Linear Transformer Topological Masking with Graph Random Features

GDL 2025
Group id: ——
Project id: ——

Paolo Deidda, Raffaele Perri, Paul Leopold Seipl
{paolo.deidda, raffaele.perri, paul.leopold.seipl}@usi.ch

Abstract
Concise and self-contained description of your project, motivation and main findings.

GENERAL NOTES

The report should be written as an article intended to present the findings of your work. Your aim should be to be clear and objective, substantiating your claims with references or empirical/theoretical evidence. We are well aware of the fact that carrying out machine learning experiments might be difficult and that often the final performance might be disappointing. For this reason, you will not be evaluated solely on quantitative aspect of your work, but mainly on the quality of your analysis and report. The length of the report should be between 4 and 8 pages (without considering references).

Contents

1 Introduction

Transformers have established themselves as a dominant architecture across various machine learning modalities, deriving their power from the attention mechanism which models complex dependencies between tokens [**?**]. However, the standard Transformer treats input data as a set, making it invariant to permutation and inherently unaware of structural dependencies, such as the connectivity in graph-structured data. To address this, *Topological Masking* is often employed, where the attention mechanism is modulated by a function of the graph structure (e.g., shortest path distance or adjacency), injecting a necessary structural inductive bias [**? ?**].

A fundamental computational conflict arises when scaling this approach to large graphs. Standard "Vanilla" Softmax attention requires explicitly computing and storing an $N \times N$ attention matrix, resulting in quadratic $\mathcal{O}(N^2)$ time and space complexity. While Linear Attention mechanisms address this bottleneck by leveraging low-rank decompositions $\phi(Q)(\phi(K)^T V)$ to achieve $\mathcal{O}(N)$ complexity [**?**], they rely strictly on the associativity of matrix multiplication. Introducing a topological mask M generally breaks this associativity—since $(A \times B) \odot M \neq A \times (B \odot M)$—forcing the re-materialization of the dense attention matrix and negating the efficiency gains of linear attention.

In this work, we focus on the reproduction of "Linear Transformer Topological Masking with Graph Random Features" [**?**]. The authors propose a novel solution to the masking conflict by approximating the topological mask using Graph Random Features (GRFs). By decomposing the mask into sparse feature vectors derived from random walks, the method allows the mask to be fused with query and key features via a tensor product, preserving the $\mathcal{O}(N)$ complexity of linear attention while maintaining the expressivity of topological masking.

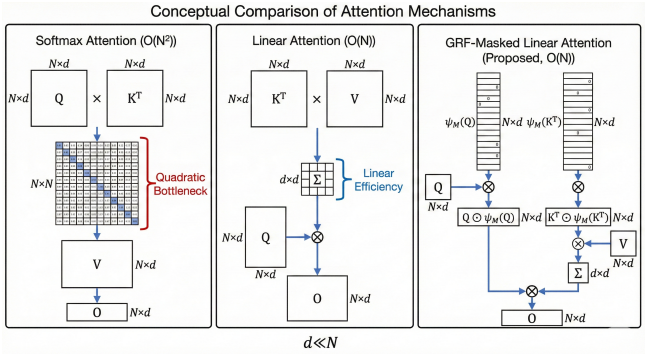


Figure 1. Conceptual comparison of attention mechanisms. The left panel illustrates the quadratic bottleneck of Softmax attention. The middle panel shows the efficient flow of Linear attention. The right panel demonstrates the proposed GRF method, which preserves linear complexity by fusing sparse topological features ψ_M with the query/key representations.

Contributions and Replication Results Our primary contribution is a validation of the theoretical and empirical claims presented in the original paper. We successfully implemented the proposed GRF-based masking mechanism and conducted a rigorous analysis of its computational scaling behavior.

- **Time Complexity Verification:** We replicated the scaling experiment comparing Softmax, Linear, and GRF-Masked attention. Our results confirm the theoretical claims: while Softmax attention scales quadratically, exhibiting a "computational explosion" for $N > 8$, the GRF-Masked method maintains strict linear scaling $\mathcal{O}(N)$.
- **Overhead Analysis:** We quantified the overhead introduced by the GRF mechanism. While retaining the linear complexity class, we observe a constant factor increase in FLOPs compared to unmasked linear attention, attributable to the sparsity of the generated graph features.

!!!TO DO: ADD THE OTHER TWO EXPERIMENTS RESULTS OVERVIEW!!!

2 Related works

Standard and Topological Attention The standard self-attention mechanism, introduced by **?**], computes a dense $N \times N$ attention matrix $A = \text{softmax}(QK^\top/\sqrt{d})$. While highly effective at modeling global dependencies, this approach incurs quadratic $\mathcal{O}(N^2)$ time and space complexity, prohibiting its use on long sequences or large graphs. To adapt Transformers to graph-structured data, *topological masking* is often employed to inject structural inductive bias [**?**]. This typically involves modulating the attention scores with a mask $M(\mathcal{G})$ derived from the graph topology (e.g., shortest path distances), forcing tokens to attend preferentially to their structural neighbors. However, applying such masks generally requires materializing the full $N \times N$ matrix, maintaining the prohibitive quadratic bottleneck.

Linear Attention and The Masking Conflict To address the scalability limits of standard attention, *Linear Attention* mechanisms have been proposed [**? ?**]. These methods replace the softmax kernel with a feature map decomposition $\phi(\cdot)$, allowing the computation to be reordered via associativity: $D^{-1}\phi(Q)(\phi(K)^\top V)$. This reduces complexity to $\mathcal{O}(N)$. However, as noted in the foundational literature, introducing an element-wise topological mask M breaks this associativity, i.e., $(A \times B) \odot M \neq A \times (B \odot M)$. Consequently, standard linear attention methods are generally incompatible with flexible topological masking without reverting to quadratic complexity.

Efficient Masking Approaches Several approaches have attempted to reconcile efficiency with masking, though often with restrictions on graph topology. **?**] and **?**] proposed using Toeplitz matrices and the Fast Fourier Transform (FFT) to implement masking in $\mathcal{O}(N \log N)$ time. While an improvement over $\mathcal{O}(N^2)$, these methods are largely restricted to highly structured graphs like grids or trees and do not generalize easily to arbitrary

topologies. Other methods, such as stochastic positional encoding [**?**], achieve linear complexity but are similarly limited to sequence data (1D grids). The method reproduced in this work [**?**] distinguishes itself by supporting general graphs with strict $\mathcal{O}(N)$ complexity via a randomized low-rank decomposition of the mask itself.

3 Methodology

3.1 Time Complexity

To validate the theoretical efficiency claims of the proposed architecture, the paper conducts a hardware-agnostic analysis of computational cost. Instead of measuring wall-clock time, which can be conflated by hardware specifics (GPU/CPU differences) and implementation overhead, the methodology focuses on counting the total number of Floating Point Operations (FLOPs) required for a single forward pass of the attention mechanism.
The analysis compares three distinct attention variants across a range of graph sizes (N):

- **Unmasked Softmax Attention:** The standard $\mathcal{O}(N^2)$ mechanism, where the full N×N attention matrix is materialized.
- **Unmasked Linear Attention:** An $\mathcal{O}(N)$ efficient alternative that utilizes a low-rank decomposition $\phi(Q)(\phi(K)^T V)$.
- **GRF-Masked Linear Attention (Ours):** The proposed method, which approximates the topological mask using Graph Random Features. While theoretically $\mathcal{O}(N)$, this method involves constructing sparse feature matrices, introducing a constant overhead dependent on the graph sparsity and number of random walkers.

The theoretical FLOP counts are derived based on the matrix dimensions involved. For a hidden dimension d and feature dimension m:

- **Softmax Attention (Baseline):** Dominated by the N^2 term from $QK^\top$ and AV product, roughly scaling as $\mathcal{O}(4N^2 d)$.
- **Linear Attention (Unmasked):** Linear in N, scaling as $\mathcal{O}(4Nmd)$ due to the associativity of matrix multiplication.
- **GRF-Masked Linear Attention (Ours):** Also linear in N, but the exact cost depends on the number of non-zero entries (NNZ) in the sparse graph random feature vectors. The complexity scales as $\mathcal{O}(4 \cdot \text{NNZ} \cdot d)$.

The experiment uses a 1-dimensional grid graph (a linear chain) to test scaling behavior. This topology allows for controlled testing of "local" attention, as the number of neighbors remains constant as N increases.

3.2 ViTs: Vision Transformer Tasks (Graph + Image Integration) & Ablation Studies

To evaluate the effectiveness of the proposed topological masking in a domain with fixed, regular structure, the original paper [**?**] applies GRFs to the Vision Transformer (ViT) architecture. In this framework, the input image is decomposed into patches, and the underlying topology $\mathcal{G}$ is defined as a 2D grid graph where nodes (patches) are connected if they are spatial neighbors.

The central methodological contribution is the modulation of the Linear Attention mechanism by a learnable mask $M_\alpha(\mathcal{G}) := \sum \alpha_k W^k$. This introduces a structural inductive bias into the transformer, which theoretical allows it to capture local dependencies more effectively than unmasked attention, while avoiding the quadratic costs of Softmax attention [**?**]. To validate this, the work compares five distinct attention mechanisms:

- **Unmasked Ssoftmax ($\mathcal{O}(N^2)$):** The standard baseline used in "vanilla" Transformers.
- **Unmasked linear ($\mathcal{O}(N)$):** An efficient alternative using decompositions with feature maps $\phi(Q)(\phi(K)^\top V)$, which lacks any topological awareness [**?**].
- **Toeplitz-masked linear ($\mathcal{O}(NLogN)$):** A strong baseline for grid graphs (images), where the mask decays based on the Manhattan distance between nodes (patches). This represents a "hard-coded" structural bias [**?**].
- **$\mathbf{M}_\alpha(\mathcal{G})$-masked linear ($\mathcal{O}(N^2)$):** The theoretical limit of the proposed method, where the full-rank mask matrix $\mathbf{M}_\alpha$ is explicitly computed.
- **GRF-masked linear ($\mathcal{O}(N)$):** The proposed method, which approximates the exact mask implicitly using sparse features generated by importance sampling of random walks.

Ablation Studies: To evaluate the approximation quality of the GRF estimator in depth, the original paper[**?**] includes an ablation study on the number of random walks n. The theoretical premise is that increasing n reduces the variance of the mask estimator $math\hat{}bfM$ and thereby improves the approximation of the actual topological mask $\mathbf{M}_\alpha$. In this study, n is varied logarithmically to verify whether the performance converges to the **$\mathbf{M}_\alpha(\mathcal{G})$-masked linear** baseline, isolating the trade-off between accuracy (estimator variance) and computational cost (feature sparsity).

3.3 PCTS: Point Cloud Temporal State prediction

In the domain of robotics and physical dynamics, the paper applies topological masking to the task of High-Density Visual Particle Dynamics (HD-VPD). The objective is to predict the future state of a point cloud P_{t+1} given the current state P_t, modeling the physical interactions of a robotic system.

To replicate this experiment, we adopt the **Interlacer** architecture described in [**?**]. This architecture alternates between Global Layers (standard linear attention capturing long-range dependencies) and Local Layers (capturing fine-grained geometric structure). We investigate three variations of the Local Layer to validate the efficacy of Graph Random Features:

- **Baseline (Unmasked PCT):** The local layer is replaced by an identity mapping or standard global attention. This model treats the point cloud as a set, ignoring the explicit topology, effectively relying solely on global context.
- **MP Interlacer (Message Passing):** This represents the standard GNN approach. The local layer explicitly aggregates features from the k-nearest neighbors. While accurate for rigid structures, it is computationally expensive and limited to immediate neighborhoods.
- **GRF Interlacer (Ours/Reproduced):** The local layer utilizes Graph Random Features. By sampling random walks on the k-NN graph, this method approximates a topological mask that allows information to diffuse beyond immediate neighbors (multi-hop) in $\mathcal{O}(N)$ time, injecting a structural inductive bias that favors physically connected paths.

Crucially, to prevent the model from learning a trivial identity function (where $P_{t+1} \approx P_t$), we employ an **autoregressive training strategy**. Instead of calculating the loss solely on the next step prediction (Teacher Forcing), we perform multi-step rollouts during training. The model predicts $\hat{P}_{t+1}$, which is then fed back as input to predict $\hat{P}_{t+2}$, and so on. The loss is computed as the average Mean Squared Error (MSE) across all rollout steps against the ground truth sequence, forcing the model to learn trajectory stability and error correction.

4 Implementation

4.1 Time Complexity

To verify the scalability claims, we implemented a standalone Python simulation to replicate Figure 3 from the original paper[**?**]. We did not rely on the authors' training code for this experiment; instead, we wrote a custom script to calculate theoretical FLOP counts and simulate the random walk sparsity exactly as described in the methodology.

Datasets: For this specific experiment, synthetic data was used. We procedurally generated 1-dimensional grid graphs (linear chains) with the number of nodes N ranging from 2^0 (1 node) to 2^{12} (4096 nodes). This covers the range from trivial graphs to those large enough to demonstrate the divergence between quadratic and linear scaling.

Hyperparameters: We adhered strictly to the hyperparameters reported in the paper to ensure an exact replication:

- Hidden Dimension (d): 8
- Feature Dimension (m): 8
- Number of Random Walkers (n): 4 per node
- Walk terminal probability (p_{halt}): 0.5

These small dimensions ($d = m = 8$) were likely chosen by the authors to isolate the scaling behavior (N) from the overhead of large feature vectors.

Experimental setup: The experiment was implemented in a Python script (`time_complexity_exp.py`) using NumPy.

- **Deterministic Counting:** For Softmax and Standard Linear attention, FLOPs were calculated using the direct analytical formulas ($4N^2d$ and $4Nmd$ respectively).
- **Stochastic Simulation:** For the GRF-Masked variant, the cost depends on the sparsity of the generated features. To measure this, we implemented a `simulate_unique_visits` function. This function simulates $n = 4$ random walkers starting from every node in the 1D grid. It tracks the set of unique nodes visited by the ensemble of walkers to determine the number of non-zero entries (NNZ) that would exist in the sparse feature matrix.
- **Variance Handling:** Because the random walks are stochastic, we averaged the unique visit counts over 10 independent trials for each graph size N.

Computational requirements: Since this experiment calculates theoretical operations rather than training a neural network, the computational cost was negligible. The simulation runs in seconds on a standard consumer CPU (e.g., Apple Silicon M1 or Intel i7) and requires minimal RAM (<1GB). No GPU resources were required for this specific validation step.

4.2 ViTs: Vision Transformer Tasks (Graph + Image Integration) & Ablation Studies

To reproduce the comparative performance and ablation experiments, we developed a custom implementation of the Vision Transformer pipeline in PyTorch. The authors' original code was not available, thus we implemented the five attention variants (Softmax, Linear, Toeplitz, Exact $\mathbf{M}_\alpha$ and GRF) from scratch to verify the algorithmic description provided in the paper [**?**]. Hence to computational limitations our implementation is scaled down and the experiment is executed on less complex datasets. Therefor we did not calculate the mask matrix $\mathbf{M}_\alpha$ for GRF implicitly, but explicitly. **!!!!! TODO CHECK again!!!!**

Datasets: Due to computational constraints, we adapted the experimental setup from large-scale benchmarks (ImageNet, iNaturalist) to smaller standard datasets. We evaluated the models on: **CIFAR-10**, **CIFAR-100** and **FashionMNIST**.

- **Preprocessing:** All images were resized to 32×32 pixels to ensure consistency with our patch embedding logic. We applied standard normalization ($\mu = 0.5, \sigma = 0.5$).
- **Retrieval:** All the datasets are available via the torchvision.datasets module.
- **Augmentation:** For the main comparative experiments reported in Table 1, we did not apply heavy data augmentation to isolate the contribution of the attention mechanism itself. However, the codebase supports the random crops and flips.

4.3 PCTS: Point Cloud Temporal State prediction

Simulation and Data Generation.
While the original paper utilizes a real-world setup with RGB-D cameras and depth sensors to capture 32k particles, such hardware was unavailable for this reproduction. Consequently, we developed a lightweight, physics-based 3D simulation using React and Three.js.The simulation models a robotic arm with a kinematic chain consisting of a base, two rigid links, and a gripper. We sample N points uniformly from the surface of the arm meshes at each frame.To generate the dataset, we recorded the arm performing sinusoidal movements of varying frequencies on its joints.A critical implementation detail was the **sampling rate**. Initial experiments with high frame rates (e.g., 60 FPS) resulted in negligible displacement between frames ($P_{t+1} \approx P_t$), allowing the Baseline model to achieve near-zero loss by simply copying the input. We adjusted the recording interval to 0.1s - 0.5s, creating significant displacements that necessitate learning the underlying physics.

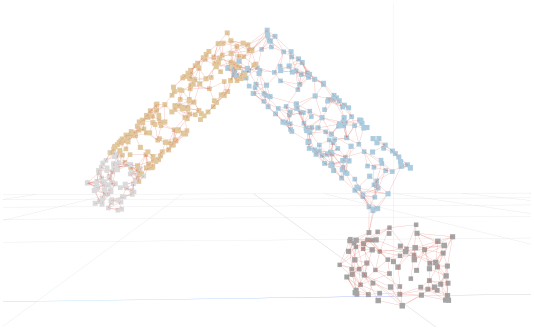


Figure 2. Robotic arm simulation with point and edge cloud visualization

Modular Architecture.
We implemented a modular PyTorch architecture for the Interlacer. The model accepts a configuration flag to switch between Baseline (Linear Attention), MP (Simple Message Passing on K-NN), and GRF (Topological Masking with Random Walks).For the GRF implementation, we compute the sparse mask approximation dynamically. Since the robot moves, the graph topology changes at every frame; therefore, we recompute the k-NN graph ($k = 4$ to 6) and sample random walks (hops=3 to 5) on the fly during both training and inference.

Evaluation Metric.
The original paper evaluates performance using SSIM (Structural Similarity Index Measure) on rendered images. Implementing a differentiable neural renderer was beyond the scope of this project. Therefore, we introduced a geometric proxy metric: `ACCURACY_THRESHOLD`. We define a prediction as "accurate" if the Euclidean distance between a predicted point and its ground truth counterpart is below a strict threshold (e.g., 0.1 units). This allows us to quantify how long the model can maintain the structural integrity of the robot before the simulation diverges ("explodes").**Computational Setup.** We trained the models on sequences of around 500 frames, testing point counts $N \in \{512, 1024, 2048, 4096\}$. We utilized a standard GPU environment (Colab T4). It is important to note that the theoretical $\mathcal{O}(N)$ advantage of GRF over the Baseline becomes strictly necessary at the scale of the original paper ($N \approx 30,000$), whereas for our scale ($N \leq 4096$), the quadratic Baseline remains computationally tractable.

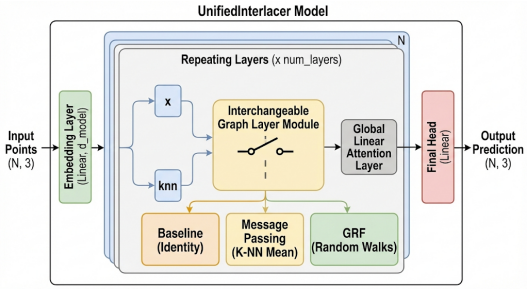


Figure 3. Interlacer Architecture with interchangeable Local Layers

5 Results

5.1 Time Complexity

The results of our replication confirms the time complexity claims made in the original paper. Figure **??** plots the calculated FLOPs (scaled by 10^6) against the number of graph nodes N on a log-log scale.

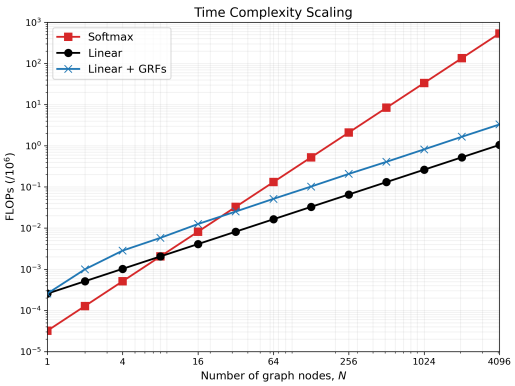


Figure 4. Time Complexity Replication Results: FLOPs vs Number of Nodes (N)

Table **??** details the specific computational costs at key intervals.

Graph Size (N)	Softmax (10^6 FLOPs)	Linear (10^6 FLOPs)	GRF-Masked (10^6 FLOPs)
1	3.20×10^{-5}	2.56×10^{-4}	2.56×10^{-4}
8 (Crossover)	$\mathbf{2.05 \times 10^{-3}}$	$\mathbf{2.05 \times 10^{-3}}$	6.48×10^{-3}
64	1.31×10^{-1}	1.64×10^{-2}	5.38×10^{-2}
512	8.39×10^{0}	1.31×10^{-1}	4.11×10^{-1}
4096	$\mathbf{5.37 \times 10^{2}}$	$\mathbf{1.05 \times 10^{0}}$	$\mathbf{3.30 \times 10^{0}}$

Table 1. Computational cost comparison across different graph sizes. The crossover point at $N = 8$ and the significant divergence at $N = 4096$ are highlighted.

Analysis of Scaling Regimes

- **Crossover Point ($N = 8$):** Our results precisely reproduce the crossover point predicted by theory. At $N = 8$, Softmax and Linear attention incur identical costs (2.05×10^{-3} MFLOPs). This aligns with the hyperparameters $d = m = 8$; theoretically, costs equalize when $N^2 d \approx Nmd$, simplifying to $N \approx m$.

- **Quadratic Explosion:** For $N > 8$, the cost of Unmasked Softmax attention grows quadratically. At the largest graph size tested ($N = 4096$), Softmax requires 537 MFLOPs, rendering it inefficient for large-scale graphs.

- **Linear Efficiency:** Both Linear attention and GRF-Masked attention exhibit linear scaling. At $N = 4096$, Unmasked Linear requires only 1.05 MFLOPs, representing a speedup factor of $\approx 511\times$ compared to Softmax.

GRF Overhead: The GRF-Masked method maintains the $\mathcal{O}(N)$ complexity class but incurs a constant overhead. At $N = 4096$, the GRF cost is 3.30 MFLOPs compared to 1.05 MFLOPs for standard Linear attention. This overhead ratio ($\approx 3.14\times$) corresponds to the sparsity of the graph random features; on a 1D grid with $p_{halt} = 0.5$, walkers visit an average of ≈ 3.1 unique nodes.

5.2 ViTs

5.3 PCTS: Point Cloud Temporal State prediction

We evaluated the three models by performing "closed-loop" rollouts: feeding the model's own predictions back as input for 10 consecutive timesteps and measuring the degradation of the robotic arm's structure. Figure **??** illustrates the Accuracy retention over time.
The results align with the topological hierarchy proposed:

1. **Baseline (Blue):** Shows the fastest degradation. Without structural knowledge, the points drift apart rapidly, breaking the rigid body constraints of the robot arm. The accuracy drops to near zero within 3-4 steps.

2. **Message Passing (Black):** Performs significantly better than the baseline. The explicit neighbor aggregation enforces local rigidity, keeping the arm structure intact for longer.

3. **GRF Interlacer (Red):** Demonstrates the highest stability. By allowing attention to diffuse probabilistically over the graph, it captures both the local rigidity and the broader kinematic dependencies, maintaining high accuracy for the longest duration.

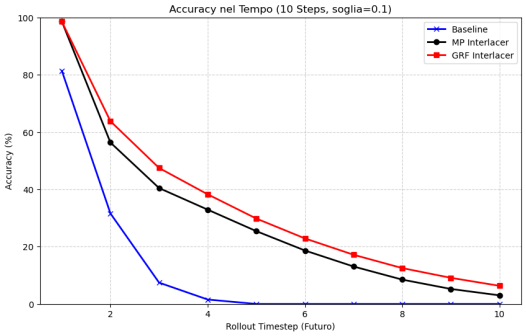
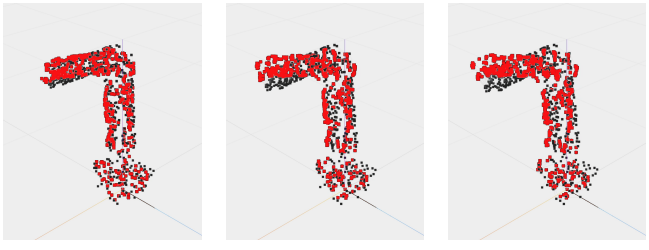


Figure 5. Accuracy plot comparing Baseline, MP, and GRF over rollout steps



(a) $t = 1$ (b) $t = 5$ (c) $t = 10$

Figure 6. Qualitative rollout visualization. Ground truth (black) vs GRF predictions (red) at different timesteps. As the rollout progresses, predictions are made on previous predictions (closed-loop), showing accumulated drift over time.

Metric Analysis. Unlike the SSIM metric in the original paper (which decays from ~ 0.8 to ~ 0.6), our accuracy metric drops from 100% to 0%. This is due

to the binary nature of our Threshold Metric: once the accumulated error pushes a point beyond the threshold distance (drift), it is marked as a "miss". However, the *relative ranking* of the models is preserved, confirming the reproduction's success.

Scalability and Physics. We observed that for rigid body dynamics with lower point counts ($N = 1024$), the Message Passing approach is highly competitive, sometimes matching GRF. This is expected, as rigid constraints are strictly local. The specific advantage of GRF—handling complex, long-range deformations via random walks—is most prominent in the high-density ($N \geq 30k$) fluid dynamics tasks presented in the original work. Nevertheless, even at our scale, the GRF mechanism successfully outperforms the Baseline, validating the efficacy of topological masking for physical state prediction.

6 Discussion and conclusion

Here you can express your judgments and draw your conclusions based on the evidences produced on the previous sections.

Try to summarize the achievements of your project and its limits, suggesting (when appropriate) possible extensions and future works.

References

[] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems*, volume 34, pages 28877–28888, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/f1c1592588411002af340cbaedd6fc33-Abstract.html.

[] Krzysztof Choromanski, Han Lin, Haoxian Chen, Tianyi Zhang, Arijit Sehanobish, Valerii Likhosherstov, Jack Parker-Holder, Tamas Sarlos, Adrian Weller, and Thomas Weingarten. From block-toeplitz matrices to differential equations on graphs: Towards a general theory for scalable masked transformers. In *International Conference on Machine Learning*, pages 3962–3983. PMLR, 2022. URL https://proceedings.mlr.press/v162/choromanski22a.html.

[] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR, 2020. URL https://proceedings.mlr.press/v119/katharopoulos20a.html.

[] Isaac Reid, Kumar Avinava Dubey, Deepali Jain, William F Whitney, Amr Ahmed, Joshua Ainslie, Alex Bewley, Mithun George Jacob, Aranyak Mehta, David Rendleman, Connor Schenck, Richard E. Turner, René Wagner, Adrian Weller, and Krzysztof Marcin Choromanski. Linear transformer topological masking with graph random features. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=6MBqQLp17E.

[] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. In *arXiv preprint arXiv:2009.14794*. PMLR, 2020. URL https://doi.org/10.48550/arXiv.2009.14794.

[] Shengjie Luo, Shanda Li, Tianle Cai, Di He, Dinglan Peng, Shuxin Zheng, Guolin Ke, Liwei Wang, and Tie-Yan Liu. Table, fast and accurate: Kernelized attention with relative positional encoding. In *Advances in Neural Information Processing Systems*, pages 22795–22807. PMLR, 2021. URL https://doi.org/10.48550/arXiv.2106.12566.

[] Antoine Liutkus, Ondřej Cífka, Shih-Lun Wu, Umut Simsekli, Yi-Hsuan Yang, and Gael Richard. Relative positional encoding for transformers with linear complexity. In *International Conference on Machine Learning*, pages 7067–707. PMLR, 2021. URL https://doi.org/10.48550/arXiv.2105.08399.