LET THE HEADS TALK: BEYOND DIAGONAL GRAPH ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Sheaf Neural Networks generalize scalar-weighted message passing by replacing scalar edge weights with linear transport maps between local feature spaces. Yet the role of this matrix-valued transport is entangled with the broader sheaf-diffusion construction. We isolate the transport primitive through quiver representations and establish a direct connection with multi-head attention. Treating attention heads as coordinates of a local transport space reveals that standard multi-head attention implements diagonal edge maps: along each directed interaction, a source head can contribute only to the corresponding receiver head. Allowing off-diagonal entries instead enables edge-conditioned communication across heads before neighborhood aggregation. We show that this operation cannot, in general, be absorbed into a single shared linear map applied after aggregation. Building on this characterization, we introduce **Topological Attention** (Top-A), a multi-head attention that learns edge-dependent off-diagonal routes while preserving the original same-head paths and exactly recovering vanilla attention when the additional routing vanishes. We evaluate Top-A on relational reasoning, heterogeneous graph learning, and algorithmic reasoning, including out-of-distribution generalization, with heterophilic node classification as a contrast setting. The results show that cross-head transport is most useful when the task benefits from interaction-dependent transformations, while heterophily alone provides no systematic advantage. These findings identify edge-conditioned cross-head communication as a distinct computational primitive of matrix-valued transport.

1 INTRODUCTION

Graph Neural Networks (GNNs) (Scarselli et al., 2009; Micheli, 2009; Defferrard et al., 2016; Gilmer et al., 2017; Kipf & Welling, 2017) learn by propagating information along graph edges, with most message-passing architectures using scalar weights to modulate neighboring representations. Sheaf Neural Networks (SNNs) (Hansen & Gebhart, 2020; Bodnar et al., 2022) generalize this paradigm by equipping edges with linear transport operators between local feature spaces, allowing neighboring representations to be transformed rather than merely reweighted. Introduced through cellular sheaf theory (Hansen & Ghrist, 2019), this richer form of local transport was motivated by phenomena such as heterophily (Luan et al., 2024) and oversmoothing (Cai & Wang, 2020; Oono & Suzuki, 2020; Rusch et al., 2023), and has since inspired a growing family of architectures. Yet, the classical sheaf construction couples several components: restriction maps connect node and edge spaces, their compositions induce transport between neighboring nodes, and the resulting transformations are assembled through the sheaf Laplacian. Consequently, the specific computational capability contributed by richer local transport remains unclear. While the original motivation for SNNs is rooted in sheaf diffusion and topology, recent architectures increasingly learn edge-dependent matrix transformations directly (Hajij et al., 2025; Ribeiro et al., 2026). Moreover, recent evidence questions whether gains under heterophily and oversmoothing arise specifically from the learned sheaf structure (Hernandez Caralt et al., 2026; Fiorini et al., 2026). This motivates a more fundamental question: *what is actually useful about sheaf transport?* We address it by isolating the local transformation induced by learned sheaf structure. Rather than asking when SNNs outperform conventional GNNs, we ask which additional form of message passing matrix-valued transport enables and how it differs from standard graph propagation. This leads us from cellular sheaves to quiver representations (Hajij et al., 2025), where directed interactions are represented directly by linear maps. At this level, transport

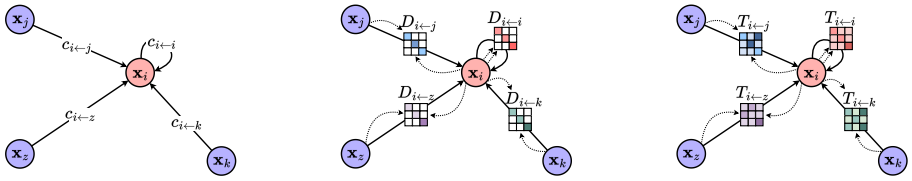
Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

itself can be studied independently of the restriction-map factorization and diffusion dynamics of classical SNNs, while revealing a direct connection with multi-head attention. If attention heads are treated as coordinates of the local transport space, multi-head attention induces a diagonal map on each directed interaction: each source head contributes only to the corresponding receiver head, even though different heads may assign different weights to the same interaction. Thus, standard attention spans only the diagonal subfamily of matrix-valued head-space transport. Off-diagonal entries extend this mechanism with edge-conditioned communication across heads before neighborhood aggregation. We show that this edge-specific mixing cannot, in general, be absorbed into a shared post-aggregation projection, and therefore constitutes a distinct operation beyond standard multi-head attention. Motivated by this, we introduce **Topological Attention** (Top-A), which augments standard attention with edge-conditioned off-diagonal routing across heads while preserving the original same-head paths. When the routing vanishes, Top-A reduces exactly to vanilla attention, providing a controlled way to study when cross-head transport is useful independently of other architectural changes. We evaluate this inductive bias on algorithmic reasoning with CLRS (Veličković et al., 2022), relational reasoning with STaR (Khalid & Schockaert, 2025), heterogeneous graph learning (Harper & Konstan, 2015), and, as a contrast setting, classical heterophilic node classification (Platonov et al., 2023). Together, these benchmarks probe edge-specific computation across relational composition, directed heterogeneous interactions, and algorithmic size generalization, while isolating whether heterophily alone is sufficient to make cross-head transport beneficial.

Our main contributions are: **(i)** We use quiver representations to disentangle matrix-valued edge transport from the broader sheaf-diffusion construction, establishing a direct connection to multi-head attention. **(ii)** We characterize matrix-valued transport as edge-conditioned communication across attention heads and prove that, in general, it cannot be reduced to a shared post-aggregation projection. **(iii)** We introduce Top-A, an architectural extension implementing this transport primitive, and evaluate it across relational, heterogeneous, and algorithmic reasoning, including compositional- and size-generalization regimes.

2 Background and Related Work

We start by providing the required background on graph models, SNNs, graph quivers, and graph attention as message passing. A deeper discussion of related work is provided in Appendix A.



(a) Convolutional (b) Multi-head Attention (c) **Top-A (ours)**

Figure 1: **From Scalar Weighting to Matrix-valued Transport.** Comparison of three graph neural network paradigms for a directed interaction $j \to i$, with sender j and receiver i. *Convolutional* layers weight incoming features with fixed scalar coefficients $c_{i\leftarrow j}$. *Attentional* layers replace these constants with data-dependent coefficients $\alpha_{i\leftarrow j} = a(\mathbf{x}_i, \mathbf{x}_j)$. *Topological Attention* replaces the diagonal head-space operator $D_{i\leftarrow j}$ with the matrix-valued transport $T_{i\leftarrow j} = D_{i\leftarrow j}(I_H + \lambda\Omega_{i\leftarrow j})$, where $\Omega_{i\leftarrow j}$ contains off-diagonal routes between attention heads. Standard multi-head attention is recovered when $\Omega_{i\leftarrow j} = 0$.

Message Passing Neural Networks. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ the set of edges. We write a directed interaction from sender node j to receiver node i as $j \to i$, and define the incoming neighborhood of node i as $\mathcal{N}(i) = \{j : (j, i) \in \mathcal{E}\}$. Each node in the graph is associated with a hidden state vector $\mathbf{x}_i \in \mathbb{R}^d$, which provides the node representation. GNNs are functions that learn the complex relational information of a graph, typically framed through the Message Passing Neural Network (MPNN) framework (Gilmer et al., 2017; Bronstein et al., 2021) where the representation for a node u at layer ℓ is computed as:

$$\mathbf{x}_i^{(\ell)} = \phi^{(\ell)}\left(\mathbf{x}_i^{(\ell-1)}, \bigoplus^{(\ell)}\left\{\psi^{(\ell)}\left(\mathbf{x}_j^{(\ell-1)}, \mathbf{x}_i^{(\ell-1)}\right) : j \in \mathcal{N}(i)\right\}\right) \tag{1}$$

where $\psi^{(\ell)}(\mathbf{x}_j, \mathbf{x}_i)$ computes the message $m_{i\leftarrow j}^{(\ell)}$ from sender j to receiver i, $\bigoplus^{(\ell)}$ aggregates the messages in a permutation-invariant way, and $\phi^{(\ell)}$ updates the node representation. Many widely used GNNs instantiate this framework by associating each neighbor with a scalar coefficient that modulates its contribution to the aggregated representation (Kipf & Welling, 2017; Veličković et al., 2018), as shown in Figures 1a and 1b.

Sheaf Neural Networks. SNNs (Hansen & Gebhart, 2020; Bodnar et al., 2022) enrich graph message passing by associating nodes and edges with local vector spaces, called *stalks*, together with linear maps between them. Given a cellular sheaf $\mathcal{F}$ over $\mathcal{G}$, each node $i \in \mathcal{V}$ and edge $e \in \mathcal{E}$ is assigned a stalk $\mathcal{F}(i) = \mathbb{R}^d$ and $\mathcal{F}(e) = \mathbb{R}^d$, respectively, while a restriction map $F_{i\to e} : \mathcal{F}(i) \to \mathcal{F}(e)$ is associated with every node-edge incidence. For two adjacent nodes i and j sharing an edge e, these restriction maps induce the node-to-node linear operator:

$$T_{i\leftarrow j} = F_{i\to e}^{\top} F_{j\to e} : \mathcal{F}(j) \to \mathcal{F}(e) \to \mathcal{F}(i) \tag{2}$$

which transforms the representation of node j before it is aggregated at node i. Neural Sheaf Diffusion (NSD) (Bodnar et al., 2022) builds the propagation dynamics from the resulting sheaf Laplacian, providing a principled form of matrix-valued local communication between neighboring representations.

Graph Quivers. The node-to-node transport induced by SNNs can also be represented directly, without retaining its factorization through the edge stalk. A natural formalism for doing so is a representation of the directed graph quiver (Gabriel, 1972; Derksen & Weyman, 2017). Let the computational graph be viewed as a quiver whose vertices are the nodes of $\mathcal{G}$ and whose arrows are the directed interactions $e : j \to i$, with source j and target i. A quiver representation assigns a vector space $\mathcal{Q}(i)$ to every vertex and a linear map:

$$T_{i\leftarrow j} : \mathcal{Q}(j) \longrightarrow \mathcal{Q}(i) \tag{3}$$

to every directed arrow $j \to i$. The corresponding neighborhood transport takes the form:

$$\mathbf{x}_i^{(\ell)} = \sum_{j\in\mathcal{N}(i)} T_{i\leftarrow j}^{(\ell)} \mathbf{x}_j^{(\ell-1)} \tag{4}$$

The sheaf-induced message in Equation (2) is recovered by choosing:

$$T_{i\leftarrow j} = F_{i\to e}^{\top} F_{j\to e} : \mathcal{F}(j) \to \mathcal{F}(i) \tag{5}$$

The quiver representation therefore isolates the effective directed transport $T_{i\leftarrow j}$ from the specific restriction-map factorization and diffusion operator that generate it in classical SNNs. Accordingly, we adopt this abstraction throughout the paper: we study the linear map carried directly by each oriented interaction, following the perspective developed by Hajij et al. (2025), who cast topological neural message passing through local maps assigned to quiver arrows.

Graph Attention as Message Passing. Attention-based GNNs (Veličković et al., 2018; Dwivedi & Bresson, 2021; Brody et al., 2022; Rampášek et al., 2022; Shi et al., 2021) make a sender's contribution depend on its interaction with the receiver. This includes Graph Attention Networks (GATs) (Veličković et al., 2018; Brody et al., 2022), which typically attend over local neighborhoods, and Graph Transformers (GTs) (Dwivedi & Bresson, 2021; Rampášek et al., 2022; Shi et al., 2021), which combine attention with Transformer-style architectures and structural information.

Omitting layer superscripts, let $V_j^{(h)} \in \mathbb{R}^C$ be the value produced by source head h, and $s_{i\leftarrow j}^{(h)}$ its score for the interaction $j \to i$. Receiver-wise normalization gives:

$$\alpha_{i\leftarrow j}^{(h)} = \frac{\exp s_{i\leftarrow j}^{(h)}}{\sum_{k\in\mathcal{N}_{\mathrm{att}}(i)} \exp s_{i\leftarrow k}^{(h)}} \tag{6}$$

where $\mathcal{N}_{\mathrm{att}}(i) = \mathcal{N}(i)$ for methods such as Veličković et al. (2018) and $\mathcal{N}_{\mathrm{att}}(i) = \mathcal{V}$ for transformer-like architectures, e.g., Shi et al. (2021). The corresponding head output is:

$$M_i^{(h)} = \sum_{j\in\mathcal{N}_{\mathrm{att}}(i)} \alpha_{i\leftarrow j}^{(h)} V_j^{(h)}, \tag{7}$$

where the scoring rule and attended neighborhood depend on the architecture. This formulation leaves the joint H-head representation space implicit. The quiver perspective makes it explicit, allowing us to identify the edge map induced by standard multi-head attention, which we characterize next. Appendix A provides a more in depth comparison with more recent attention mechanisms.

3 Attention as Diagonal Head-Space Transport

The quiver formulation introduced in Section 2 describes message passing through linear maps attached to directed interactions. Multi-head attention already comes with a natural local space on which such maps can act: the space indexed by its H attention heads. We therefore take the heads as the coordinates of the local transport space, while the C-dimensional representation carried by each head is propagated in parallel. In this scenario, for a sender j, we stack the value representations, $V_j^{(h)}$, introduced in Section 2 as $V_j = [(V_j^{(1)})^\top, \dots, (V_j^{(H)})^\top]^\top \in \mathbb{R}^{H\times C}$, where $V_j^{(h)} \in \mathbb{R}^C$ denotes the value carried by head h as before. For the directed interaction $j \to i$, the corresponding attention coefficients define a diagonal matrix of the form:

$$D_{i\leftarrow j} = \mathrm{Diag}\left(\alpha_{i\leftarrow j}^{(1)}, \dots, \alpha_{i\leftarrow j}^{(H)}\right) \in \mathbb{R}^{H\times H} \tag{8}$$

This diagonal organization of attention over the head space is illustrated in Figure 1b. The H headwise updates in Equation (7) can be written as:

$$M_i^{\mathrm{att}} = \sum_{j\in\mathcal{N}_{\mathrm{att}}(i)} D_{i\leftarrow j} V_j \tag{9}$$

The form expressed in Equation (9) makes the connection with quivers explicit. Once the attention coefficients are computed, they define a linear map on each directed interaction. Taking $\mathcal{Q}(i) = \mathbb{R}^H$ as the local space at each node, the interaction $j \to i$ carries the linear map:

$$T_{i\leftarrow j}^{\mathrm{att}} = D_{i\leftarrow j} : \mathbb{R}^H \longrightarrow \mathbb{R}^H \tag{10}$$

The within-head coordinates are carried along unchanged by this head-space map. Under the decomposition $\mathbb{R}^{HC} \simeq \mathbb{R}^H \otimes \mathbb{R}^C$, the corresponding operator on the full value representation is therefore $D_{i\leftarrow j} \otimes I_C$.

Proposition 3.1 (Multi-head attention as diagonal transport)**.** *Before any shared post-aggregation readout, normalized multi-head attention is diagonal transport on its native head space. In particular, for every directed interaction $j \to i$ and receiver head $r \in \{1, \dots, H\}$:*

$$(D_{i\leftarrow j} V_j)^{(r)} = \alpha_{i\leftarrow j}^{(r)} V_j^{(r)} \tag{11}$$

where $(\cdot)^{(r)}$ denotes the r-th head component.

A proof of the proposition is given in Appendix B.2. Although different heads may assign different weights to the same interaction, the correspondence between source and receiver coordinates is fixed: information carried by source head r is passed only to receiver head r along that edge. Standard multi-head attention therefore occupies the diagonal subfamily of linear maps on the head space. Similarly, the multi-head copresheaf attention construction of Hajij et al. (2025) uses edge-dependent linear maps that act independently within the value-channel space of each head (see Appendix A for further discussion). Figure 1 summarizes this progression from scalar weighting to diagonal attention and, ultimately, matrix-valued head-space transport.

Beyond diagonal head transport. The same representation also makes clear what lies outside this diagonal family. Consider a general linear map associated with the interaction $j \to i$, $T_{i\leftarrow j} \in \mathbb{R}^{H\times H}$. Its action on the sender values is:

$$[T_{i\leftarrow j} V_j]^{(r)} = \sum_{s=1}^{H} T_{i\leftarrow j}^{rs} V_j^{(s)} \tag{12}$$

An off-diagonal entry $T_{i\leftarrow j}^{rs}$, with $r \neq s$, allows information carried by source head s to contribute to receiver head r on that interaction. Since the map is attached to a directed edge, these cross-head correspondences may vary across the neighborhood: two senders connected to the same receiver can induce different transformations before their contributions are aggregated. On the full value representation, the edge operator becomes $T_{i\leftarrow j} \otimes I_C$: it mixes information across heads while leaving the C-dimensional coordinates within each head unchanged. Standard attention is recovered by setting $T_{i\leftarrow j} = D_{i\leftarrow j}$, while the off-diagonal entries of $T_{i\leftarrow j}$ introduce the additional edge-dependent routes through which information can move between heads before aggregation, as illustrated in Figure 2.

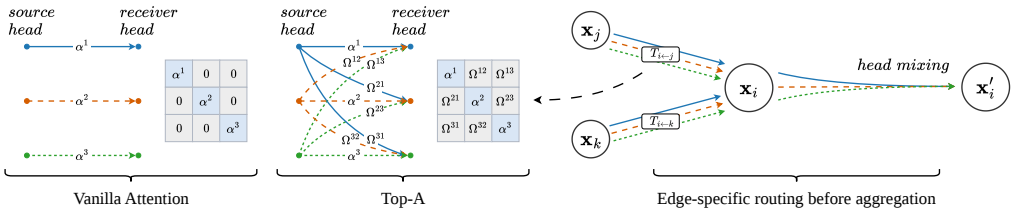


Figure 2: **Vanilla attention versus Top-A Cross-head Routing.** Standard multi-head attention restricts information transfer to corresponding source and receiver heads, yielding a diagonal transport structure. Top-A generalizes this mechanism through learnable cross-head coefficients Ω^{rs}, enabling source head s to contribute to receiver head r. The resulting edge-specific transport $T_{i\leftarrow j}$ is applied before neighborhood aggregation, followed by head mixing to produce the updated representation h_i'.

Section 4 defines a parameterization of these routes that preserves the original diagonal attention paths.

Relation to sheaf transport. The head-space perspective also sharpens the connection to the sheaf construction of Section 2. Identifying node stalk coordinates with attention heads, the sheaf-induced map $T_{i\leftarrow j} = F_{i\to e}^{\top} F_{j\to e}$ then defines a matrix-valued transport over the same head space. Its off-diagonal entries therefore have the same computational role described above: they route information across different head coordinates. For an undirected edge, however, the two directional transports are coupled through the restriction maps and satisfy $T_{j\leftarrow i} = T_{i\leftarrow j}^{\top}$. By contrast, the directed maps studied here need not to satisfy this transpose-reciprocity constraint. At the level of local operators, classical sheaf transport is therefore an undirected instance of matrix-valued head-space transport, whereas the quiver formulation assigns independent maps to the two directed interactions. Appendix B.1 formalizes this connection and proves that symmetric multi-head attention can be realized exactly as diagonal cellular-sheaf transport. Together, these observations connect standard attention to the diagonal component of sheaf transport and motivate the more general directed, off-diagonal head-space maps used in Top-A.

4 Topological Attention

Section 3 shows that standard multi-head attention is restricted to diagonal transport in head space. In this section we present Topological Attention (Top-A) that extends this mechanism with edge-dependent communication across different heads, while retaining the original same-head attention paths. These cross-head routes are applied independently to each directed interaction before the incoming messages are aggregated.

4.1 Edge-conditioned cross-head transport

For a directed interaction $e = (j \to i)$, write $D_e := D_{i\leftarrow j}$, and let $\Omega_e := \Omega_{i\leftarrow j} \in \mathbb{R}^{H\times H}$ denote a routing matrix over the head space. We constrain $\mathrm{diag}(\Omega_e) = 0$[1] so that the additional branch acts only between different heads. Starting from the diagonal attention operator D_e, we define the Top-A transport associated with e as $T_e := T_{i\leftarrow j}$:

$$T_e = D_e\,(I_H + \lambda\Omega_e)\,, \qquad \lambda \geq 0 \tag{13}$$

where λ controls the strength of the cross-head routing. The resulting operator T_e replaces D_e in the edge message. Considering a receiver head r, the corresponding component of the incoming representation at node i is

$$\left[M_i^{Top-A}\right]^{(r)} = \left[\sum_{j\in\mathcal{N}_{\mathrm{att}}(i)} T_{i\leftarrow j} V_j\right]^{(r)} = \sum_{j\in\mathcal{N}_{\mathrm{att}}(i)} \alpha_{i\leftarrow j}^{(r)} \left[V_j^{(r)} + \lambda \sum_{\substack{s=1\\ s\neq r}}^{H} \Omega_{i\leftarrow j}^{rs} V_j^{(s)}\right]. \tag{14}$$

[1]Here and throughout the paper, we use $\mathrm{diag}(A)$ to denote the vector of diagonal entries of a matrix A, and $\mathrm{Diag}(v)$ for the diagonal matrix whose diagonal is v.

The first term is the usual same-head contribution, while the second aggregates cross-head contributions from source heads $s \neq r$ through $\Omega^{rs}_{i\leftarrow j}$. Since D_e acts on the left in Equation (13), $\alpha^{(r)}_{i\leftarrow j}$ scales the entire message entering receiver head r, including cross-head contributions. Attention therefore retains its receiver-side weighting role, while Ω_e determines how source heads are combined before entering that receiver head. Since $\mathrm{diag}(\Omega_e) = 0$, the routing branch affects only cross-head interactions, leaving the original attention coefficients on the diagonal, i.e., $\mathrm{diag}(T_e) = \mathrm{diag}(D_e)$. When $\Omega_e = 0$, Equation (13) reduces exactly to standard multi-head attention. Figure 2 illustrates this distinction, contrasting the diagonal head-to-head paths of standard attention with the edge-specific cross-head routes introduced by Top-A before aggregation.

Learning the routing matrix. Let $\xi_e \in \mathbb{R}^p$ denote a representation of the directed interaction e, constructed from quantities already available to the attention layer. In Top-A, we compute the routing matrix using a learned map $g_\theta : \mathbb{R}^p \to \mathbb{R}^{H\times H}$, as

$$\Omega_e = \mathrm{off}(\tanh g_\theta(\xi_e)), \qquad \mathrm{off}(A) = A - \mathrm{Diag}(\mathrm{diag}(A)). \tag{15}$$

The operator $\mathrm{off}(\cdot)$ removes the diagonal, ensuring that the learned routing acts only between different heads, while the elementwise $\tanh$ bounds the routing coefficients and allows them to take either sign. Since ξ_e is specific to a directed interaction, different edges can induce different routing patterns: $j \to i$ and $i \to j$ need not share the same map, nor do different senders connected to the same receiver. Backbone-specific definitions of ξ_e are given in Appendix B.4.

4.2 Routing Before Neighborhood Aggregation

We next investigate whether a post-aggregation linear projection can reproduce the same cross-head mixing. The key distintion lies in operational order: such a project acts only after all incoming messages have been combined, whereas Top-A applies an interaction-specific head-space map to each incoming message before aggregation. To formalize this distinction, we stack sender j's H head values as $v_j := \left[(V_j^{(1)})^\top, \ldots, (V_j^{(H)})^\top\right]^\top \in \mathbb{R}^{HC}$. The corresponding operators on the concatenated representation are

$$\overline{D}_{i\leftarrow j} = D_{i\leftarrow j} \otimes I_C, \qquad \overline{T}_{i\leftarrow j} = T_{i\leftarrow j} \otimes I_C.$$

Theorem 4.1 (Shared post-aggregation equivalence)**.** *Fix a receiver i and the edge operators $\{\overline{D}_{i\leftarrow j}, \overline{T}_{i\leftarrow j}\}_{j\in\mathcal{N}_{\mathrm{att}}(i)}$. A single post-aggregation map reproduces Top-A's routed aggregation for every assignment of sender values if and only if some linear map $A \in \mathbb{R}^{HC\times HC}$ satisfies*

$$\sum_{j\in\mathcal{N}_{\mathrm{att}}(i)} \overline{T}_{i\leftarrow j} v_j = A \sum_{j\in\mathcal{N}_{\mathrm{att}}(i)} \overline{D}_{i\leftarrow j} v_j, \tag{16}$$

if and only if the same map A reproduces the transport on every incoming edge:

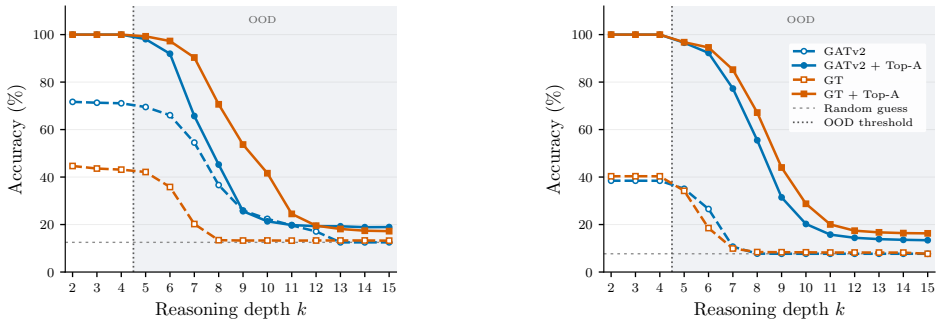
$$\overline{T}_{i\leftarrow j} = A\overline{D}_{i\leftarrow j} \qquad \text{for every } j \in \mathcal{N}_{\mathrm{att}}(i). \tag{17}$$

Theorem 4.1 gives an exact criterion for replacing edge-specific pre-aggregation routing with a shared linear map after standard attention aggregation, for fixed attention and transport operators. Because this map acts only after all incoming messages have been merged, it must implement the routed transformation of every incoming edge through the same left factor. Importantly, we allow A to be any receiver-specific linear map, so the comparison is strictly more permissive than the standard output projection used by the attention backbone.

For the transport in Equation (13), ordinary unmasked softmax attention with finite logits makes $D_{i\leftarrow j}$ invertible. Hence, Equation (17) is equivalent to

$$A = \left[D_{i\leftarrow j}\,(I_H + \lambda\Omega_{i\leftarrow j})\,D^{-1}_{i\leftarrow j}\right] \otimes I_C \tag{18}$$

for every incoming sender j. A shared post-aggregation map therefore exists only when all incoming edges induce the same effective transformation. Because $\Omega_{i\leftarrow j}$ may vary across directed interaction, Top-A neither enforces nor generally satisfies this condition. The underlying limitation is information loss: aggregation erases the identity of the individual incoming messages. A shared readout can mix only the combined representation, whereas Top-A transforms each edge separately beforehand. This ordering produces the essential computational advantage of pre-aggregation routing. Appendix B.3 provides the full proof and a two-edge example.



(a) RCC-8 (b) Interval Algebra

Figure 3: *Depth Generalization on STaR.* For each reasoning depth k, accuracy is averaged over $b \in \{1, 2, 3\}$ and three seeds. Models perform 15 recurrent rounds with shared-weights and are trained on $k \in \{2, 3, 4\}$; shading denotes unseen depths. Solid lines denote Top-A, and dashed lines the corresponding baselines.

5 Experiments

We evaluate the proposed head-space transport across complementary computational settings. In Section 5.1, a controlled relational reasoning benchmark tests whether edge-conditioned cross-head transport improves relation classification, including at reasoning depths beyond those seen during training. In Section 5.2, a real-world heterogeneous graph task assesses whether communication across head subspaces provides a useful inductive bias for integrating information from nodes and relations with distinct semantics. In Section 5.3, we look into algorithmic reasoning tasks, investigating which algorithmic procedures benefit from the addedrouting capacity of Top-A and whether these gains extend to larger problem instances. As a contrast setting, Appendix C.4 evaluates whether cross-head transport also benefits standard heterophilic node-classification benchmarks. Task details, including hyperparameter settings and search spaces, are reported in Appendices C.1, C.2, and C.3.

Attention Backbones. Because Top-A is agnostic to the underlying attention mechanism, we instantiate it with GATv2 (Brody et al., 2022) and a Graph Transformer (GT) (Shi et al., 2021), comparing each backbone with its corresponding Top-A variants. In both architectures, the interaction representation ξ_e in Equation (15) is constructed from quantities already available to the respective attention mechanism. For GATv2, ξ_e uses the same directed sender–receiver context as attention computation; for GT, it combines the receiver query and the edge-conditioned sender key. The resulting routing matrix Ω_e remains specific to each directed interaction, allowing distinct edges to induce distinct cross-head transformations. The precise backbone-specific constructions are provided in Appendix B.4, while initialization and computational cost are discussed in Appendix B.5.

5.1 Relational Reasoning on STaR

To evaluate the proposed parameterization in a setting that demands relational composition, we use STaR (Khalid & Schockaert, 2025), a controlled reasoning benchmark in which the edge relations determine how relational information is composed along a path. STaR tests systematic generalization over synthetic relation graphs in two domains, i.e., RCC-8 and Allen's Interval Algebra (IA). Each instance provides an edge-labelled graph and a query node pair, and requires inferring the relation between them. Solving this task entails composing relations along each path and integrating the partial, potentially disjunctive evidence from multiple paths. STaR is therefore well-suited to testing edge-conditioned transport: distinct relations call for distinct transformations of propagated information, while multiple paths require the resulting relational states to be combined.

All evaluated models are trained on graphs with path lengths $k \in \{2, 3, 4\}$ and numbers of paths $b \in \{1, 2, 3\}$, then evaluated on longer reasoning chains to measure generalization beyond the depths seen during training. Within each backbone family, the baseline and Top-A variant use identical input

Table 1: *STaR Relation-Classification Accuracy*. We evaluate over 3 seeds, reporting $\pm$ std $\times 100$. Bold indicates the best-performing models.

Architecture	Variant	RCC-8	Interval Algebra
GATv2	–	39.79 ± 23.83	18.05 ± 17.94
	Top-A	$\mathbf{51.79 \pm 7.14}$	$\mathbf{52.44 \pm 1.22}$
GT	–	23.34 ± 17.68	18.03 ± 15.69
	Top-A	$\mathbf{60.24 \pm 10.21}$	$\mathbf{57.17 \pm 2.54}$

Table 2: *Link Prediction on MovieLens-100K*. We evaluate over 10 seeds, reporting mean $\pm$ std $\times 100$.

Model	Variant	AUROC	AUPR
GATv2	–	88.70 ± 0.83	86.48 ± 1.07
	Top-A	$\mathbf{90.46 \pm 0.42}$	$\mathbf{88.71 \pm 0.51}$
GT	–	81.26 ± 3.67	76.24 ± 5.21
	Top-A	$\mathbf{90.20 \pm 0.85}$	$\mathbf{88.38 \pm 0.99}$

representations, backbone architectures, prediction heads, and optimization procedures. Appendix C.1 provides further details.

Results. Table 1 show that Top-A improves mean accuracy for both attention backbones across both relational domains, supporting the value of the proposed head-space parameterization for relational reasoning. Standard relation-aware attention can condition its scalar coefficients on the edge label; Top-A additionally lets each labelled interaction induce off-diagonal routing across head subspaces before aggregation. This mechanism can support relation-specific transformations and the integration of evidence propagated along multiple paths. Figure 3 plots accuracy against reasoning depth k, showing that Top-A's gains extend across multiple unseen depths. On both RCC-8 and IA, the Top-A variants outperform their corresponding attention baselines within the training-depth range $k \in \{2, 3, 4\}$ and across a substantial range of unseen depths, although the margin narrows on the most challenging configurations. Since b remains within its training range, this analysis isolates increasing reasoning depth from growth in the number of paths.

5.2 Relation-Conditioned Routing on Heterogeneous Graphs

After establishing the benefits of edge-conditioned cross-head transport in STaR's controlled relational setting, we evaluate the same parameterization on a real-world heterogeneous graph task. We use MovieLens-100K (Harper & Konstan, 2015), where users and movies forms semantically distinct node types connected by directed relations. This setting complents STaR by testing whether communication across latent head subspaces helps integrate information with different node and relation semantics. We formulate MovieLens-100K as a binary link-prediction task and compare the same two attention backbones with their corresponding Top-A variants. Within each backbone family, all models use identical heterogeneous input encoders, backbone, decoder, and per-run training protocol. The Top-A variants additionally tune the transport scale λ, with the search space and selected configurations reported in Appendix C.2.

Results. As shown in Table 2, Top-A improves mean AUROC and AUPR for both attention backbones across ten evaluation seeds. For GATv2, cross-head transport yields average gains of approximately 2.0 points across the two metrics; for GT, the gains increase to 10.5 points on average. GT+Top-A closes the performance gap with GATv2 and performs comparably to GATv2+Top-A, while reducing the variability of the original GT backbone. The transport analysis in Appendix C.2 (Figure 4) further reveals distinct effective head-space transport for the *rates* and *rated-by* directions. This directional asymmetry indicates that the two relations induce effective head-space transformations. Detailed transport and optimization analyses are reported in Appendix C.2.

5.3 CLRS Benchmark

Having evaluated cross-head transport in controlled relational and real-world heterogeneous settings, we next investigate which computational procedures benefit from Top-A's additional routing capacity. We use the CLRS Algorithmic Reasoning Benchmark (Veličković et al., 2022), which comprises 30 classical algorithms spanning sorting, searching, dynamic programming, graph algorithms, string processing, and computational geometry. This diversity makes CLRS well suited to probing architectural inductive biases, as different algorithms require substantially different forms of information transport. Rather than expecting cross-head transport to provide uniform gains across all tasks, we use CLRS to identify when Top-A's additional flexibility aligns with the computation required by the target algorithm. Following the standard benchmark protocol, models are trained on instances of size $n = 16$ and evaluated out-of-distribution at $n = 64$, testing whether the learned routing

supports reusable computational procedures beyond the training scale. Within each backbone family, all models share the same input encoders, processor architecture, decoder, and training protocol. Additional details are provided in Appendix C.3.

Results. As shown in Table 3, the gains from cross-head transport are strongly task dependent; Appendix C.3, Table 7, reports the full results. The largest gains occur on sorting and searching, where GATv2+Top-A improves over GATv2 by more than 29 and 12 points, respectively. This trend is also evident with the GT, for which Top-A improves searching algorithms by more than 24 points. By contrast, families such as greedy algorithms and dynamic programming show little or no improvement. Averaged over all 30 tasks, Top-A improves GATv2 by 6 points and GT by 8.5 points. This heterogeneous pattern supports our hypothesis that cross-head transport provides a task-dependent inductive bias rather than a uniform advantage across algorithms. The pronounced gain on sorting and searching suggest that these procedures are particularly well matched to flexible information exchange across latent head subspaces during the execution of the learned algorithm.

Table 3: CLRS-30 test scores (official CLRS score; %, ↑) after separate hyperparameter optimization, grouped by the benchmark's eight algorithm families. Entries are mean ± sample SD across 20 seeds, averaging tasks equally within each seed. Overall averages all 30 tasks equally. Δ is the improvement from adding Top-A in percentage points, computed before rounding. Bold marks the higher displayed mean within each architecture.

Family	Tasks	GATv2	GATv2 + Top-A	Δ	GT	GT + Top-A	Δ
Sorting	4	25.18 ± 7.62	$\mathbf{54.42} \pm 6.77$	$+29.24$	23.81 ± 9.40	$\mathbf{54.17} \pm 8.43$	$+30.36$
Searching	3	42.31 ± 4.46	$\mathbf{54.72} \pm 4.01$	$+12.41$	31.92 ± 1.37	$\mathbf{56.24} \pm 2.62$	$+24.32$
Divide and conquer	1	58.10 ± 4.78	$\mathbf{63.61} \pm 3.34$	$+5.51$	57.19 ± 4.66	$\mathbf{62.91} \pm 3.73$	$+5.72$
Greedy	2	$\mathbf{74.22} \pm 2.56$	73.92 ± 2.80	-0.30	$\mathbf{74.98} \pm 2.20$	73.94 ± 2.83	-1.03
Dynamic programming	3	73.54 ± 4.00	$\mathbf{73.71} \pm 5.92$	$+0.17$	$\mathbf{76.34} \pm 4.72$	70.02 ± 7.28	-6.32
Graphs	12	60.68 ± 1.54	$\mathbf{61.86} \pm 2.04$	$+1.18$	54.51 ± 1.62	$\mathbf{60.51} \pm 1.84$	$+5.99$
Strings	2	1.39 ± 0.85	$\mathbf{1.54} \pm 0.78$	$+0.15$	1.40 ± 0.79	$\mathbf{1.92} \pm 0.83$	$+0.53$
Geometry	3	70.26 ± 1.94	$\mathbf{72.63} \pm 2.40$	$+2.36$	70.04 ± 3.44	$\mathbf{71.59} \pm 4.17$	$+1.55$
Overall	30	53.22 ± 1.26	$\mathbf{59.26} \pm 1.48$	$+6.04$	49.81 ± 1.73	$\mathbf{58.36} \pm 1.20$	$+8.56$

6 Conclusion

We revisited matrix-valued transport as a principled generalization of multi-head attention beyond its diagonal structure in head space. By allowing off-diagonal maps, edges can coordinate information flow across heads before aggregation. This perspective led to Top-A, a direct realization of the resulting transport primitive that retains standard attention as a special case. Across relational, heterogeneous, and algorithmic reasoning tasks, this richer routing mechanism consistently provides a useful inductive bias. More broadly, our results establish a direct connection between sheaf transport and attention through a simple yet powerful principle: edges should determine not only how much information is transmitted, but also how it is routed across heads.

AI use statement

In this work, we used generative AI tools solely to aid and polish the writing, including improving clarity, grammar, phrasing, and overall presentation. We did not use generative AI tools to generate research ideas, conduct experiments, analyze results, or make scientific claims. All AI-assisted text was carefully reviewed, revised, and verified by the authors to ensure accuracy and consistency with the underlying research. We take full responsibility for the final content of this work, including any text produced with the aid of generative AI.

Ethics statement

This work does not involve human subjects, personally identifiable information, or the collection of sensitive data. The datasets used in our experiments are publicly available research benchmarks and are used in accordance with their respective licenses and intended research purposes. We are not aware of any significant ethical concerns arising from the proposed methodology or its experimental evaluation. As with other machine learning methods, downstream applications should nevertheless

consider potential issues related to bias, fairness, privacy, and misuse. We take responsibility for ensuring that the research is conducted and reported in accordance with established standards of research integrity.

Reproducibility statement

We have taken several steps to facilitate reproducibility. The main paper provides a complete description of the proposed method and experimental setting, while additional architectural details, implementation choices, hyperparameters, dataset information, and experimental configurations are reported in the appendix. We also provide the details required to reproduce the reported baselines, ablations, and evaluation procedures. The source code and configuration files used to produce the experimental results will be publicly released upon acceptance.

References

Cristian Bodnar, Francesco Di Giovanni, Benjamin Paul Chamberlain, Pietro Liò, and Michael M. Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in GNNs. In *Advances in Neural Information Processing Systems*, volume 35, pp. 18527–18541, 2022. doi: 10.52202/068431-1346. URL https://openreview.net/forum?id=vbPsD-BhOZ.

Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=F72ximsx7C1.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021. URL https://arxiv.org/abs/2104.13478.

Chen Cai and Yusu Wang. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*, 2020. URL https://arxiv.org/abs/2006.13318.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 3rd edition, 2009. ISBN 9780262033848.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, volume 29, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/04df4d434d481c5bb723be1b6df1ee65-Abstract.html.

Harm Derksen and Jerzy Weyman. *An Introduction to Quiver Representations*, volume 184 of *Graduate Studies in Mathematics*. American Mathematical Society, 2017. doi: 10.1090/gsm/184.

Cameron Diao and Ricky Loynd. Relational attention: Generalizing transformers for graph-structured tasks. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=cFuMmbWiN6.

Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. In *AAAI Workshop on Deep Learning on Graphs: Methods and Applications*, 2021. URL https://arxiv.org/abs/2012.09699.

Stefano Fiorini, Edoardo Coppola, and Pietro Liò. Benchmarking sheaf neural networks for inductive tasks. *arXiv preprint arXiv:2608.02558*, 2026. URL https://arxiv.org/abs/2608.02558.

Peter Gabriel. Unzerlegbare darstellungen i. *Manuscripta Mathematica*, 6(1):71–103, 1972. doi: 10.1007/BF01298413.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1263–1272. PMLR, 2017. URL https://proceedings.mlr.press/v70/gilmer17a.html.

Mustafa Hajij, Lennart Bastian, Sarah Osentoski, Hardik Kabaria, John Davenport, Dawood, Balaji Cherukuri, Joseph Kocheemoolayil, Nastaran Shahmansouri, Adrian Lew, Theodore Papamarkou, and Tolga Birdal. Copresheaf topological neural networks: A generalized deep learning framework. In *Advances in Neural Information Processing Systems*, volume 38, pp. 149759–149803, 2025. doi: 10.52202/085713-5009. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/dc62cd4ec77f3bbec8e6245d0bd91d08-Abstract-Conference.html.

Jakob Hansen and Thomas Gebhart. Sheaf neural networks. *arXiv preprint arXiv:2012.06333*, 2020. URL https://arxiv.org/abs/2012.06333.

Jakob Hansen and Robert Ghrist. Toward a spectral theory of cellular sheaves. *Journal of Applied and Computational Topology*, 3:315–358, 2019. doi: 10.1007/s41468-019-00038-7.

F. Maxwell Harper and Joseph A. Konstan. The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19:1–19:19, 2015. doi: 10.1145/2827872.

Ferran Hernandez Caralt, Mar Gonzàlez i Català, Adrián Bazaga, and Pietro Liò. On the necessity of learnable sheaf laplacians. *arXiv preprint arXiv:2603.05395*, 2026. URL https://arxiv.org/abs/2603.05395.

Irtaza Khalid and Steven Schockaert. Systematic relational reasoning with epistemic graph neural networks. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=qNp86ByQlN.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Sitao Luan, Chenqing Hua, Qincheng Lu, Liheng Ma, Lirong Wu, Xinyu Wang, Minkai Xu, Xiao-Wen Chang, Doina Precup, Rex Ying, Stan Z. Li, Jian Tang, Guy Wolf, and Stefanie Jegelka. The heterophilic graph learning handbook: Benchmarks, models, theoretical analysis, applications and challenges. *arXiv preprint arXiv:2407.09618*, 2024. URL https://arxiv.org/abs/2407.09618.

Alessio Micheli. Neural network for graphs: A contextual constructive approach. *IEEE Transactions on Neural Networks*, 20(3):498–511, 2009. doi: 10.1109/TNN.2008.2010350.

Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1ldO2EFPr.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of GNNs under heterophily: Are we really making progress? In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tJbbQfw-5wv.

Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. In *Advances in Neural Information Processing Systems*, volume 35, pp. 14501–14515, 2022. doi: 10.52202/068431-1054.

André Ribeiro, Ana Luiza Tenório, Juan Belieni, Amauri H. Souza, and Diego Mesquita. Cooperative sheaf neural networks. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=AHpexliCTM.

T. Konstantin Rusch, Michael M. Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv:2303.10993*, 2023. URL https://arxiv.org/abs/2303.10993.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009. doi: 10.1109/TNN.2008.2005605.

Noam Shazeer, Zhenzhong Lan, Youlong Cheng, Nan Ding, and Le Hou. Talking-heads attention. *arXiv preprint arXiv:2003.02436*, 2020.

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjing Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pp. 1548–1554. International Joint Conferences on Artificial Intelligence Organization, 2021. doi: 10.24963/ijcai.2021/214. URL https://doi.org/10.24963/ijcai.2021/214.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Petar Veličković, Adrià Puigdomènech Badia, David Budden, Razvan Pascanu, Andrea Banino, Misha Dashevskiy, Raia Hadsell, and Charles Blundell. The CLRS algorithmic reasoning benchmark. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 22084–22102. PMLR, 2022. URL https://proceedings.mlr.press/v162/velickovic22a.html.

Da Xiao, Qingye Meng, Shengping Li, and Xingyuan Yuan. Improving transformers with dynamically composable multi-head attention. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Let the Heads Talk: Beyond Diagonal Graph Attention

Supplementary Material

APPENDIX

A Extended Related Work **14**

B Theoretical and Architectural Details **14**

B.1 From Cellular Sheaves to Symmetric Attention 14

B.2 Proof of Proposition 3.1 15

B.3 Shared Post-Aggregation Equivalence 16

B.3.1 Proof of Theorem 4.1 16

B.3.2 A two-edge example 17

B.4 Backbone-Specific Routing Instantiations 17

B.5 Initialization and Computational Cost 18

C Additional Experimental Details, Results, and Analysis **18**

C.1 STaR Benchmark: Dataset, Task, and Model Configuration 18

C.2 MovieLens-100K: Experimental Setup and Analysis 19

C.3 CLRS-30 21

C.4 Heterophilic Node Classification 22

A Extended Related Work

This section expands on the related work introduced in Section 2, providing a more detailed discussion of the literature most closely connected to our contribution.

Relation to copresheaf attention. Hajij et al. (2025) formulate neural message passing through copresheaves, associating local representation spaces with nodes and linear maps with directed interactions. This directed-map perspective is closely related to the level of abstraction adopted in our quiver formulation. In their copresheaf attention construction, however, transport acts within each attention head: for an interaction $j \to i$, a learned map $\rho^{(h)}_{i\leftarrow j} \in \mathbb{R}^{C\times C}$ transforms the value carried by head h before attention-weighted aggregation, with the maps evaluated independently across heads. On the full representation $\mathbb{R}^{HC} \simeq \mathbb{R}^H \otimes \mathbb{R}^C$, this gives a block-diagonal operator of the form $\rho^{(1)}_{i\leftarrow j} \oplus \cdots \oplus \rho^{(H)}_{i\leftarrow j}$. Top-A instead identifies the head space itself as the transport space and associates each directed interaction with an $H \times H$ map. Its action on the full representation is $T_{i\leftarrow j} \otimes I_C$: information is mixed across heads while the within-head coordinates are carried unchanged. The two constructions therefore place matrix-valued transport on different factors of the head–channel representation: copresheaf attention acts within heads, whereas Top-A acts between them. This comparison concerns the specific attention construction of Hajij et al. (2025); the general copresheaf formalism is substantially broader.

Relation to head-composition methods. To position Top-A with respect to existing forms of cross-head communication, we distinguish our matrix-valued transport view from methods that compose attention heads directly. Importantly, we note that our goal is not to introduce an alternative head-composition mechanism, but to study the computational role of matrix-valued transport through its connection with multi-head attention.

Talking-Heads Attention (Shazeer et al., 2020) introduces linear projections across the head dimension of the attention logits before softmax and of the normalized attention weights after softmax, while DCMHA (Xiao et al., 2024) extends this principle through input-dependent composition of attention scores and weights. Like Top-A, these methods enable communication across heads, but they do so at a different stage of the attention computation. Specifically, Talking-Heads Attention and DCMHA operate on the scalar quantities that determine how each value head weights its inputs, without directly transferring value representations between heads. By contrast, Top-A acts on the value-head representation associated with each directed interaction, $\left[T_{i\leftarrow j}V_j\right]^{(r)} = \sum_{s=1}^{H} T^{rs}_{i\leftarrow j} V^{(s)}_j$, allowing the value carried by source head s to contribute directly to receiver head r before neighborhood aggregation. The parameterization $T_{i\leftarrow j} = D_{i\leftarrow j}(I_H + \lambda\Omega_{i\leftarrow j})$, with $\mathrm{diag}(\Omega_{i\leftarrow j}) = 0$, retains the original same-head attention paths while adding interaction-dependent off-diagonal routes between value heads, and recovers standard multi-head attention exactly when $\Omega_{i\leftarrow j} = 0$. From this perspective, Talking-Heads Attention and DCMHA compose attention patterns across heads, whereas Top-A provides a direct realization of edge-conditioned transport over the value-head space.

B Theoretical and Architectural Details

B.1 From Cellular Sheaves to Symmetric Attention

The main text studies the transport map $T_{i\leftarrow j}$ carried by each directed interaction $j \to i$, abstracting away from the restriction-map factorization and diffusion operator of a cellular sheaf. We now return to the cellular-sheaf setting and show that the diagonal edge transport induced by a symmetric form of multi-head attention admits an exact realization as sheaf transport. This establishes a direct connection between diagonal head-space attention and the local transport structure underlying Neural Sheaf Diffusion (Hansen & Gebhart, 2020; Bodnar et al., 2022).

Definition B.1 (Symmetric multi-head attention)**.** A multi-head attention mechanism with H heads is *symmetric* if, for every undirected edge $e = \{i, j\}$ and every head $h \in \{1, \ldots, H\}$,

$$\alpha^{(h)}_{i\leftarrow j} = \alpha^{(h)}_{j\leftarrow i} \geq 0.$$

The corresponding head-space operator is

$$D_{i\leftarrow j} = \mathrm{Diag}\left(\alpha_{i\leftarrow j}^{(1)}, \ldots, \alpha_{i\leftarrow j}^{(H)}\right) \in \mathbb{R}^{H\times H}.$$

By symmetry, $D_{i\leftarrow j} = D_{j\leftarrow i}$.

The following proposition shows that every such diagonal edge operator can be realized by cellular-sheaf restriction maps.

Proposition B.2 (Symmetric attention as sheaf transport)**.** *Consider node and edge stalks of dimension H, $\mathcal{F}(i) = \mathcal{F}(j) = \mathcal{F}(e) = \mathbb{R}^H$. Given a symmetric multi-head attention mechanism, there exist restriction maps such that the induced sheaf transport*

$$T_{i\leftarrow j} = F_{i\to e}^{\top} F_{j\to e}$$

satisfies

$$T_{i\leftarrow j} = D_{i\leftarrow j}$$

for every edge $e = \{i, j\}$.

Proof. Since $D_{i\leftarrow j}$ is diagonal with nonnegative entries, its principal square root is well defined. For each edge $e = \{i, j\}$, choose

$$F_{i\to e} = F_{j\to e} = D_{i\leftarrow j}^{1/2}.$$

Then

$$F_{i\to e}^{\top} F_{j\to e} = \left(D_{i\leftarrow j}^{1/2}\right)^{\top} D_{i\leftarrow j}^{1/2} = D_{i\leftarrow j},$$

where the final equality follows because $D_{i\leftarrow j}^{1/2}$ is diagonal. The same construction gives $T_{j\leftarrow i} = D_{j\leftarrow i} = D_{i\leftarrow j}$ in the reverse direction. □

Thus, symmetric multi-head attention induces a diagonal head-space transport that admits an exact realization within the cellular-sheaf framework.

B.2 Proof of Proposition 3.1

Proposition 3.1 (Multi-head attention as diagonal transport)**.** *Before any shared post-aggregation readout, normalized multi-head attention is diagonal transport on its native head space. In particular, for every directed interaction $j \to i$ and receiver head $r \in \{1, \ldots, H\}$:*

$$(D_{i\leftarrow j} V_j)^{(r)} = \alpha_{i\leftarrow j}^{(r)} V_j^{(r)} \tag{11}$$

where $(\cdot)^{(r)}$ denotes the r-th head component.

Proof. Once the attention coefficients for the interaction $j \to i$ have been computed, they define the diagonal matrix

$$D_{i\leftarrow j} = \mathrm{Diag}\left(\alpha_{i\leftarrow j}^{(1)}, \ldots, \alpha_{i\leftarrow j}^{(H)}\right).$$

The sender values are arranged as

$$V_j = \begin{bmatrix} (V_j^{(1)})^{\top} \\ \vdots \\ (V_j^{(H)})^{\top} \end{bmatrix},$$

so multiplying by $D_{i\leftarrow j}$ simply scales each head by its own attention coefficient:

$$D_{i\leftarrow j} V_j = \begin{bmatrix} \alpha_{i\leftarrow j}^{(1)} (V_j^{(1)})^{\top} \\ \vdots \\ \alpha_{i\leftarrow j}^{(H)} (V_j^{(H)})^{\top} \end{bmatrix}.$$

Therefore, for any receiver head r,

$$(D_{i\leftarrow j} V_j)^{(r)} = \alpha_{i\leftarrow j}^{(r)} V_j^{(r)}.$$

Because the off-diagonal entries of $D_{i\leftarrow j}$ are zero, a value from source head $s \neq r$ cannot contribute to receiver head r. The edge map induced by standard multi-head attention is therefore diagonal in head space. □

B.3 Shared Post-Aggregation Equivalence

This section proves Theorem 4.1 and gives a two-edge example illustrating why edge-specific routing before aggregation cannot, in general, be replaced by a single linear map applied afterwards.

B.3.1 Proof of Theorem 4.1

Recall that $\overline{D}_{i\leftarrow j} = D_{i\leftarrow j} \otimes I_C$ and $\overline{T}_{i\leftarrow j} = T_{i\leftarrow j} \otimes I_C$.

Theorem 4.1 (Shared post-aggregation equivalence)**.** *Fix a receiver i and the edge operators $\{\overline{D}_{i\leftarrow j}, \overline{T}_{i\leftarrow j}\}_{j\in\mathcal{N}_{\mathrm{att}}(i)}$. A single post-aggregation map reproduces Top-A's routed aggregation for every assignment of sender values if and only if some linear map $A \in \mathbb{R}^{HC\times HC}$ satisfies*

$$\sum_{j\in\mathcal{N}_{\mathrm{att}}(i)} \overline{T}_{i\leftarrow j} v_j = A \sum_{j\in\mathcal{N}_{\mathrm{att}}(i)} \overline{D}_{i\leftarrow j} v_j, \tag{16}$$

if and only if the same map A reproduces the transport on every incoming edge:

$$\overline{T}_{i\leftarrow j} = A\overline{D}_{i\leftarrow j} \qquad \text{for every } j \in \mathcal{N}_{\mathrm{att}}(i). \tag{17}$$

Proof. Fix a receiver i, the corresponding edge operators, and write $\mathcal{N}_i = \mathcal{N}_{\mathrm{att}}(i)$.

Suppose first that a linear map A satisfies

$$\sum_{j\in\mathcal{N}_i} \overline{T}_{i\leftarrow j} v_j = A \sum_{j\in\mathcal{N}_i} \overline{D}_{i\leftarrow j} v_j$$

for every choice of the sender values. For any $j \in \mathcal{N}_i$, set all sender values except v_j to zero. Since the edge operators are fixed, this gives

$$\overline{T}_{i\leftarrow j} v_j = A\overline{D}_{i\leftarrow j} v_j$$

for every v_j, which implies

$$\overline{T}_{i\leftarrow j} = A\overline{D}_{i\leftarrow j}.$$

Since j was arbitrary, this holds for every incoming edge.

Conversely, if

$$\overline{T}_{i\leftarrow j} = A\overline{D}_{i\leftarrow j} \qquad \text{for every } j \in \mathcal{N}_i,$$

then, for any collection of sender values,

$$\sum_{j\in\mathcal{N}_i} \overline{T}_{i\leftarrow j} v_j = \sum_{j\in\mathcal{N}_i} A\overline{D}_{i\leftarrow j} v_j = A \sum_{j\in\mathcal{N}_i} \overline{D}_{i\leftarrow j} v_j.$$

This proves both directions. □

The theorem shows that, conditional on the edge operators induced for a receiver, edge-specific routing can be moved after aggregation only if the same linear map A reproduces the routed transformation of every incoming edge. We allow A to be an arbitrary linear map for the receiver i, making this comparison more permissive than the usual output projection of the attention backbone.

For the Top-A transport in Equation (13), if $D_{i\leftarrow j}$ is invertible, the condition becomes

$$A = \left[D_{i\leftarrow j}\left(I_H + \lambda\Omega_{i\leftarrow j}\right) D_{i\leftarrow j}^{-1}\right] \otimes I_C$$

for every incoming sender j. A single post-aggregation map therefore exists only if all incoming edges induce the same effective transformation. Since $\Omega_{i\leftarrow j}$ may vary with the directed interaction, Top-A does not enforce this condition, and it need not hold in general.

Remark. Theorem 4.1 itself does not require $D_{i\leftarrow j}$ to be invertible. Invertibility is used only to express the condition explicitly as Equation (18). For ordinary unmasked softmax attention with finite logits, all attention coefficients are positive, so $D_{i\leftarrow j}$ is invertible for every attended interaction.

B.3.2 A TWO-EDGE EXAMPLE

A simple example makes the distinction concrete. Consider a receiver i with two incoming senders j_1 and j_2, two attention heads ($H = 2$), and one value coordinate per head ($C = 1$). Suppose both heads assign equal attention to the two incoming edges, so that

$$D_{i\leftarrow j_1} = D_{i\leftarrow j_2} = \frac{1}{2} I_2.$$

Set $\lambda = 1$, choose some $\rho \in (0, 1)$, and let the two edges use different routing matrices,

$$\Omega_{i\leftarrow j_1} = \begin{bmatrix} 0 & \rho \\ 0 & 0 \end{bmatrix}, \qquad \Omega_{i\leftarrow j_2} = \begin{bmatrix} 0 & 0 \\ \rho & 0 \end{bmatrix}.$$

The corresponding Top-A transports are

$$T_{i\leftarrow j_1} = \frac{1}{2} \begin{bmatrix} 1 & \rho \\ 0 & 1 \end{bmatrix}, \qquad T_{i\leftarrow j_2} = \frac{1}{2} \begin{bmatrix} 1 & 0 \\ \rho & 1 \end{bmatrix}.$$

If a single linear map A applied after aggregation could reproduce both edge-specific transports, Theorem 4.1 would require

$$A = T_{i\leftarrow j_1} D_{i\leftarrow j_1}^{-1} = I_2 + \Omega_{i\leftarrow j_1}$$

and also

$$A = T_{i\leftarrow j_2} D_{i\leftarrow j_2}^{-1} = I_2 + \Omega_{i\leftarrow j_2}.$$

These two requirements are incompatible because $\Omega_{i\leftarrow j_1} \neq \Omega_{i\leftarrow j_2}$.

The same point can be seen directly. Let $v = (1, 0)^\top$ and compare the two collections of incoming values

$$(v_{j_1}, v_{j_2}) = (v, 0), \qquad (v_{j_1}, v_{j_2}) = (0, v).$$

Standard attention maps both to the same aggregate, $\frac{1}{2}v$. Any linear map applied after aggregation therefore receives the same input in the two cases and must return the same output.

With Top-A, the edge-specific maps are applied before the messages are summed:

$$T_{i\leftarrow j_1} v = \frac{1}{2} \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad T_{i\leftarrow j_2} v = \frac{1}{2} \begin{bmatrix} 1 \\ \rho \end{bmatrix}.$$

Thus the two collections, which standard attention collapses to the same aggregate, produce different outputs under edge-specific pre-aggregation routing.

B.4 BACKBONE-SPECIFIC ROUTING INSTANTIATIONS

The routing construction in Section 4.1 does not depend on a particular attention scoring rule. Different backbones provide different representations of the same interaction, which we use to construct the edge-dependent routing matrix Ω_e.

GATv2. For GATv2 (Brody et al., 2022), let z_i denote the representation entering the attention layer. We define the directed pre-attention context

$$u_{i\leftarrow j} = W_{\text{recv}} z_i + W_{\text{send}} z_j + W_E \mathbf{e}_{i\leftarrow j} + W_G \mathbf{g},$$

where $\mathbf{e}_{i\leftarrow j}$ and $\mathbf{g}$ denote edge- and graph-level features, respectively. We then set

$$\xi_{i\leftarrow j} = \text{LeakyReLU}(u_{i\leftarrow j}),$$

and use $\xi_{i\leftarrow j}$ as the interaction representation in Equation (15). The resulting descriptor is mapped jointly to the full $H \times H$ routing matrix.

Graph Transformer. For the graph Transformer (Shi et al., 2021), we first form the edge-conditioned sender key

$$K_{i\leftarrow j} = K_j + E_{i\leftarrow j} + G,$$

where K_j is the sender key and $E_{i\leftarrow j}$ and G denote edge- and graph-level embeddings in the same key space. Combining this with the receiver query Q_i, we define

$$\xi_{i\leftarrow j} = \text{LeakyReLU}\left(\text{vec}(Q_i + K_{i\leftarrow j})\right),$$

which is supplied to the routing generator in Equation (15).

B.5 Initialization and Computational Cost

Initialization. For the routing generator, we use

$$g_\theta(\xi_e) = \mathrm{reshape}_{H\times H}\left(W_\Omega \xi_e + b_\Omega\right),$$

with $g_\theta : \mathbb{R}^p \to \mathbb{R}^{H\times H}$. We initialize $W_\Omega = 0$ and $b_\Omega = 0$, so that $g_\theta(\xi_e) = 0$ and therefore $\Omega_e = 0$ for every interaction. Equation (13) then reduces to

$$m_e^{Top-A} = D_e V_j,$$

which is exactly the message produced by the corresponding attention baseline. With matched backbone parameters, Top-A therefore implements the same function as the baseline at initialization.

This zero initialization does not prevent the routing generator from learning. For $\lambda > 0$, $\tanh'(0) = 1$, so the parameters W_Ω and b_Ω can receive first-order gradients at initialization.

Computational cost. The additional cost comes from generating and applying a dense $H \times H$ routing matrix in head space. For the routing generator, an interaction representation $\xi_e \in \mathbb{R}^p$ requires $O(pH^2)$ operations to produce Ω_e, while applying the resulting matrix to $V_j \in \mathbb{R}^{H\times C}$ costs $O(H^2C)$ per interaction. Across the graph, the additional computation is therefore $O\big(|\mathcal{E}|H^2(p+C)\big)$. If the routing matrices are materialized explicitly, they require $O(|\mathcal{E}|H^2)$ additional memory.

C Additional Experimental Details, Results, and Analysis

This section complements the experimental evaluation in section 5 with additional dataset and implementation details, hyperparameter configurations, extended results, and further analyses for each experimental setting. All experiments were run on NVIDIA A100 SXM4 GPUs with 80 GB of memory. The code will be released upon acceptance.

C.1 STaR Benchmark: Dataset, Task, and Model Configuration

STaR (Khalid & Schockaert, 2025) is a relational reasoning benchmark that provides a controlled setting in which edge-dependent transformations are themselves part of the computation. Each instance consists of an edge-labelled graph together with a query pair of nodes, and the task is to infer the relation holding between them. STaR contains two relational domains, each comprising 57,600 training instances and 153,600 test instances: RCC-8, with eight qualitative spatial relations, and Allen's Interval Algebra (IA), with thirteen temporal relations. In both domains, the relation carried by an edge determines not only whether information is relevant, but also how the current relational state should be transformed before propagation. Unlike conventional single-path reasoning benchmarks, STaR requires both composing relations along individual paths and integrating the partial, potentially disjunctive information provided by multiple paths. Problem difficulty is controlled by the path length k and the number of paths b connecting the queried nodes. Models are trained on graphs with $k \in \{2, 3, 4\}$ and $b \in \{1, 2, 3\}$, and evaluated at $k \in \{2, 3, \ldots, 15\}$ while keeping $b \in \{1, 2, 3\}$. This allows us to evaluate systematic generalization to reasoning depths beyond those observed during training while keeping the number of paths within the training range.

Relation labels are supplied as edge information to both the attention backbone and the Top-A routing mechanism, so that the comparison isolates the additional capacity provided by cross-head transport rather than access to relational information. Within each backbone family, the baseline and Top-A variant share the same input representation, backbone architecture, prediction head, optimization procedure, and tuning budget. As throughout our experiments, the routing branch is zero-initialized, so that each Top-A model starts from the same function as its corresponding attention baseline. Performance is measured by query-relation classification accuracy across the different (k, b) configurations. We otherwise follow the experimental setting of Khalid & Schockaert (2025).

Hyperparameters and training protocol. All STaR experiments use a fixed hyperparameter configuration. Both GATv2 and the GT use a hidden dimension of 64, four attention heads, and 15 recurrent message-passing rounds with shared processor weights. The Top-A variant has a hidden dimension of 64 and a transport scale of 1. Training uses AdamW with learning rate 10^{-3}, weight decay 10^{-5}, and batch size 128. Attention dropout is disabled, while residual dropout is set to 0.1.

Models train for at most 40 epochs, with early stopping after eight epochs without improvement in validation accuracy; the checkpoint with the highest validation accuracy is retained.

For both RCC-8 and Interval Algebra, the released training mixture covers $b \in \{1, 2, 3\}$ and $k \in \{2, 3, 4\}$. An 80/20 training–validation split is stratified jointly by target relation and (b, k) configuration. Each experiment uses three training seeds. Within each architecture, the baseline and Top-A variants share the training configuration, initial shared parameters, minibatch order, and dropout randomness. The baseline freezes the off-diagonal controller at zero, whereas Top-A trains it from the same zero initialization.

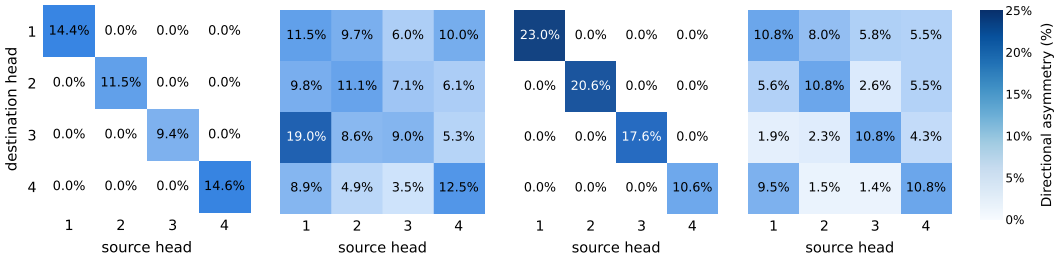
C.2 MovieLens-100K: Experimental Setup and Analysis

Dataset, task, and model configuration. MovieLens-100K (Harper & Konstan, 2015) provides a real-world heterogeneous setting in which information is exchanged between two semantically distinct node types. We represent the dataset as a bipartite graph containing 943 users, described by 24-dimensional demographic features, and 1,682 movies, described by 18-dimensional genre features. Each observed rating induces two directed interactions, *rates* ($u \to m$) and *rated-by* ($m \to u$), which exchange the roles of sender and receiver while connecting the same user–movie pair. We formulate the task as binary user–movie link prediction. We use the official 80k/20k rating split for training and test positive edges, respectively, and hold out 10% of the training ratings as a fixed validation set. Validation edges are removed from the message-passing graph. Negative examples are sampled uniformly from unobserved user–movie pairs using a 1:1 positive-to-negative ratio, with validation and test negatives fixed across runs. Type-specific input encoders map user and movie attributes to a shared 64-dimensional representation, to which we add a learned node-type embedding. Relation type is represented by a one-hot edge feature and is available to both the attention mechanism and the Top-A routing generator. We evaluate GATv2 and a sparse GT, together with their corresponding Top-A variants. All architectures use $H = 4$ attention heads, two message-passing layers, and a dot-product link-prediction decoder. For each backbone family, the baseline and Top-A variant share the same input encoders, backbone architecture, decoder, and training protocol. Hyperparameters are selected on validation performance using tuning seeds that are disjoint from the final evaluation seeds. The learning-rate search is shared across all models, while the Top-A variants additionally select the transport scale λ, as detailed below. Final results are evaluated over ten seeds that are disjoint from those used for hyperparameter selection, and test AUROC and AUPR are measured at the best-validation epoch. For each evaluation seed, the baseline and Top-A variant use the same initialization for all shared parameters, while the routing generator is zero-initialized. Consequently, the paired models implement the same function before optimization.

Hyperparameter selection and implementation details. We select the hyperparameters independently for each model using mean validation AUROC over three tuning seeds (0–2), which are disjoint from the ten evaluation seeds (42–51). The learning-rate grid is shared across all four models, while the route scale λ applies only to the Top-A variants. All remaining architectural and optimization choices are fixed within each backbone family and are reported in Table 4. The configurations selected by validation performance are reported in Table 5.

Table 4: MovieLens-100K hyperparameter search space, shared by GATv2, GT, and their respective Top-A variants.

Hyperparameter	Search space
Learning rate	$\{5 \times 10^{-4}, 10^{-3}, 2 \times 10^{-3}\}$
Route scale λ (Top-A only)	$\{0.5, 1\}$
Hidden size	64 (4 heads $\times$ 16 channels)
Attention heads	4
Layers	2
Dropout probability (input / feature / attention)	0.2
Weight decay	10^{-5}
Optimiser	Adam
Max. epochs / early-stopping patience	1000 / 100
Negative sampling	1:1 uniform, resampled every epoch



(a) GATv2 (b) GATv2 + Top-A (c) GT (d) GT + Top-A

Figure 4: *Relative Directional Asymmetry of the Mean Head-to-Head Transport.* Entrywise normalized asymmetry N_{rs} from Equation (21), multiplied by 100, between the mean effective transport for *rates* ($u \to m$) and *rated-by* ($m \to u$). Rows index receiver heads and columns source heads. Standard GATv2 and GT have zero off-diagonal entries because vanilla attention does not permit cross-head transport. With Top-A, nonzero off-diagonal asymmetry indicates that the effective cross-head contributions differ between the two propagation directions.

Table 5: Selected configurations on MovieLens-100K. Each is the point of the grid in Table 4 with the highest mean validation AUROC over the three tuning seeds.

Model	Variant	Learning rate	λ	Val. AUROC	Params
GATv2	–	10^{-3}	–	91.06	28,416
	Top-A	2×10^{-3}	1	92.46	30,496
GT	–	2×10^{-3}	–	79.34	44,928
	Top-A	10^{-3}	0.5	92.51	47,008

Directional asymmetry of the mean head-to-head transport. The paired relations *rates* and *rated-by* provide a natural probe of whether the learned transport depends on interaction direction. For every observed rating $(u, m) \in \mathcal{R}$, the graph contains the directed interactions $u \to m$ and $m \to u$. These edges connect the same pair of nodes but exchange their sender and receiver roles. For Top-A, the effective head-space transport associated with an interaction e is:

$$T_e = D_e \left(I_H + \lambda \Omega_e \right) \tag{19}$$

while the corresponding baseline is recovered by setting $\Omega_e = 0$, giving $T_e = D_e$. We average the effective transport separately over the two directions:

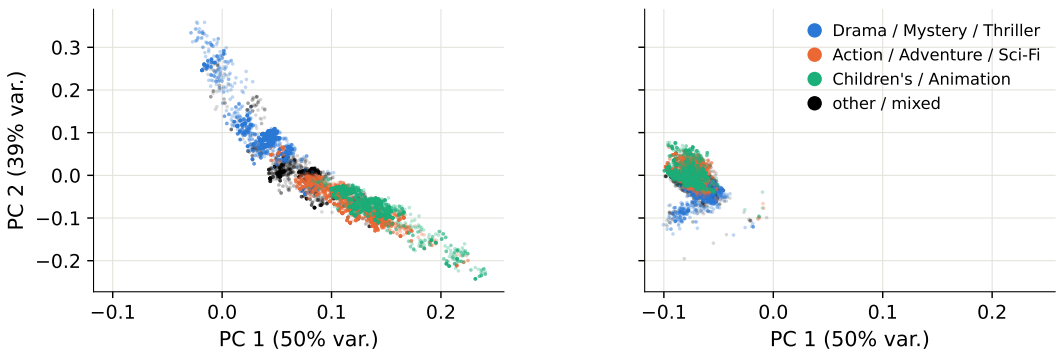
$$\overline{T}_{\text{rates}} = \frac{1}{|\mathcal{R}|} \sum_{(u,m) \in \mathcal{R}} T_{u \to m}, \qquad \overline{T}_{\text{rated-by}} = \frac{1}{|\mathcal{R}|} \sum_{(u,m) \in \mathcal{R}} T_{m \to u} \tag{20}$$

and compare their entries through the normalized directional asymmetry:

$$N_{rs} = \frac{\left| \overline{T}_{\text{rates},rs} - \overline{T}_{\text{rated-by},rs} \right|}{\left| \overline{T}_{\text{rates},rs} \right| + \left| \overline{T}_{\text{rated-by},rs} \right|} \in [0, 1] \tag{21}$$

with $N_{rs} := 0$ when both entries vanish. Here, r indexes the receiver head and s the source head. The normalization makes the comparison independent of a common rescaling of the two transports: $N_{rs} = 0$ when the corresponding route is identical in both directions, whereas $N_{rs} = 1$ when it is used in only one direction or the two directional entries have opposite signs. Figure 4 reports this quantity for the two backbones and their Top-A variants. Standard attention contains no off-diagonal transport, so directional differences are confined to the diagonal attention paths. With Top-A, the effective transport also contains off-diagonal contributions, whose directional asymmetry shows that cross-head information transfer differs between *rates* and *rated-by*. The effective head-space transport is therefore not identical across the two directed relations.

Dependence of effective transport on movie content. We next examine whether the edge-specific effective transport varies only with relation direction or also across individual interactions. For



(a) *rated-by* (movie → user) (b) *rates* (user → movie)

Figure 5: *Per-edge Transport Reflects Movie Content.* PCA of the twelve off-diagonal transport coefficients for individual edges, colored by the genre of the movie endpoint. The learned transports separate according to genre, particularly for *rated-by* edges, where distinct semantic groups occupy different regions of the projection. The effect is weaker but still visible for *rates* edges.

each edge, we vectorize the $H(H-1) = 12$ off-diagonal entries of its effective transport and project these vectors onto their first two principal components. We then color the resulting edge-level representations according to the genre of the movie endpoint. As shown in Figure 5, the effective cross-head transports exhibit structure associated with movie genre. The organization is most pronounced for *rated-by* edges, where Drama/Mystery/Thriller, Action/Adventure/Sci-Fi, and Children's/Animation movies occupy different regions of the projection, while a weaker structure is visible for *rates*. Since genre enters through the movie features rather than the link-prediction target, these results show that the effective cross-head transport varies with node content and not only with the discrete relation direction. Together with the directional analysis above, this shows that the transport realized by Top-A varies across both relation direction and individual interactions.

C.3 CLRS-30

Setup. We evaluate on the CLRS Algorithmic Reasoning Benchmark (Veličković et al., 2022), which contains execution trajectories from 30 classical algorithms drawn from the *Introduction to Algorithms* textbook (Cormen et al., 2009). The benchmark spans a broad range of computational procedures, including sorting, searching, divide-and-conquer, greedy algorithms, dynamic programming, graph algorithms, string matching, and computational geometry. This diversity makes CLRS particularly suitable for studying architectural inductive biases, since different tasks require substantially different forms of computation, such as repeated state updates, recursive reasoning, dynamic programming, and structured information propagation.

Each algorithm is represented through a graph-oriented encoding in which the objects manipulated by the algorithm are treated as nodes and their relations as edges. Depending on the task, features may be associated with nodes, edges, or the entire graph. Each execution trajectory contains the algorithm input, its final output, and a sequence of intermediate states, referred to as *hints*, which expose the evolution of the underlying computation. These hints provide intermediate supervision during training and are designed so that successive algorithmic states can generally be related through local message-passing operations.

We use the canonical `CLRS30_v1.0.0` dataset and its predefined training, validation, and test splits. Each algorithm has 1,000 training trajectories at nominal problem size $n = 16$. Validation uses the same nominal size, while testing uses $n = 64$ to assess out-of-distribution generalization, subject to the benchmark's task-specific size conventions. Validation and test splits contain 32 trajectories for most tasks, with the benchmark's prescribed multipliers for tasks with fewer prediction targets: 1,024 trajectories for maximum subarray (Kadane), and 2,048 for binary search, minimum, quickselect, both string-matching algorithms, and segment intersection.

We train a separate model for each algorithm, encoding and decoding hints and supervising intermediate predictions. Each final run uses 10,000 training steps, and the checkpoint with the highest

Table 6: CLRS-30 hyperparameter search space and final selected configurations for GATv2, GT, and their respective Top-A variants. Each variant is tuned independently using 20 Optuna TPE trials on eight core algorithms, followed by multi-seed and full-budget confirmation. Final configurations are selected using validation performance averaged over the eight core tasks and three confirmation seeds, then frozen for final evaluation. Learning rate is searched on a logarithmic scale; all other parameters use categorical choices. Selected learning rates are rounded to four significant figures. Message-passing steps are per hint transition.

		Final selected configuration			
Hyperparameter	Search space	GATv2	GATv2 + Top-A	GT	GT + Top-A
Learning rate	$[10^{-4}, 3 \times 10^{-3}]$ (log scale)	4.727×10^{-4}	4.563×10^{-4}	7.348×10^{-4}	5.416×10^{-4}
Hidden size	$\{64, 128, 256\}$	256	256	256	256
Attention heads	$\{2, 4, 8\}$	8	8	8	8
Message-passing steps	$\{1, 2\}$	1	1	1	1
Batch size	$\{16, 32\}$	32	32	32	32
Weight decay	$\{0, 10^{-4}, 10^{-3}\}$	10^{-3}	10^{-4}	10^{-4}	10^{-4}
Dropout probability	$\{0, 0.1, 0.2\}$	0.1	0.1	0.1	0.1

validation score is restored for test evaluation. For optimal BST, we omit the auxiliary `root_h` hint loss in all variants.

Hyperparameter selection. We adapt the eight-core-task, reduced-budget tuning strategy of Diao & Loynd (2023), using Optuna TPE and additional multi-seed confirmation stages while retaining the canonical CLRS-30 dataset splits. The core tasks are activity selection, Bellman–Ford, binary search, maximum subarray (Kadane), Graham scan, insertion sort, matrix-chain order, and naive string matching.

The procedure is run independently for GATv2, GT, and their respective Top-A variants, using identical search spaces, training-step budgets, trial-seed schedules, and selection criteria. Table 6 lists the search space. For each variant, we first evaluate 20 configurations on the eight core tasks, using one seed per configuration and 2,000 training steps per task. The three highest-ranked configurations are then evaluated on three confirmation seeds at the same budget. The best two proceed to confirmation at 10,000 steps, using the same three seeds.

At each stage, configurations are ranked by the mean best-validation score across the core tasks and applicable seeds. The winning configuration is used for all 30 algorithms, with 20 paired final-evaluation seeds disjoint from the search and confirmation seeds. All selected configurations use hidden size 256, eight attention heads, one message-passing step per hint transition, batch size 32, and dropout probability 0.1. These comparisons therefore assess separately tuned systems.

Additional results. Table 7 reports per-task CLRS-30 test performance for GATv2, GATv2 + Top-A, GT, and GT + Top-A, with all 30 tasks evaluated across 20 paired seeds. Entries report the mean and sample standard deviation, the paired mean improvement in percentage points, and the number of seeds on which adding Top-A strictly improves the score. The macro-average weights all 30 tasks equally; its standard deviation and win count are computed from seed-level task averages.

GATv2 + Top-A increases the macro-average from 53.22% to 59.26% (+6.04 percentage points), while GT + Top-A increases it from 49.81% to 58.36% (+8.56 percentage points). For both architectures, the task-averaged score improves in all 20 seeds.

Both architectures show particularly pronounced gains on binary search, insertion sort, bubble sort, and quicksort. GT also benefits substantially on shortest-path tasks, including Dijkstra (+21.50 percentage points), Bellman–Ford (+13.77), Floyd–Warshall (+12.23), and DAG shortest paths (+10.90). Improvements are not uniform: heapsort deteriorates for both architectures, and optimal BST exhibits the largest mean decline for GT (-15.93 percentage points).

C.4 Heterophilic Node Classification

Heterophily motivated much of the early work on SNNs: when neighboring nodes tend to carry different labels, transforming a neighboring representation before aggregation can be more expressive

Table 7: CLRS-30 test performance (official CLRS score) after separate hyperparameter optimization for each architecture and variant. Entries are mean ± sample standard deviation across 20 paired seeds, in percent (higher is better). Δ is the improvement from adding Top-A in percentage points, computed before rounding. Wins counts seeds with strictly higher scores with Top-A than without it (ties are not wins). Bold indicates the higher mean within each architecture, with ties at displayed precision unbolded; it does not denote statistical significance. All evaluations are complete. The macro-average gives equal weight to all 30 tasks; its standard deviation is computed across seed-level task averages, and its wins compare those averages across the 20 seeds. Optimal BST uses the auxiliary root-hint loss workaround described in the experimental setup.

	GATv2				GT			
Task	GATv2	GATv2 + Top-A	Δ	Wins	GT	GT + Top-A	Δ	Wins
Activity selector	**67.88** ± 4.44	65.29 ± 5.16	−2.59	7/20	**69.19** ± 4.48	66.63 ± 5.85	−2.56	6/20
Articulation points	**28.88** ± 5.21	24.24 ± 10.35	−4.64	4/20	**24.56** ± 11.04	22.27 ± 12.01	−2.29	10/20
Bellman–Ford	88.81 ± 1.18	**90.38** ± 2.36	+1.57	13/20	78.23 ± 1.61	**92.00** ± 1.12	+13.77	20/20
BFS	**99.99** ± 0.02	99.98 ± 0.02	−0.00	1/20	**99.70** ± 0.75	97.99 ± 2.07	−1.71	3/20
Binary search	40.59 ± 12.98	**80.39** ± 8.00	+39.80	20/20	5.60 ± 0.89	**79.91** ± 8.09	+74.32	20/20
Bridges	32.91 ± 5.67	**34.46** ± 7.74	+1.56	12/20	29.94 ± 0.60	**30.00** ± 0.43	+0.06	10/20
Bubble sort	10.41 ± 16.92	**54.17** ± 16.83	+43.76	18/20	5.98 ± 5.09	**51.53** ± 23.64	+45.55	20/20
DAG shortest paths	94.63 ± 2.10	**96.59** ± 0.89	+1.96	17/20	82.14 ± 4.15	**93.03** ± 4.65	+10.90	19/20
DFS	21.70 ± 4.41	**27.44** ± 6.57	+5.74	15/20	18.93 ± 5.80	**25.45** ± 6.34	+6.52	14/20
Dijkstra	80.20 ± 5.25	**88.49** ± 2.64	+8.29	20/20	69.11 ± 7.51	**90.61** ± 2.72	+21.50	20/20
Maximum subarray (Kadane)	58.10 ± 4.78	**63.61** ± 3.34	+5.51	16/20	57.19 ± 4.66	**62.91** ± 3.73	+5.72	17/20
Floyd–Warshall	**34.68** ± 2.86	31.73 ± 4.16	−2.94	5/20	27.62 ± 5.89	**39.85** ± 3.59	+12.23	20/20
Graham scan	71.40 ± 5.07	**78.61** ± 6.13	+7.20	17/20	66.70 ± 8.12	**72.96** ± 11.18	+6.26	14/20
Heapsort	**54.47** ± 25.45	28.83 ± 14.37	−25.64	4/20	**45.04** ± 20.21	33.43 ± 21.74	−11.61	7/20
Insertion sort	28.49 ± 20.03	**84.49** ± 5.55	+56.00	19/20	37.24 ± 21.72	**87.33** ± 3.86	+50.10	19/20
Jarvis march	58.95 ± 2.30	**60.08** ± 2.25	+1.13	13/20	61.96 ± 4.88	**62.27** ± 2.32	+0.32	10/20
KMP matcher	0.00 ± 0.01	**0.05** ± 0.12	+0.05	4/20	0.05 ± 0.10	**0.78** ± 1.14	+0.73	16/20
LCS length	84.81 ± 0.97	**85.21** ± 0.39	+0.40	13/20	**85.33** ± 0.63	84.86 ± 1.42	−0.48	9/20
Matrix chain order	77.98 ± 6.92	**83.50** ± 1.58	+5.51	17/20	**86.48** ± 5.30	83.92 ± 1.48	−2.56	2/20
Minimum	**86.36** ± 4.84	83.39 ± 8.26	−2.97	8/20	**90.16** ± 3.80	88.48 ± 3.61	−1.67	5/20
MST (Kruskal)	**64.00** ± 6.38	56.98 ± 10.83	−7.02	5/20	**73.12** ± 4.03	69.57 ± 4.91	−3.56	6/20
MST (Prim)	59.70 ± 7.71	**66.28** ± 7.23	+6.58	13/20	52.26 ± 8.63	**60.76** ± 8.21	+8.51	15/20
Naive string matcher	2.79 ± 1.70	**3.03** ± 1.53	+0.25	11/20	2.74 ± 1.60	**3.06** ± 1.22	+0.32	12/20
Optimal BST	**57.83** ± 9.23	52.42 ± 17.87	−5.41	7/20	**57.21** ± 12.46	41.28 ± 21.51	−15.93	8/20
Quickselect	0.00 ± 0.00	**0.40** ± 1.18	+0.40	5/20	0.01 ± 0.02	**0.33** ± 1.50	+0.33	1/20
Quicksort	7.36 ± 5.14	**50.20** ± 14.13	+42.84	20/20	6.99 ± 6.62	**44.39** ± 25.57	+37.40	18/20
Segments intersect	**80.43** ± 0.67	79.19 ± 0.79	−1.24	4/20	**81.47** ± 0.38	79.53 ± 0.74	−1.94	0/20
Strongly connected comp.	32.68 ± 4.80	**34.22** ± 7.24	+1.54	15/20	35.42 ± 3.53	**38.54** ± 6.56	+3.11	14/20
Task scheduling	80.57 ± 1.30	**82.55** ± 1.15	+1.98	17/20	80.76 ± 1.04	**81.25** ± 1.53	+0.49	13/20
Topological sort	90.03 ± 12.34	**91.55** ± 7.88	+1.52	11/20	63.13 ± 4.77	**66.00** ± 17.63	+2.87	13/20
Macro-average (30 tasks)	53.22 ± 1.26	**59.26** ± 1.48	+6.04	20/20	49.81 ± 1.73	**58.36** ± 1.20	+8.56	20/20

than merely reweighting it (Bodnar et al., 2022). Since Top-A implements a matrix-valued transport mechanism within multi-head attention, the benchmarks of Platonov et al. (2023) provide a natural contrast setting in which to ask whether cross-head transport provides a consistent advantage on heterophilic graphs. This question is also relevant in light of recent work questioning whether the empirical advantages of sheaf models on heterophilic graphs can be attributed specifically to learned sheaf structure (Hernandez Caralt et al., 2026; Fiorini et al., 2026).

Setup. We evaluate on the seven benchmarks of Platonov et al. (2023), using the ten official train/validation/test splits provided with each dataset. The multi-class datasets (*roman-empire*, *amazon-ratings*, *squirrel*, and *chameleon*) are evaluated by accuracy, while the binary datasets (*minesweeper*, *tolokers*, and *questions*) are evaluated by ROC-AUC. Early stopping and hyperparameter selection use the corresponding validation metric. Within each backbone family, the baseline and its Top-A variant share the same encoder, processor depth, decoder, optimization schedule, data split, and seed. Each Top-A run is initialized from the corresponding same-seed baseline, while the routing generator is zero-initialized, so that the two models implement the same function before training. All models use $H = 4$ attention heads and five message-passing layers, with head width 64 for GATv2 and 56 for the GT. Parameter counts for all models are reported in Table 10. Each arm is given the same hyperparameter search over learning rate $\in \{10^{-3}, 3 \cdot 10^{-4}\}$ and dropout $\in \{0, 0.2, 0.5\}$, tuned on split 0 using two seeds and selected by validation performance alone. Before training, every Top-A run is verified to match its paired baseline at $\Omega_e := 0$.

Table 8: Performance comparison on the datasets of Platonov et al. (2023). AUROC is reported for *minesweeper*, *tolokers*, and *questions*, while accuracy is reported for the remaining datasets. We additionally report GATv2 and GT with and without Top-A.

Model **Edge Homophily**	**roman-empire** 0.05	**amazon-ratings** 0.38	**minesweeper** 0.68	**tolokers** 0.59	**questions** 0.84	**squirrel** 0.20	**chameleon** 0.23
Baselines							
GCN	73.69 ± 0.74	48.70 ± 0.63	89.75 ± 0.52	83.64 ± 0.67	76.09 ± 1.27	39.47 ± 1.47	40.89 ± 4.12
SAGE	85.74 ± 0.67	53.63 ± 0.39	93.51 ± 0.57	82.43 ± 0.44	76.44 ± 0.62	36.09 ± 1.99	37.77 ± 4.14
GAT	80.87 ± 0.30	49.09 ± 0.63	92.01 ± 0.68	83.70 ± 0.47	77.43 ± 1.20	35.62 ± 2.06	39.21 ± 3.08
GAT-sep	88.75 ± 0.41	52.70 ± 0.62	93.91 ± 0.35	83.78 ± 0.43	76.79 ± 0.71	35.46 ± 3.10	39.26 ± 2.50
GT	86.51 ± 0.73	51.17 ± 0.66	91.85 ± 0.76	83.23 ± 0.64	77.95 ± 0.68	36.30 ± 1.98	38.87 ± 3.66
GT-sep	87.32 ± 0.39	52.18 ± 0.80	92.29 ± 0.47	82.52 ± 0.92	78.05 ± 0.93	36.66 ± 1.63	40.31 ± 3.01
CO-GNN	89.44 ± 0.50	54.20 ± 0.34	97.35 ± 0.63	84.84 ± 0.96	75.97 ± 0.89	39.39 ± 2.76	41.14 ± 5.40
Sheaf Neural Networks							
O(d)-NSD	80.41 ± 0.72	42.76 ± 0.54	92.15 ± 0.84	78.83 ± 0.76	69.69 ± 1.46	35.79 ± 3.34	37.93 ± 2.24
BuNN	91.75 ± 0.39	53.74 ± 0.51	98.99 ± 0.16	84.78 ± 0.80	78.75 ± 1.09	–	–
CSNN	92.63 ± 0.50	52.07 ± 1.00	99.07 ± 0.25	85.45 ± 0.53	79.31 ± 1.22	41.18 ± 2.23	43.09 ± 3.17
Topological Attention Baselines and Models							
GATv2	87.64 ± 0.56	44.16 ± 0.52	94.90 ± 0.35	84.82 ± 0.76	59.51 ± 6.51	37.68 ± 2.53	40.00 ± 3.91
GATv2 + Top-A (ours)	87.81 ± 0.41	45.23 ± 1.07	94.96 ± 0.48	85.09 ± 0.75	58.59 ± 7.03	36.96 ± 2.12	40.09 ± 3.53
GT	88.10 ± 0.40	52.75 ± 0.56	94.01 ± 0.32	77.89 ± 7.76	60.08 ± 3.45	38.17 ± 2.27	41.19 ± 3.32
GT + Top-A (ours)	89.00 ± 0.53	52.69 ± 0.67	93.97 ± 0.29	84.76 ± 0.67	60.57 ± 4.49	38.39 ± 2.33	41.11 ± 3.83

Results. As shown in Table 8, cross-head routing does not provide a systematic advantage on these heterophilic benchmarks. Across the 14 dataset–backbone comparisons, most changes are small and vary in sign across datasets and architectures. The clearest exception is the GT on *tolokers*, where Top-A improves ROC-AUC from 77.89 to 84.76. A smaller positive effect is also observed for the GT on *roman-empire* (+0.90 accuracy points), while the remaining comparisons remain close to their corresponding baselines. Overall, adding Top-A does not consistently improve performance over the corresponding attention backbone in this benchmark suite.

What heterophily does and does not imply. These results help clarify the role of the transport inductive bias that motivates Top-A. Heterophily was one of the original settings in which matrix-valued sheaf transport was argued to be useful, since neighboring representations may need to be transformed rather than simply reweighted. By isolating a related matrix-valued transport mechanism within attention, we can ask whether this additional flexibility is itself sufficient to yield systematic gains on heterophilic graphs. The results suggest that it is not: cross-head transport provides no consistent advantage across these benchmarks. Together with the main experiments, this supports a more specific view of the inductive bias, whose usefulness appears most clearly in settings where individual interactions call for different transformations before aggregation.

Hyperparameters and Selected Configurations Table 9 reports the complete configuration used for heterophilic node classification. Only the learning rate and dropout are searched, using the same six-cell grid for every arm; all remaining settings are fixed in advance. Hyperparameters are selected according to validation performance, following the standard protocol for these benchmarks. Table 10 reports the selected configuration for each dataset and model together with its validation score and parameter count.

Table 9: *Heterophilic Node Classification: Hyperparameters.* The same hyperparameter grid is assigned to every arm and selected on validation performance.

Hyperparameter	**GATv2**	**GT**
Tuned per arm (identical grid, validation-selected)		
Learning rate	10^{-3} / $3 \cdot 10^{-4}$	
Dropout (= input dropout)	0 / 0.2 / 0.5	
Architecture		
Attention heads H	4	4
Layers	5	5
Channels per head	64	56
Residual connections	yes	yes
LayerNorm in the block	–	yes
Activation	–	ReLU
LeakyReLU negative slope	0.2	–
Attention dropout	0	–
Output ℓ_2 normalisation	yes	yes
Self-loops added	yes	yes
Jumping knowledge	no	no
Optimisation		
Optimiser	Adam	Adam
Weight decay	$5 \cdot 10^{-4}$ / 10^{-3}	$5 \cdot 10^{-4}$ / 10^{-3}
Max. epochs	1000	1000
Early-stopping patience	200	200
Top-A transport		
Route scale λ	1	1
Route dropout	0	0
Routing initialization	zero	zero

Table 10: *Heterophilic Node Classification: Selected Configurations.* Learning rate and dropout chosen for each arm by mean validation score from Table 9. A dash in the variant column marks the plain backbone; Top-A denotes the same backbone augmented with cross-head routing.

Dataset	**Model**	**Variant**	**Learning rate**	**Dropout**	**Val.**	**Params**
Roman-empire	GATv2	–	0.001	0.2	88.47	1,069,842
	GATv2	Top-A	0.001	0.5	88.17	1,080,162
	GT	–	0.001	0.5	88.59	1,081,714
	GT	Top-A	0.001	0.5	89.55	1,099,714
Amazon-ratings	GATv2	–	0.0003	0.2	44.02	1,066,501
	GATv2	Top-A	0.0003	0.2	45.12	1,076,821
	GT	–	0.0003	0.2	52.43	1,078,789
	GT	Top-A	0.0003	0.2	52.29	1,096,789
Squirrel-filt.	GATv2	–	0.001	0	39.35	1,524,485
	GATv2	Top-A	0.001	0.2	39.14	1,534,805
	GT	–	0.001	0	39.83	1,479,525
	GT	Top-A	0.001	0.2	40.32	1,497,525
Chameleon-filt.	GATv2	–	0.001	0.5	47.04	1,584,901
	GATv2	Top-A	0.001	0.5	47.21	1,595,221
	GT	–	0.0003	0.5	47.04	1,532,389
	GT	Top-A	0.0003	0.5	46.86	1,550,389
Minesweeper	GATv2	–	0.001	0.2	94.02	990,722
	GATv2	Top-A	0.001	0.2	94.05	1,001,042
	GT	–	0.0003	0.2	93.26	1,012,482
	GT	Top-A	0.0003	0.2	93.17	1,030,482
Tolokers	GATv2	–	0.001	0	85.05	991,490
	GATv2	Top-A	0.001	0	84.97	1,012,050
	GT	–	0.001	0.2	85.35	1,013,154
	GT	Top-A	0.0003	0.2	84.60	1,031,154
Questions	GATv2	–	0.001	0.5	64.72	1,065,986
	GATv2	Top-A	0.001	0.5	64.87	1,076,306
	GT	–	0.001	0.2	63.26	1,078,338
	GT	Top-A	0.001	0.2	64.79	1,096,338