BSA: Ball Sparse Attention for Large-scale Geometries

Catalin E. Brita*, **Hieu Nguyen***, **Lohithsai Yadala Chanchu***, **Domonkos Nagy***
University of Amsterdam

Abstract

Self-attention scales quadratically with input size, limiting its use for large-scale physical systems. Although sparse attention mechanisms provide a viable alternative, they are primarily designed for regular structures such as text or images, making them inapplicable for irregular geometries. In this work, we present Ball Sparse Attention (BSA), which adapts Native Sparse Attention (NSA) [28] to unordered point sets by imposing regularity using the ball-tree structure from the Erwin Transformer [31]. We modify each of NSA's components to work with ball-based neighborhoods, yielding a global receptive field at sub-quadratic cost. On an airflow pressure prediction task, we achieve accuracy comparable to full attention while reducing computational cost by a significant margin. We open-source BSA.

1 Introduction

Scientific applications such as climate modeling [6], molecule property prediction [17], or fluid flow simulation [16] increasingly rely on transformer-based architectures to capture complex, long-range dependencies in irregular data [1, 5, 14, 15]. However, standard self-attention scales quadratically with input size, making it impractical for large-scale tasks in the scientific domain. This has motivated the development of scalable strategies for large-scale physical systems.

Sparse attention mechanisms mitigate quadratic scaling by computing attention for a strategically chosen subset of token pairs. These range from predefined or random sparsity patterns (e.g., BigBird [29]) to learned, data-dependent sparsity, as seen in Native Sparse Attention (NSA) [28]. NSA can select and compress tokens across the full sequence, allowing it to capture fine global dependencies.

However, NSA is designed to work with text sequences that have a regular structure, unlike many physical systems. This poses a challenge as such structures are represented as unordered sets that do not have a canonical ordering that sparse methods utilize. Many approaches [12, 20, 26] induce regular structure and transform point clouds into sequences to serve as inputs for sparse attention.

Recently proposed Erwin [31] organizes points into a ball tree: the leaf level represents the full sequence, and balls in higher levels of the tree represent larger neighborhoods. Erwin enables linear-time attention by processing nodes in parallel within local neighborhoods of fixed size. It combines fine-grained local attention with progressive pooling for capturing global interactions. This approach performs well at local interactions but may require several steps for distant ones, as accumulating global information requires multiple layers. The progressive coarsening of such hierarchical methods also results in a loss of fidelity, as coarsened features cannot be processed at finer scales.

To fix these issues, we present Ball Sparse Attention (BSA), which includes Ball Tree Attention within NSA's framework to achieve a global receptive field at sub-quadratic cost. Our contributions are (a) a hybrid architecture integrating Ball Tree Attention within NSA's framework for scalable scientific modeling, (b) a locality-based sparsification strategy for attention that preserves modeling capacity, and (c) a comprehensive validation of Ball Sparse Attention on airflow pressure modeling and stress field prediction in hyperelastic materials.

*Equal contribution.

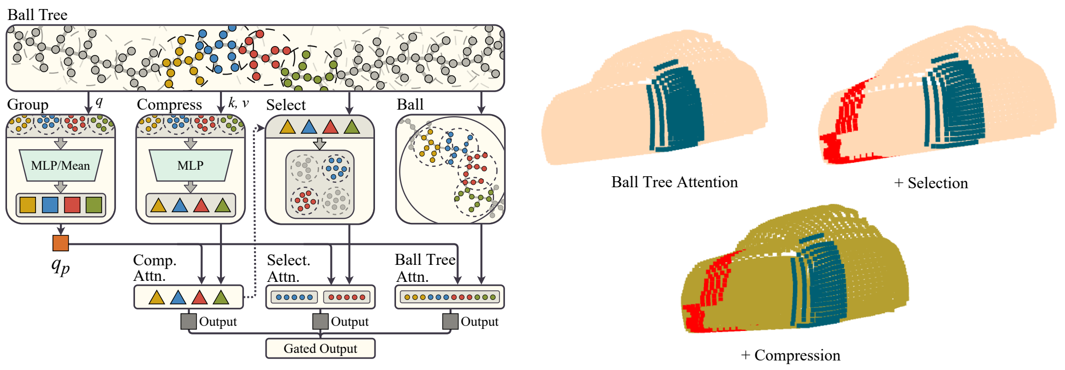

Figure 1: **Left:** Ball Sparse Attention (BSA) pipeline. A ball tree imposes spatial locality, then three sparse-attention branches—grouping (block clustering), compression (MLP-based token pooling), and selection (top-k block retrieval) operate alongside fine-grained Ball Tree Attention. A learnable gate fuses their outputs into the final attention. **Right:** Receptive field visualization of a car in Shapenet dataset with different components: Ball Tree Attention; Ball Tree Attention and selection; Ball Tree Attention, selection, and compression. The receptive field increases with more components.

2 Methodology

2.1 Background

The **self-attention mechanism** builds on the scaled dot-product attention [22]. For an input matrix $X \in \mathbb{R}^{N \times C}$, X is projected into queries, keys, and values:

$$Q = XW_q, \quad K = XW_k, \quad V = XW_v, \tag{1}$$

where $W_q,\ W_k\ W_v \in \mathbb{R}^{C \times d_k}$ are learnable weight matrices. Then, the attention output is computed:

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}\Big(\frac{QK^T}{\sqrt{d_k}} + \mathcal{B}\Big)V \tag{2}$$

Even with highly optimized implementations (e.g., [7]), self-attention scales quadratically with the sequence length, making it hard to process sequences longer than tens of thousands of tokens. Recent studies have attempted to address this by imposing geometric or sparsity-based relaxations.

Ball Tree Attention (BTA): Zhdanov et al. [31] partition the sequence into disjoint balls B of size m, each having m feature vectors $X_B \in \mathbb{R}^{m \times C}$. Then, BTA applies self-attention *within* each ball:

$$X'_B = \mathrm{Attn}^{\mathrm{ball}}(X_B) := Attn\left(X_B W_q, X_B W_k, X_B W_v\right), \tag{3}$$

where the weights are shared across balls and maintain row correspondence with X_B.

Native Sparse Attention (NSA): Yuan et al. [28] preserves full sequence resolution by sparsifying attention in three branches: (1) *compression*, (2) *selection*, and (3) *sliding window*. These branches combine via gated attention $\mathrm{Attn} = \sum_{b \in \{\mathrm{sld}, \mathrm{cmp}, \mathrm{slc}\}} \sigma(\gamma_b) \odot Attn^b$, where each gate γ_b is passed through a sigmoid function $\sigma(\cdot)$ and used to modulate its branch output Attn^b.

Compression: NSA splits K and V into non-overlapping blocks of length ℓ (stride $= \ell$) and maps each block to a single coarse token via a MLP ϕ (or mean):

$$K^{\mathrm{cmp}} = \left\{\phi\big(K_{(i-1)\ell:i\ell-1}\big)\right\}_{i=1}^{\lceil N/\ell \rceil}, \quad V^{\mathrm{cmp}} \quad = \left\{\phi\big(V_{(i-1)\ell:i\ell-1}\big)\right\}_{i=1}^{\lceil N/\ell \rceil}, \tag{4}$$

where $K_{(i-1)\ell:i\ell-1}, V_{(i-1)\ell:i\ell-1} \in \mathbb{R}^{\ell \times d_k}, \forall i \leq \lceil N/\ell \rceil$ (0-padded), and $\phi(\cdot)$ outputs a vector in $\mathbb{R}^{d_k}$. The *compressed attention* is $\mathrm{Attn}^{\mathrm{cmp}} = \mathrm{Attn}(Q, K^{\mathrm{cmp}}, V^{\mathrm{cmp}})$.

Selection: For each query position t, we reuse the coarse keys to build the similarity matrix:

$$S = Q(K^{\mathrm{cmp}})^T \in \mathbb{R}^{N \times \lceil N/\ell \rceil}, \quad S_{tj} = \langle q_t, k_j^{\mathrm{cmp}} \rangle \tag{5}$$

The indices of the top $k^\star$ blocks are then selected for each t:

$$\mathcal{I}_t = \text{top-k}(S_{t,\cdot}, k^\star) \subset \{1, \cdots, \lceil N/\ell \rceil\} \tag{6}$$

Selected blocks are then converted to token-level representations by concatenating their KV tokens:

$$K_t^{\text{slc}} = \text{Cat}\Big\{K_{(i-1)\ell:i\ell-1} \mid i \in \mathcal{I}_t\Big\}, \quad V_t^{\text{slc}} \quad = \text{Cat}\Big\{V_{(i-1)\ell:i\ell-1} \mid i \in \mathcal{I}_t\Big\} \tag{7}$$

The *selection attention* is $\text{Attn}^{\text{slc}} = \text{Attn}(Q, K^{\text{slc}}, V^{\text{slc}})$. Despite attending to only a fraction of pairs, NSA matches full attention on NLP tasks and achieves an $11\times$ speedup in computation [28].

2.2 Ball Sparse Attention

Ball Sparse Attention (BSA): We propose BSA, an attention mechanism that inherits NSA's three-branch design: *compression*, *selection*, and *local attention*. However, we replace the conventional sliding window with Ball Tree Attention [31] (see BTA in Figure 1 when $q_p = q_t$), allowing the model to attend within continuous geometric regions in $\mathbb{R}^D$ and avoid artificial discontinuities. The compression and selection branches remain as in NSA, and we combine all three branches as follows:

$$\text{Attn} = \sum_{b \in \{\text{ball,cmp,slc}\}} \sigma(\gamma_b) \odot \mathit{Attn}^b. \tag{8}$$

Group selection: Beyond locality in Ball Tree Attention, we also exploit locality during *selection*, in the top-k computation. To achieve this, we group query positions t into contiguous groups of size g (see Groping in Figure 1), and enforce one set of selected blocks across all queries (Q) in the same group:

$$G_p = \big\{(p-1)g, \dots, pg-1\big\}, \quad p = 1, \dots, \lceil N/g \rceil, \quad q_p = \frac{1}{|G_p|} \sum_{t \in G_p} Q_t \tag{9}$$

where p indexes each group and G_p is the indices in the p-th group, and we pool the Q in each group. We then average the similarity scores within a group and perform top-k selection on these averages:

$$\bar{S}_{pj} = \frac{1}{|G_p|} \sum_{t \in G_p} S_{tj}, \quad j = 1, \dots, \lceil N/\ell \rceil \quad \text{and,} \quad \mathcal{I}_p \quad = \text{top-k}\big(\bar{S}_{p,\cdot}, \, k^\star\big), \quad \mathcal{I}_t \equiv \mathcal{I}_p \;\; \forall\, t \in G_p. \tag{10}$$

This reduces top-k calls by a factor of g and allows KV blocks to be fetched in contiguous chunks, improving GPU cache utilisation and lowering memory-access latency. To further optimize the runtime, we shrink the dimensionality of the query before the *selection* stage by coarsening Q with an MLP ϕ before computing the similarity matrix:

$$Q^{\text{cmp}} = \big\{\, \phi\big(Q_{(i-1)\ell:i\ell-1}\big)\big\}_{i=1}^{\lceil N/\ell \rceil}, \tag{11}$$

with $Q_{(i-1)\ell:i\ell-1}, \forall i \leq \lceil N/\ell \rceil$ (zero-padded). The similarity matrix and its selection step are:

$$\widetilde{S} = Q^{\text{cmp}}(K^{\text{cmp}})^T \in \mathbb{R}^{\lceil N/\ell \rceil \times \lceil N/\ell \rceil}, \quad \widetilde{\mathcal{I}_p} \quad = \text{top-k}(\widetilde{S}_{p,\cdot}, k^\star) \subset \{1, \cdots, \lceil N/\ell \rceil\}, \tag{12}$$

then proceed as above. This yields a small loss in resolution but improves similarity computation.

Group compression: Finally, we down-sample Q not only for *selection* but also for *compression*:

$$\text{Attn}^{\text{cmp}} = \text{Attn}\Big(\underbrace{(I_{\lceil N/\ell \rceil} \otimes \mathbf{1}_\ell)}_{\text{repeat}} Q^{\text{cmp}}, K^{\text{cmp}}, V^{\text{cmp}}\Big), \tag{13}$$

where $I_{\lceil N/\ell \rceil} \otimes \mathbf{1}_\ell$ repeats each compressed query ℓ times. Compressing Q in both *selection* and *compression* maximizes speed-up at the price of a drop in downstream accuracy.

3 Experiments and Results

3.1 Experiments setup

Task: We evaluate the proposed architecture on an airflow pressure modeling task with the ShapeNet dataset [21] and stress field prediction task on the Elasticity benchmark [10].

Training details: The model consists of 18 transformer blocks, each containing an RMSNorm layer [30], a Ball Sparse Attention layer (BSA), and a SwiGLU [19] for non-linear transformation. The BSA parameters and training hyperparameters are detailed in Appendix A.

3.2 Experimental results

Table 1: Shapenet test MSE.

Model	MSE
PointNet ([18])	43.36
GINO ([11])	35.24
UPT ([2])	31.66
Transolver ([25])	19.88
PTv3 ([26])	19.09
GP-UPT ([3])	17.02
Erwin ([31])	15.85
BSA (Ours)	14.47
Full attention ([22])	13.29

Table 2: Elasticity test rMSE.

Model	rMSE
LSM ([24])	0.0218
LNO ([23])	0.0069
Galerkin ([4])	0.0240
Oformer ([9])	0.0183
Gnot ([8])	0.0086
Ono ([27])	0.0118
Transolver ([25])	0.0064
Erwin ([31])	0.0035
BSA (Ours)	0.0039

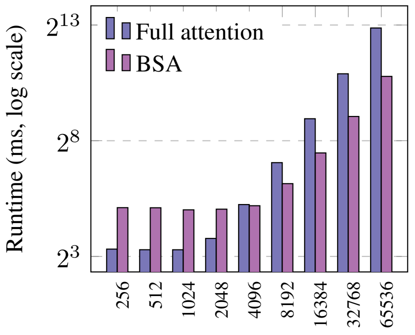

Figure 2: Runtime of BSA and Full attention with increasing sequence length.

BSA Comparison with previous methods: Table 1 shows that BSA outperforms all previous methods, and 1.18 MSE worse than full attention on Shapenet. On the stress field prediction task, Table 2 shows that BSA achieves approximately the same performance as Erwin [31] and Transolver [25]. This outcome can be attributed to the small-scale nature of the Elasticity dataset, with sequence lengths of 972, where BSA fails to demonstrate a clear advantage.

Accuracy–Efficiency trade-off: Table 3 illustrates that all BSA variants achieve significantly better results than Erwin [31] and approach the accuracy of Full Attention [22]; these BSA variants use more GFLOPs than Erwin but fewer than Full Attention. Among the BSA variants, although those without group selection and without query coarsening require higher GFLOPs and runtime, the MSE difference is marginal (0.16), supporting our assumption about local structure in this physical task.

Table 3: Comparison of Global, BSA, and Erwin attention.

Attention type	MSE	Runtime (ms)	GFLOPS
Erwin	16.12	19.35	14.60
Full attention	13.29	37.82	87.08
BSA	14.47	36.53	46.05
– group select	14.44	66.92	53.17
– query coarsen	14.31	42.30	53.17
+ group compress	14.80	23.42	29.13

Sparse attention scaling: Figure 2 shows that, while full attention is faster at short sequence lengths due to BSA's MLP overhead, the difference shrinks as the sequence length increases. BSA becomes faster than full attention at length 4096 and at the largest sequence length, it is 5 times faster.

Sparse attention receptive field: Figure 1 shows the receptive field of each component in BSA. Initially, Ball Tree Attention only attends to points inside a local ball. With selection, it can attend to different balls far away from the current point. Finally with compression, it achieves a global receptive field by attending to all coarse key-value representations of each local ball.

4 Conclusion

We introduce BSA, a novel sparse attention mechanism suited for large scale physical systems. Our approach includes Ball Tree Attention within the NSA framework and introduces a locality-based sparsification strategy that preserves the performance of high-quality long-range modeling while drastically reducing computational efficiency. Through grouped selection and hybrid attention, we preserve local and global context with minimal overhead.

Our results show that BSA achieves competitive performance comparable to full attention while reducing runtime by up to 5×. It also significantly expands the receptive field, enabling better information propagation across distant points. BSA's efficiency and competitive performance on large sequence lengths makes it ideal for tasks involving the simulation of large-scale physical systems.

Future work: We will test our grouping mechanism on more datasets and develop a GPU kernel to exploit group BSA operations for improved computational efficiency and reduced memory footprint.

References

[1] Josh Abramson et al. "Accurate structure prediction of biomolecular interactions with AlphaFold 3". In: *Nature* 630.8016 (May 2024), pp. 493–500. ISSN: 1476-4687. DOI: 10.1038/s41586-024-07487-w. URL: http://dx.doi.org/10.1038/s41586-024-07487-w.

[2] Benedikt Alkin et al. "Universal Physics Transformers". In: *arXiv preprint arXiv:2402.12365* (2024).

[3] Maurits Bleeker et al. "NeuralCFD: Deep Learning on High-Fidelity Automotive Aerodynamics Simulations". In: *CoRR* abs/2502.09692 (2025). URL: https://doi.org/10.48550/arXiv.2502.09692.

[4] Shuhao Cao. "Choose a Transformer: Fourier or Galerkin". In: *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. Ed. by Marc'Aurelio Ranzato et al. 2021, pp. 24924–24940.

[5] Siyuan Chen et al. "Geometrically aware transformer for point cloud analysis". In: *Scientific Reports* 15.1 (May 2025). ISSN: 2045-2322. DOI: 10.1038/s41598-025-00789-7. URL: http://dx.doi.org/10.1038/s41598-025-00789-7.

[6] Declan Curran et al. "Resolution-Agnostic Transformer-based Climate Downscaling". In: *NeurIPS 2024 Workshop on Tackling Climate Change with Machine Learning*. 2024. URL: https://www.climatechange.ai/papers/neurips2024/74.

[7] Tri Dao et al. "FlashAttention: Fast and Memory-Efficient Exact Attention with I/O-Awareness". In: *Advances in Neural Information Processing Systems*. Vol. 35. 2022, pp. 16344–16359.

[8] Zhongkai Hao et al. "GNOT: A General Neural Operator Transformer for Operator Learning". In: *arXiv preprint arXiv:2302.14376* (2023).

[9] Zijie Li, Kazem Meidani, and Amir Barati Farimani. "Transformer for Partial Differential Equations' Operator Learning". In: *Transactions on Machine Learning Research* (2023). ISSN: 2835-8856. URL: https://openreview.net/forum?id=EPPqt3uERT.

[10] Zongyi Li et al. *Fourier Neural Operator for Parametric Partial Differential Equations*. 2021. arXiv: 2010.08895 [cs.LG]. URL: https://arxiv.org/abs/2010.08895.

[11] Zongyi Li et al. "Geometry-Informed Neural Operator for Large-Scale 3D PDEs". In: *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*. Ed. by Alice Oh et al. 2023.

[12] Zhijian Liu et al. "FlatFormer: Flattened Window Attention for Efficient Point Cloud Transformer". In: June 2023, pp. 1200–1211. DOI: 10.1109/CVPR52729.2023.00122.

[13] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: 1711.05101 [cs.LG]. URL: https://arxiv.org/abs/1711.05101.

[14] Ziwei Luo et al. "D2T-Net: A dual-domain transformer network exploiting spatial and channel dimensions for semantic segmentation of urban mobile laser scanning point clouds". In: *International Journal of Applied Earth Observation and Geoinformation* 132 (2024), p. 104039. ISSN: 1569-8432. DOI: https://doi.org/10.1016/j.jag.2024.104039. URL: https://www.sciencedirect.com/science/article/pii/S1569843224003935.

[15] Siqi Miao et al. "Locality-sensitive hashing-based efficient point transformer with applications in high-energy physics". In: *Proceedings of the 41st International Conference on Machine Learning*. ICML'24. Vienna, Austria: JMLR.org, 2024.

[16] Wenhui Peng et al. "Linear attention coupled Fourier neural operator for simulation of three-dimensional turbulence". In: *Physics of Fluids* 35.1 (Jan. 2023), p. 015106.

[17] Zihan Pengmei et al. *Transformers are efficient hierarchical chemical graph learners*. 2023. arXiv: 2310.01704 [cs.LG]. URL: https://arxiv.org/abs/2310.01704.

[18] Charles R Qi et al. "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation". In: *arXiv preprint arXiv:1612.00593* (2016).

[19] Noam Shazeer. "GLU Variants Improve Transformer". In: *CoRR* abs/2002.05202 (2020). URL: https://arxiv.org/abs/2002.05202.

[20] Pei Sun et al. "SWFormer: Sparse Window Transformer for 3D Object Detection in Point Clouds". In: Nov. 2022, pp. 426–442. ISBN: 978-3-031-20079-3. DOI: 10.1007/978-3-031-20080-9_25.

[21] Nobuyuki Umetani and Bernd Bickel. “Learning three-dimensional flow for interactive aerodynamic design”. In: *ACM Trans. Graph.* 37.4 (July 2018).

[22] Ashish Vaswani et al. “Attention Is All You Need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*. 2017, pp. 6000–6010. URL: `https://proceedings.neurips.cc/paper/7181-attention-is-all-you-need.pdf`.

[23] Tian Wang and Chuang Wang. “Latent Neural Operator for Solving Forward and Inverse PDE Problems”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2024.

[24] Haixu Wu et al. “Solving High-Dimensional PDEs with Latent Spectral Models”. In: *International Conference on Machine Learning*. 2023.

[25] Haixu Wu et al. “Transolver: A Fast Transformer Solver for PDEs on General Geometries”. In: *International Conference on Machine Learning*. 2024.

[26] Xiaoyang Wu et al. “Point Transformer V3: Simpler, Faster, Stronger”. In: *CVPR*. 2024.

[27] Zipeng Xiao et al. “Improved Operator Learning by Orthogonal Attention”. In: *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. 2024.

[28] Jingyang Yuan et al. *Native Sparse Attention: Hardware-Aligned and Natively Trainable Sparse Attention*. 2025. arXiv: `2502.11089 [cs.CL]`. URL: `https://arxiv.org/abs/2502.11089`.

[29] Manzil Zaheer et al. “Big bird: Transformers for longer sequences”. In: *Advances in Neural Information Processing Systems* 33 (2020).

[30] Biao Zhang and Rico Sennrich. “Root Mean Square Layer Normalization”. In: *Advances in Neural Information Processing Systems 32*. Vancouver, Canada, 2019. URL: `https://openreview.net/references/pdf?id=S1qBAf6rr`.

[31] Maksim Zhdanov, Max Welling, and Jan-Willem van de Meent. *Erwin: A Tree-based Hierarchical Transformer for Large-scale Physical Systems*. 2025. arXiv: `2502.17019 [cs.LG]`. URL: `https://arxiv.org/abs/2502.17019`.

A Training hyperparameters

Table 4 details the sparse attention hyperparameters. The model is trained with mean squared error loss for regression task, and it is trained for 100000 iterations with AdamW optimizer [13] with a cosine learning rate scheduler, learning rate 0.001, and weight decay 0.01.

Table 4: Sparse attention parameters

Parameter	Value
Ball size	256
Compression block size	8
Compression block sliding stride	8
Selection block size	8
Number of blocks selected	4

B Impacts of rotation

Table 5: Sparse attention between regular selections and group selections, with and without rotations.

Model	MSE	Runtime (ms)	GFLOPS
BSA	14.44	77.13	53.17
Group BSA	14.31	30.04	46.05
BSA w/o rotation	15.97	74.75	53.17
Group BSA w/o rotation	16.13	40.65	46.05

Tree rotation is proposed in Erwin [31] to achieve distant point interactions, addressing the local nature of the Erwin architecture. While our sparse attention mechanism offers long-range interactions through compression and selection mechanisms, Table 5 shows that rotation improves performance significantly, reducing MSE by over 1 point for both regular sparse attention and variants with group selection. This improvement occurs because rotating the points alters the block structures, enabling points to attend to more meaningful points rather than being limited by coarse-grained selection or a small number of selections. These results demonstrate the importance of rotation in this task, even when long-range sparse attention layers are employed.

C Runtime scaling analysis

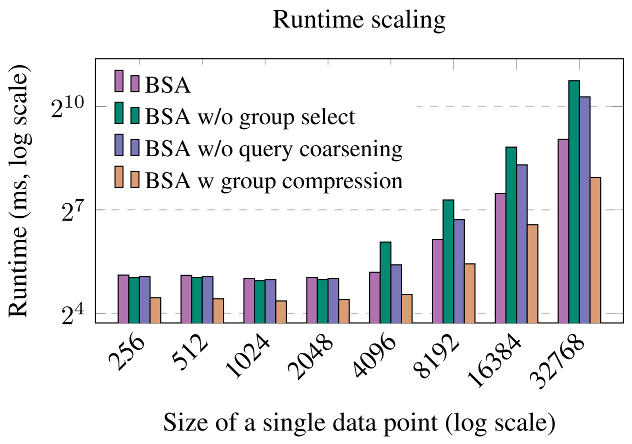

Figure 3: Runtime of BSA and its variants with increasing sequence length (from 256 to 32768).