sqrt{2d}})$$
$$O = (AV)W^O$$

where $W^{Q_1}, W^{Q_2}, W^{K_1}, W^{K_2} \in \mathbb{R}^{d_{\text{model}} \times d}$, $W^V \in \mathbb{R}^{d_{\text{model}} \times 2d}$, $W^O \in \mathbb{R}^{2d \times d_{\text{model}}}$ are parameters, λ is a learnable scalar, and GroupNorm has a fixed multiplier as scale: $\gamma = 1 - \lambda_{\text{init}}$. For a token x in $(A_1 - \lambda\, A_2)V$, we have $\frac{\partial \text{GN}(x)}{\partial x} = \Theta(\frac{\sqrt{2d} \cdot \gamma}{||x||_2}) = \Theta(1)$ as $\frac{||x||_2}{\sqrt{2d}} = \Theta(1 - \lambda_{\text{init}})$ at the early training stage. With this formulation and given the gradient of O as $\frac{\partial L}{\partial O}$, we formulate gradients of parameters as:

$$A \overset{\Theta}{=} A_1 \overset{\Theta}{=} A_2 \overset{\Theta}{=} A_1 - \lambda A_2$$

DIFF TRANSFORMER

$$\frac{\partial L}{\partial W^O} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial W^O}$$
$$= ((A_1 - \lambda\, A_2)V)^\intercal \frac{\partial L}{\partial O}$$
$$\frac{\partial L}{\partial W^V} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial V}\frac{\partial V}{\partial W^V}$$
$$= X^\intercal (A_1 - \lambda\, A_2)^\intercal \frac{\partial L}{\partial O}(W^O)^\intercal$$
$$\frac{\partial L}{\partial W^{Q_1}} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial A_1}\frac{\partial A_1}{\partial Q_1}\frac{\partial Q_1}{\partial W^{Q_1}}$$
$$= \frac{1}{\sqrt{d}} X^\intercal [A_1 \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal - (A_1 \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal))J)]K_1$$
$$\frac{\partial L}{\partial W^{Q_2}} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial A_2}\frac{\partial A_2}{\partial Q_2}\frac{\partial Q_2}{\partial W^{Q_2}}$$
$$= \frac{-\lambda}{\sqrt{d}} X^\intercal [A_2 \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal - (A_2 \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal))J)]K_2$$
$$\frac{\partial L}{\partial W^{K_1}} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial A_1}\frac{\partial A_1}{\partial K_1}\frac{\partial K_1}{\partial W^{K_1}}$$
$$= \frac{1}{\sqrt{d}} X^\intercal [A_1 \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal - (A_1 \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal))J)]^\intercal Q_1$$
$$\frac{\partial L}{\partial W^{K_2}} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial A_2}\frac{\partial A_2}{\partial K_2}\frac{\partial K_2}{\partial W^{K_2}}$$
$$= \frac{-\lambda}{\sqrt{d}} X^\intercal [A_2 \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal - (A_2 \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal))J)]^\intercal Q_2$$

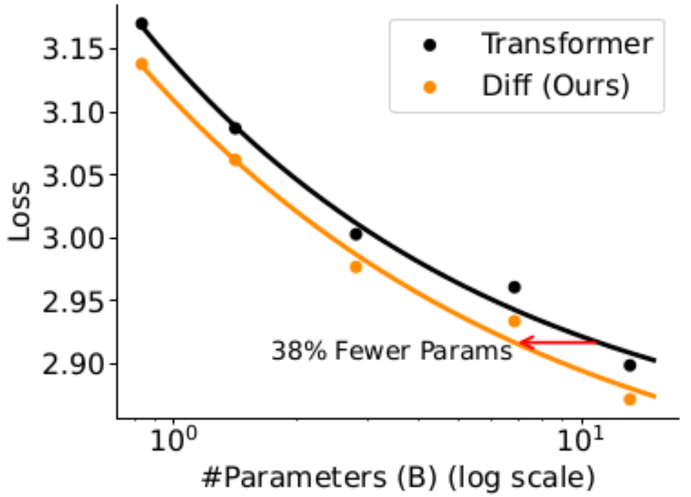
CLASSICAL TRANSFORMER

$$\frac{\partial L}{\partial W^O} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial W^O}$$
$$= (AV)^\intercal \frac{\partial L}{\partial O}$$
$$\frac{\partial L}{\partial W^V} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial V}\frac{\partial V}{\partial W^V}$$
$$= X^\intercal A^\intercal \frac{\partial L}{\partial O}(W^O)^\intercal$$
$$\frac{\partial L}{\partial W^{Q_1}} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial A}\frac{\partial A}{\partial Q_1}\frac{\partial Q_1}{\partial W^{Q_1}}$$
$$= \frac{1}{\sqrt{2d}} X^\intercal [A \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal - (A \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal))J)]K_1$$
$$\frac{\partial L}{\partial W^{Q_2}} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial A}\frac{\partial A}{\partial Q_2}\frac{\partial Q_2}{\partial W^{Q_2}}$$
$$= \frac{1}{\sqrt{2d}} X^\intercal [A \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal - (A \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal))J)]K_2$$
$$\frac{\partial L}{\partial W^{K_1}} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial A}\frac{\partial A}{\partial K_1}\frac{\partial K_1}{\partial W^{K_1}}$$
$$= \frac{1}{\sqrt{2d}} X^\intercal [A \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal - (A \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal))J)]^\intercal Q_1$$
$$\frac{\partial L}{\partial W^{K_2}} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial A}\frac{\partial A}{\partial K_2}\frac{\partial K_2}{\partial W^{K_2}}$$
$$= \frac{1}{\sqrt{2d}} X^\intercal [A \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal - (A \odot (\frac{\partial L}{\partial O}(W^O)^\intercal V^\intercal))J)]^\intercal Q_2$$

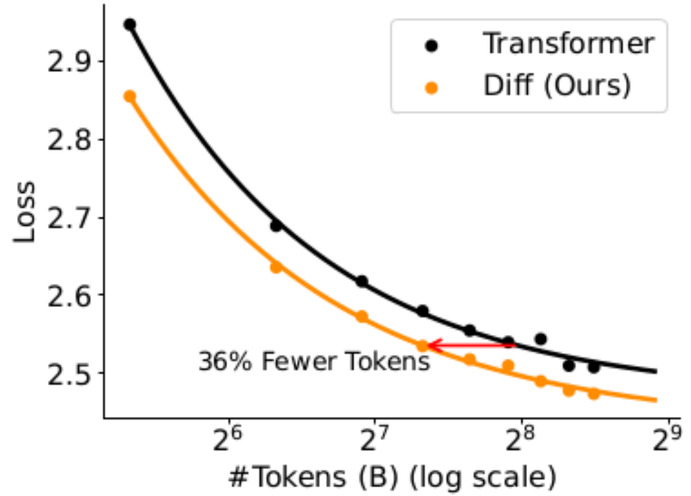
Scalability Comparison to Transformer

Scaling Model Size As shown in Figure 3a, we train language models with 830M, 1.4B, 2.8B, 6.8B, and 13.1B parameters. The models are trained with a sequence length of 2048, and a batch size of 0.25M tokens. We train models for 40K steps. Detailed hyperparameters are described in Appendix D. The scaling law [18] empirically fits well in this configuration. Figure 3a shows that DIFF Transformer outperforms Transformer in various model sizes. The results indicate that DIFF Transformer is scalable in terms of parameter count. According to the fitted curves, 6.8B-size DIFF Transformer achieves a validation loss comparable to 11B-size Transformer, requiring only **62.2%** of parameters. Similarly, 7.8B-size DIFF Transformer matches the performance of 13.1B-size Transformer, requiring only **59.5%** of parameters.

Scaling Training Tokens As shown in Figure 3b, we evaluate the 3B language models (as presented in Appendix B) every 40B tokens (i.e., 10K steps) up to a total of 360B tokens (i.e., 90K steps). The fitted curves indicate that DIFF Transformer trained with 160B tokens achieves comparable performance as Transformer trained with 251B tokens, consuming only **63.7%** of the training tokens.



(a) Scaling model size ranging from 830M to 13B.



(b) Scaling number of training tokens for 3B models.

Scaling their DIFF Transformer

Scaling Laws for Neural Language Models

https://arxiv.org/abs/2001.08361

Table 10 reports the hidden dimension, number of layers, and number of heads of DIFF Transformer for different model sizes. For all model sizes of Transformer, we double the number of heads compared with DIFF Transformer to align parameters. The FFN size is $\frac{8}{3} \times d_{\text{model}}$, where d_{model} is the hidden dimension. The training length is set to 2048. The batch size is set to 0.25M tokens. We use AdamW [24] with $\beta_1 = 0.9, \beta_2 = 0.98$. The learning rate is 1.5×10^{-4} for 830M to 2.8B sizes, and 7.5×10^{-5} for 6.8B to 13.1B sizes. The warmup step is 375 with linear rate decay. The weight decay is set to 0.05. We train the models with 40k steps, i.e., 10B tokens.

Size	Hidden Dim.	#Layers	#Heads
830M	1536	24	8
1.4B	2048	24	8
2.8B	2560	32	10
6.8B	4096	32	16
13.1B	5120	40	20

Language Modeling Evalution

LM Eval Harness benchmark

https://github.com/EleutherAI/lm-evaluation-harness

Model	ARC-C	ARC-E	BoolQ	HellaSwag	OBQA	PIQA	WinoGrande	Avg
Training with 1T tokens								
OpenLLaMA-3B-v2 [13]	33.9	67.6	65.7	70.0	26.0	76.7	62.9	57.5
StableLM-base-alpha-3B-v2 [39]	32.4	67.3	64.6	68.6	26.4	76.0	62.1	56.8
StableLM-3B-4E1T [40]	—	66.6	—	—	—	**76.8**	63.2	—
DIFF-3B	**37.8**	**72.9**	**69.0**	**71.4**	**29.0**	**76.8**	**67.1**	**60.6**

Model	ARC-C	ARC-E	BoolQ	HellaSwag	OBQA	PIQA	WinoGrande	Avg
Training with 350B tokens (Zero-Shot)								
Transformer-3B	32.2	66.8	**62.9**	63.4	26.2	74.5	61.6	55.4
DIFF-3B	**33.0**	**68.3**	60.1	**66.2**	**27.6**	**75.5**	**62.7**	**56.2**
Training with 350B tokens (5-Shot)								
Transformer-3B	34.0	**69.5**	65.3	63.4	25.0	75.2	62.6	56.4
DIFF-3B	**35.0**	**69.5**	**67.2**	**66.9**	**27.6**	**76.1**	**63.8**	**58.0**

Long-Context Evaluation

3.3 Long-Context Evaluation

We extend the 3B-size language models (described in Appendix B) to 64K context length. We continue training the 3B checkpoints for additional 1.5B tokens. Most hyperparameters are kept the same as in Section 3.1. The learning rate is 8e-5. The RoPE [36] θ is increased to 640,000. The training corpus is up-sampled according to sequence length [11].

Results Figure 4 presents cumulative average negative log-likelihood (NLL) of the tokens at varying positions [32], where lower NLL indicates better performance. The evaluation is conducted on book data within 64K length. We observe a consistent decrease in NLL as the context length increases. DIFF Transformer achieves lower NLL values than Transformer. The results demonstrate that DIFF Transformer can effectively leverage the increasing context.

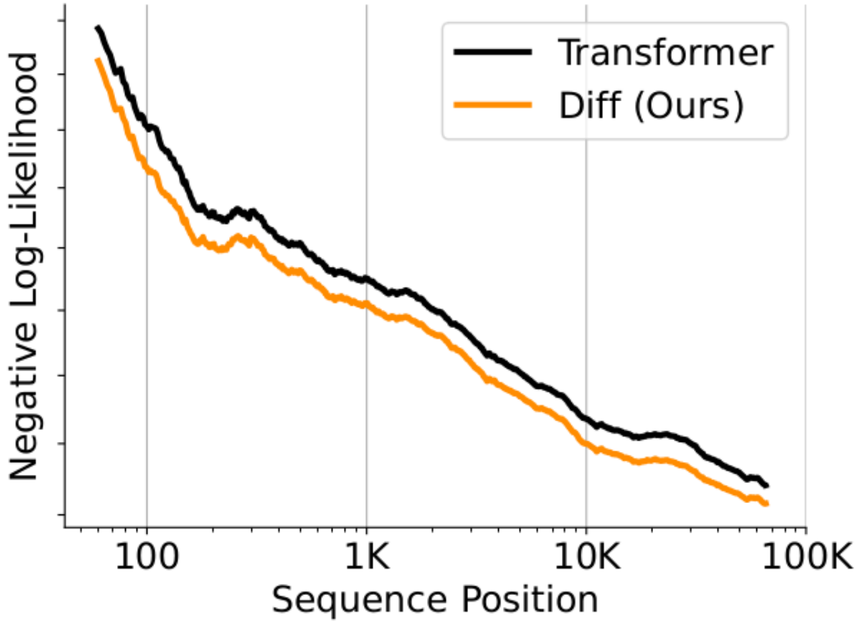
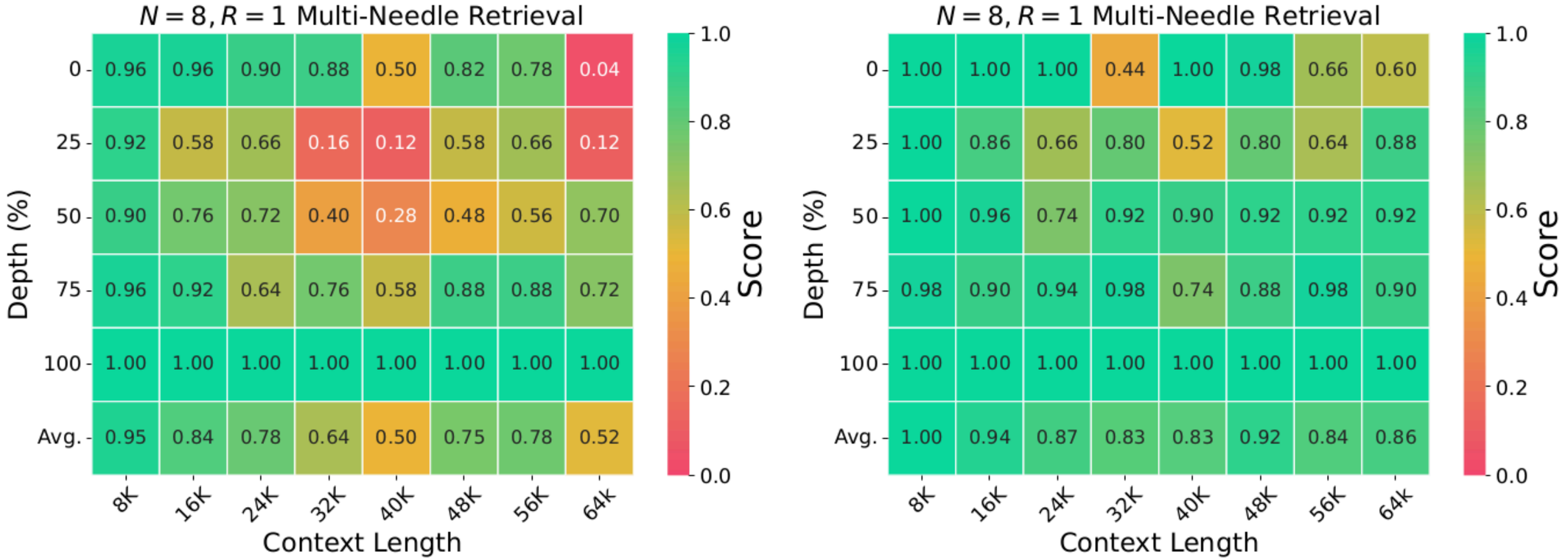


Figure 4: Cumulative average negative log-likelihood (lower is better) on book data. DIFF Transformer leverages long context more effectively.

Key Information Retrieval Results

Needle in the Haystack Test

https://github.com/gkamradt/LLMTest_NeedleInAHaystack



(a) Transformer.

(b) DIFF Transformer.

Figure 5: Multi-needle retrieval results in 64k length.

Key Information Retrieval

The Needle-In-A-Haystack [17] test is widely used to evaluate the ability to extract critical information embedded in a large context. We follow the multi-needle evaluation protocol of LWM [22] and Gemini 1.5 [32]. The needles are inserted into varying depths within contexts of different lengths. Each needle consists of a concise sentence that assigns a unique magic number to a specific city. The goal is to retrieve the magic numbers corresponding to the query cities. We position the answer needle at five different depths within the context: 0%, 25%, 50%, 75%, and 100%, while placing other distracting needles randomly. Each combination of depth and length is evaluated using 50 samples. The average accuracy is reported. Let N denote the total number of number-city pairs and R the number of query cities.

Model	$N = 1$ $R = 1$	$N = 2$ $R = 2$	$N = 4$ $R = 2$	$N = 6$ $R = 2$
Transformer	**1.00**	0.85	0.62	0.55
DIFF	**1.00**	**0.92**	**0.84**	**0.85**

Key Information Retrieval

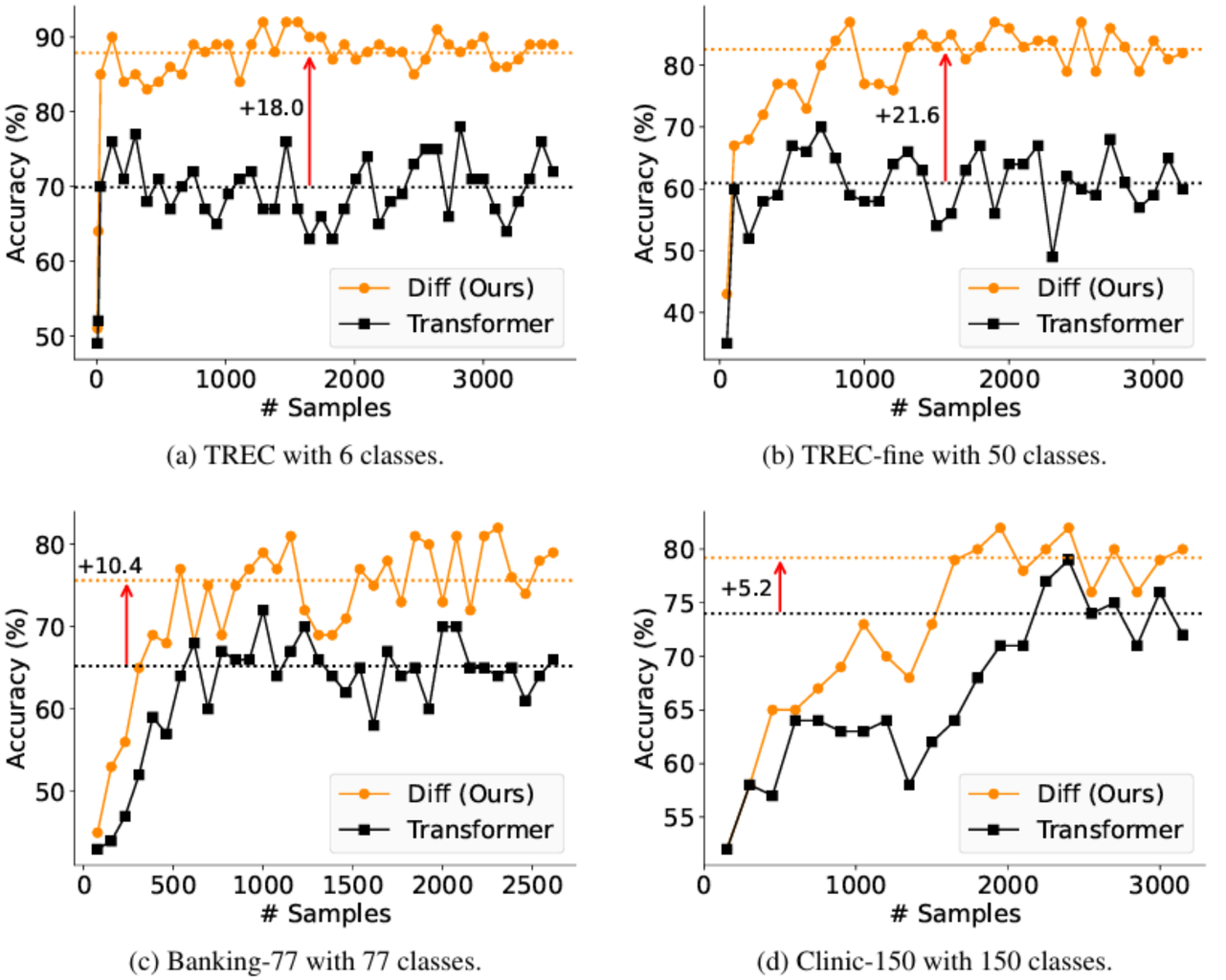
Attention Score Analysis Table 3 presents the attention scores allocated to the answer span and the noise context for the key information retrieval task. The scores indicate the model's ability to preserve useful information against attention noise. We compare the normalized attention scores when key information is inserted at different positions (i.e., depths) within the context. Compared with Transformer, DIFF Transformer allocates higher attention scores to the answer span and has lower attention noise.

Model	Attention to Answer ↑					Attention Noise ↓				
	0%	25%	50%	75%	100%	0%	25%	50%	75%	100%
Transformer	0.03	0.03	0.03	0.07	0.09	0.51	0.54	0.52	0.49	0.49
DIFF	**0.27**	**0.30**	**0.31**	**0.32**	**0.40**	**0.01**	**0.02**	**0.02**	**0.02**	**0.01**

In-Context Learning

In-Context Learning with Long-Context Models: An In-Depth Exploration
https://arxiv.org/abs/2405.00200

Many-Shot In-Context Learning As presented in Figure 6, we compare the accuracy of many shot classification between Transformer and our architecture. We evaluate the 3B-size language models that support **64K input length** (Section 3.3). We follow the evaluation protocol of [3] and use **constrained decoding**. We **incrementally increase the number of demonstration samples** from 1-shot until the total length reaches 64K length. Specifically, the **TREC** [15] dataset has 6 classes, **TREC-fine** [15] has 50 classes, **Banking-77** [5] has 77 classes, and **Clinic-150** [20] has 150 classes. The results show that DIFF Transformer consistently outperforms Transformer across datasets and varying numbers of demonstration samples. Moreover, the improvement in average accuracy is substantial, ranging from ***5.2% to 21.6%***.



(a) TREC with 6 classes.

(b) TREC-fine with 50 classes.

(c) Banking-77 with 77 classes.

(d) Clinic-150 with 150 classes.

Figure 6: Many-shot in-context learning accuracy on four datasets. Demonstration examples increase from 1-shot until the total length reaches 64K tokens. The dashed lines represent the average accuracy after the performance becomes stable.

Contextual Hallucination Evaluation

***Lookback Lens*: Detecting and Mitigating Contextual Hallucinations in Large Language Models Using Only Attention Maps**
https://arxiv.org/abs/2407.07071

We evaluate contextual hallucination of the 3B-size language models (described in Appendix B) on text summarization and question answering. Notice that we focus on the cases where the input context contains correct facts, but the model still fails to produce accurate outputs.

We follow the evaluation protocol of [6]. We feed the model output along with ground-truth responses to GPT-4o [27]. Then we ask GPT-4o to make binary judgements on whether the model outputs are accurate and free of hallucinations. Previous studies [6, 31] have shown that the above hallucination evaluation protocol has relatively high agreement between GPT-4o judgments and human annotations. The automatic metric is reliable and mirrors the human evaluation. For each dataset, the accuracy is averaged over 100 samples.

Model	XSum	CNN/DM	MultiNews
Transformer	0.44	0.32	0.42
DIFF	**0.53**	**0.41**	**0.61**

(a) Accuracy (i.e., free of hallucinations) on text summarization datasets.

Model	Qasper	HotpotQA	2WikiMQA
Transformer	0.28	0.36	0.29
DIFF	**0.39**	**0.46**	**0.36**

(b) Accuracy (i.e., free of hallucinations) on question answering datasets.

Activation Outliers Analysis

The DIFF Transformer demonstrates resilience in quantized attention logits by maintaining high zero-shot performance at reduced **bit-widths** (16 to 6 bits) on the HellaSwag dataset. Using dynamic post-training quantization with absmax quantization, it outperforms the standard Transformer model, which suffers accuracy loss at lower bit-widths. Notably, the **4-bit** DIFF Transformer achieves similar accuracy to the **6-bit** Transformer and exceeds the **4-bit** Transformer's accuracy by about **25%**, suggesting that DIFF Transformer's design mitigates activation outliers and enables efficient low-bit implementations such as FlashAttention.

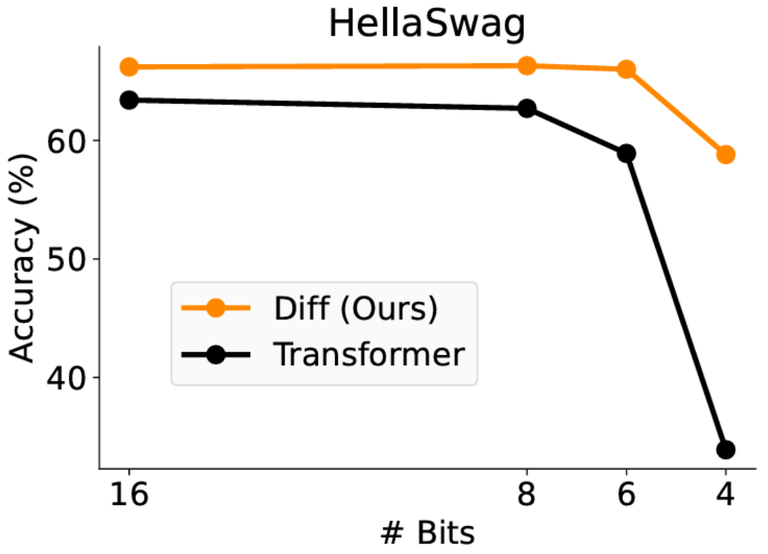


Figure 8: Zero-shot accuracy on the HellaSwag [12] dataset. We quantize the attention logits from 16 bits (i.e., unquantized) to 8 bits, 6 bits, and 4 bits.

Model	Activation Type	Top-1	Top-2	Top-3	Top-10	Top-100	Median
Transformer	Attention Logits	318.0	308.2	304.9	284.7	251.5	5.4
DIFF	Attention Logits	38.8	38.8	37.3	32.0	27.4	3.3
Transformer	Hidden States	3608.6	3607.4	3603.6	3552.1	2448.2	0.6
DIFF	Hidden States	1688.2	1672.5	1672.1	1624.3	740.9	1.2

Table 5: Largest activation values in attention logits and hidden states. Top activation values are considered as activation outliers, due to their significantly higher magnitude than the median. DIFF Transformer mitigates outliers compared to Transformer.

Ablation Studies

Model	#heads	d	GN	Valid. Set↓	Fine-Grained Slices AR-Hit↓	Fine-Grained Slices Others↓
Transformer	16	128	✗	3.087	0.898	3.272
Transformer	8	256	✗	3.088	0.899	3.273
+ GroupNorm	8	256	✓	3.086	0.899	3.271
DIFF Transformer	8	128	✓	**3.062**	**0.880**	**3.247**
− GroupNorm	8	128	✗	3.122	0.911	3.309
with $\lambda_{\text{init}} = 0.8$	8	128	✓	3.065	0.883	3.250
with $\lambda_{\text{init}} = 0.5$	8	128	✓	3.066	0.882	3.251

Table 6: Ablation studies of 1.4B-size models. We report language modeling loss on the validation set. We also follow Arora et al. [1] to report fine-grained metrics, where "AR-Hit" evaluates n-grams previously seen in the context. "#Heads" is number of heads. "d" is head dimension. "GN" indicates whether GroupNorm is used.

Conclusion

In this work, we introduce Differential Transformer (a.k.a. DIFF Transformer), which amplifies attention to the relevant context while canceling noise. Experimental results on language modeling show that DIFF Transformer outperforms Transformer in terms of scaling properties, long-context modeling, key information retrieval, hallucination mitigation, in-context learning, and reduction of activation outliers. The results emphasize the importance of reducing attention noise. Moreover, the differential attention mechanism can be easily implemented with FlashAttention [8]. The findings position DIFF Transformer as a distinctive and promising foundation architecture for large language models. In the future, we can develop efficient low-bit attention kernels due to the reduced magnitude of activation outliers. As the attention pattern becomes much sparser, we would also like to utilize the property to compress key-value caches.

Implementation of Differential Attention

```
def Attention(X, W_q, W_k, W_v):
    Q = X @ W_q
    K = X @ W_k
    V = X @ W_v
    # Q, K, V: [b, n, d]
    s = 1 / sqrt(d)
    A = Q @ K.transpose(-1, -2) * s

    return
      softmax(A) @ V
```

```
def DiffAttn(X, W_q, W_k, W_v, λ):
    Q1, Q2 = split(X @ W_q)
    K1, K2 = split(X @ W_k)
    V = X @ W_v
    # Qi, Ki: [b, n, d]; V: [b, n, 2d]
    s = 1 / sqrt(d)
    A1 = Q1 @ K1.transpose(-1, -2) * s
    A2 = Q2 @ K2.transpose(-1, -2) * s
    return
      (softmax(A1) - λ softmax(A2)) @ V
```

```
def FlashDiffAttn_1(X, W_q, W_k, W_v, λ):
    Q1, Q2 = split(X @ W_q)
    K1, K2 = split(X @ W_k)
    V = X @ W_v

    A1 = flash_attn(Q1, K1, V)


    A2 = flash_attn(Q2, K2, V)


    return A1 - λ A2
```

```
def FlashDiffAttn_2(X, W_q, W_k, W_v, λ):
    Q1, Q2 = split(X @ W_q)
    K1, K2 = split(X @ W_k)
    V1, V2 = split(X @ W_v)

    A11 = flash_attn(Q1, K1, V1)
    A12 = flash_attn(Q1, K1, V2)
    A1 = Concat(A11, A12)
    A21 = flash_attn(Q2, K2, V1)
    A22 = flash_attn(Q2, K2, V2)
    A2 = Concat(A21, A22)
    return A1 - λ A2
```

A Slight Drawback

Model	Model Size	Length	Throughput	
			Forward + Backward	Forward
Transformer	3B	2K	7247	51228
DIFF	3B	2K	6635 (−9%)	46811 (−9%)
Transformer	3B	4K	7491	48762
DIFF	3B	4K	6718 (−12%)	44521 (−10%)
Transformer	13B	2K	998	14346
DIFF	13B	2K	942 (−6%)	13653 (−5%)

THANK YOU

TEAM - 1

Bhavik Patel (bhavik.patel@iitgn.ac.in)

Guntas Singh Saran (guntassingh.saran@iitgn.ac.in)

Hitesh Kumar (hitesh.kumar@iitgn.ac.in)

Ruchi Jagodara (ruchit.jagodara@iitgn.ac.in)

Jinil Patel (jinilkumar.patel@iitgn.ac.in)

Indian Institue of Technology Gandhinagar
Palaj, Gujarat - 382355