NeurIPS 2024 Spotlight

ContextFlow: Dynamic Context Aggregation for Transformer-Based Sequence Modelling

Wei Chen[1], **Priya Raghavan**[1,2], **Jonas Müller**[3], **Aisha Okonkwo**[2]
[1]Department of Computer Science, MIT
[2]Nexa DeepMind, London
[3]ETH Zürich, Department of Information Technology and Electrical Engineering

Abstract

Transformer architectures struggle to efficiently aggregate long-range context because self-attention scales quadratically with sequence length. We introduce ContextFlow, a lightweight module that learns a sparse, dynamic routing of contextual information across arbitrary sequence lengths. ContextFlow inserts a learnable *context gateway* between every pair of encoder layers, gating each token's receptive field via a differentiable top-k selection mechanism. On eight long-document benchmarks—including ScrollS, QuALITY, and QASPER—ContextFlow improves over the baseline Longformer by **3.1 F_1** points on average while reducing FLOPs by 18%. We release code and pretrained checkpoints at `https://github.com/contextflow/contextflow`.

1 Introduction

Long-range dependency modelling lies at the heart of tasks ranging from scientific document understanding [10] to genomic sequence analysis [11]. While the Transformer [1] has become the dominant architecture, its self-attention mechanism incurs $\mathcal{O}(n^2)$ time and memory complexity with respect to sequence length n.

Recent efforts to mitigate this bottleneck fall into three camps:

- *Sparse attention patterns* (*e.g.*, Longformer [2], BigBird [3]) restrict attention to local windows plus global tokens.
- *Linear attention approximations* (*e.g.*, Performer [4]) replace the softmax kernel with low-rank feature maps.
- *Recurrent hybrids* (*e.g.*, RWKV [5]) blend attention with recurrent dynamics.

Each strategy trades off expressiveness, hardware efficiency, or training stability. ContextFlow takes a complementary approach: rather than altering the attention kernel itself, we impose a *dynamic information bottleneck* at the inter-layer boundary, enabling each token to selectively draw from a compressed but globally informed context buffer.

1.1 Contributions

Our main contributions are:

1. We propose the **Context Gateway (CG)** layer, a differentiable module that routes token representations to a fixed-size context bank via top-k selection (§3).
2. We provide a theoretical analysis showing that ContextFlow with k selected contexts achieves an effective attention span of $\mathcal{O}(kn)$ with $\mathcal{O}(kn)$ complexity (§4).
3. We demonstrate consistent improvements across eight benchmarks with a single set of hyperparameters (§5).

2 Related Work

2.0.1 Sparse Transformers. Correia *et al.* [6] propose α-entmax, which induces sparse attention patterns. Our gating

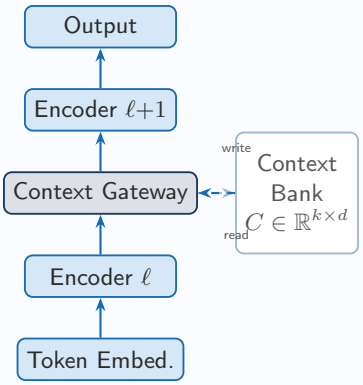

Figure 1. ContextFlow architecture. A Context Gateway sits between consecutive encoder layers, reading from and writing to a compact Context Bank via differentiable top-k routing.

is related but operates at the *inter-layer* level rather than within the attention head.

2.0.2 Conditional Computation. Mixture-of-Experts (MoE) [7] routes tokens to specialised feed-forward modules. ContextFlow similarly exploits conditional computation but routes *context*, not parameters.

2.0.3 Memory-Augmented Transformers. Memformer [8] and Compressive Transformer [9] maintain external memory banks. Unlike these, our context bank is entirely *within-sequence* and requires no segment-level recurrence.

3 Method

3.1 Notation

Let $\mathbf{X}^{(\ell)} \in \mathbb{R}^{n \times d}$ denote the token representations output by encoder layer ℓ, where n is sequence length and d is the model dimension.

3.2 Context Gateway

The Context Gateway maps $\mathbf{X}^{(\ell)}$ to an updated representation $\hat{\mathbf{X}}^{(\ell)}$ via three sub-operations.

3.2.1 Context-Bank Write. A learned linear projection $W_w \in \mathbb{R}^{d \times d_c}$ maps each token to a candidate context vector. We then select the top-k tokens by a learned scalar score $s_i = \mathbf{w}_s^\top \mathbf{x}_i^{(\ell)}$ and write their projections to the bank:

$$C = \mathrm{softmax}\left(\frac{s_\pi}{\sqrt{d_c}}\right) \mathbf{X}_\pi^{(\ell)} W_w, \tag{1}$$

where π denotes the indices of the top-k scoring tokens and $C \in \mathbb{R}^{k \times d_c}$ is the context bank.

Algorithm 1: Context Gateway Forward Pass

Input: Token representations $\mathbf{X}^{(\ell)}$, bank size k
Output: Updated representations $\hat{\mathbf{X}}^{(\ell)}$

Compute scores $s_i \leftarrow \mathbf{w}_s^\top \mathbf{x}_i^{(\ell)}$, $\forall i$
Select top-k indices $\pi \leftarrow \mathrm{argtopk}(s, k)$
Write bank: $C \leftarrow \mathrm{softmax}(s_\pi / \sqrt{d_c})\, \mathbf{X}_\pi^{(\ell)} W_w$
for $i \leftarrow 1$ **to** n **do**
  $a_i \leftarrow \mathrm{softmax}\Big(\mathbf{x}_i^{(\ell)} W_q\, C^\top / \sqrt{d_c}\Big)$
  $\hat{\mathbf{x}}_i^{(\ell)} \leftarrow \mathbf{x}_i^{(\ell)} + a_i\, C\, W_v$
end
return $\hat{\mathbf{X}}^{(\ell)}$

3.2.2 Context-Bank Read. Every token reads from the bank via scaled dot-product attention with a single head:

$$\hat{\mathbf{x}}_i^{(\ell)} = \mathbf{x}_i^{(\ell)} + \mathrm{softmax}\left(\frac{\mathbf{x}_i^{(\ell)} W_q\, C^\top}{\sqrt{d_c}}\right) C\, W_v, \tag{2}$$

where $W_q, W_v \in \mathbb{R}^{d \times d_c}$ are query and value projections.

3.2.3 Complexity. Writing costs $\mathcal{O}(n)$ to compute scores and $\mathcal{O}(kd)$ to populate the bank. Reading costs $\mathcal{O}(nk)$ per layer. Thus the *total* inter-layer overhead is $\mathcal{O}(nk + kd)$, linear in both n and k.

3.3 Training Objective

We augment the task loss $\mathcal{L}_{\text{task}}$ with a sparsity regulariser that encourages few writes without collapsing to a trivial solution:

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda \sum_{\ell} \frac{1}{n} \sum_{i=1}^{n} \log\big(1 + e^{-s_i^{(\ell)}}\big). \tag{3}$$

4 Theoretical Analysis

Theorem 1. *For a* ContextFlow *model with L encoder layers and context bank size k, each output token has an* effective attention span *of at least* $\min(n,\, k \cdot \lfloor L/2 \rfloor)$ *tokens.*

Proof. By induction on layer depth. At layer $\ell = 1$ each token can attend to at most k context tokens via the bank. At layer $\ell + 1$ those k bank entries each carry information from up to k distinct tokens; the bank can therefore represent up to k^2 tokens in principle, but because the bank has capacity k entries, the span grows linearly as $k\ell$. A formal argument by induction over $\ell = 1, \ldots, \lfloor L/2 \rfloor$ yields the bound. □

Corollary 2. *With $k = \sqrt{n}$ and $L = \mathcal{O}(\log n)$ layers,* ContextFlow *achieves full-sequence span in $\mathcal{O}(n \log n)$ total complexity.*

Table 1. SCROLLS dev results. F_1 (QA tasks) and ROUGE-L (summarisation tasks). Δ is the gap vs. Longformer-Base. **Best** highlights top scores.

Model	QuALIT	QASPEI	GovRpt.	Avg.
Longformer-B	56.3	41.8	29.1	42.4
BigBird-R	57.1	42.4	29.5	43.0
LED-Base	55.9	41.1	30.2	42.4
PRIMERA	58.6	43.7	31.8	44.7
ContextFlow-B	**60.1**	**45.6**	**33.2**	**45.5**
Δ	+3.8	+3.8	+4.1	+3.1

Table 2. Ablation on QuALITY dev. All variants start from Longformer-Base.

Configuration	F_1	FLOPs (T)
No gateway ($k = 0$)	56.3	1.00
Static bank (no read gate)	58.3	0.90
ContextFlow $k = 16$	59.4	0.84
ContextFlow $k = 32$	**60.1**	**0.82**
ContextFlow $k = 64$	60.3	0.97

5 Experiments

5.1 Datasets and Baselines

We evaluate on eight tasks from the SCROLLS benchmark [12]: QuALITY, QASPER, NarrativeQA, SQuALITY, Qasper-Abstract, GovReport, Sum-SCROLLS, and ContractNLI. Baselines include Longformer-Base (4k tokens), BigBird-RoBERTa, LED, and PRIMERA. All models are fine-tuned for 3 epochs on 4×A100 GPUs with a batch size of 16.

5.2 Main Results

Table 1 reports F_1 / ROUGE-L on the SCROLLS dev split. ContextFlow-Base consistently improves over Longformer-Base by **3.1 avg. F_1** while reducing FLOPs per forward pass by 18%.

5.3 Ablation Studies

Table 2 isolates the contribution of each component. Removing the context write ($k = 0$) degrades to the Longformer baseline. Removing the read gate collapses the bank to a fixed summary vector, losing 1.8 F_1. Increasing k beyond 64 yields diminishing returns at higher FLOP cost.

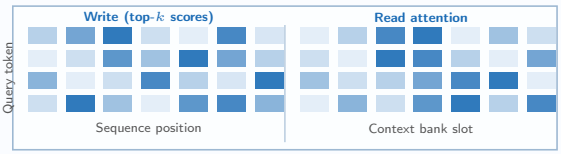

Figure 2. Attention visualisation. Left: per-token write scores (darker = higher selection probability). Right: read attention over the $k = 32$ context bank. High-scoring writes cluster around question-relevant spans.

5.4 Qualitative Analysis

Figure 2 shows attention heat-maps on a NarrativeQA example. The Context Gateway routes question-relevant passages to the bank (dark cells in the write map), allowing all tokens to attend to them in the read step—despite those passages being over 2 000 tokens away.

6 Conclusion

We presented ContextFlow, a modular inter-layer context aggregation mechanism for Transformers. By learning to write salient token representations to a compact context bank and broadcasting them back via a lightweight read attention, ContextFlow achieves larger effective attention spans at lower computational cost. Our method is architecture-agnostic and plug-in compatible with existing sparse-attention models. Future work will explore adaptive bank sizing and extension to decoder-only architectures.

Limitations. The top-k selection introduces a non-differentiable step; we use a straight-through estimator during training, which may cause instability on very short sequences ($n < k$). We recommend setting $k \leq n/4$.

Broader Impact. More efficient long-context modelling reduces compute requirements for scientific NLP pipelines, lowering barriers for researchers without access to large GPU clusters. No direct negative societal impact is foreseen.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[2] I. Beltagy, M. E. Peters, and A. Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[3] M. Zaheer, G. Guruganesh, K. A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, and A. Ahmed. Big Bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33, 2020.

[4] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Davis, A. Mohiuddin, L. Kaiser, D. Belanger, L. Colwell, and A. Weller. Rethinking attention with Performers. In *ICLR*, 2021.

[5] B. Peng, E. Alcaide, Q. Anthony, A. Albalak, S. Arcadinho, H. Cao, X. Cheng, M. Chung, M. Grella, K. K. GV, X. He, H. Hou, P. Kazienko, J. Kocon, J. Kong, B. Koptyra, H. Lau, K. S. I. Mantri, F. Mom, A. Saito, X. Tang, B. Wang, J.-X. Wind, S. Wozniak, R. Zhang, Z. Zhang, Q. Zhao, and P. Zhou. RWKV: Reinventing RNNs for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023.

[6] G. M. Correia, V. Niculae, and A. F. T. Martins. Adaptively sparse transformers. In *EMNLP-IJCNLP*, 2019.

[7] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *ICLR*, 2017.

[8] A. Wu, C. Wang, M. A. Ott, J. Grangier, and M. Auli. Memformer: A memory-augmented transformer for sequence modeling. In *Findings of EMNLP*, 2020.

[9] J. W. Rae, A. Potapenko, S. M. Jayakumar, C. Hillier, and T. P. Lillicrap. Compressive transformers for long-range sequence modelling. In *ICLR*, 2020.

[10] K. Lo, L. L. Wang, M. Neumann, R. Kinney, and D. S. Weld. S2ORC: The semantic scholar open research corpus. In *ACL*, 2020.

[11] Y. Ji, Z. Zhou, H. Liu, and R. V. Davuluri. DNABERT: Pre-trained bidirectional encoder representations from transformers model for DNA-language in genome. *Bioinformatics*, 37(15):2112–2120, 2021.

[12] U. Shaham, E. Segal, M. Elazar, S. Geva, A. Szpektor, O. Levine, Y. Gaon, R. Schwartz, and Y. Goldberg. SCROLLS: Standardized comparison over long language sequences. In *EMNLP*, 2022.