REUSABLE GRAPH REPRESENTATIONS FOR PATH PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph–path learning predicts properties of ordered paths in shared graphs, with applications in network routing and comparative biology. In this article, we advocate for caching graph representations: sharing their encoding cost across many path queries is crucial for large-scale applications. We make three contributions. First, we introduce the Edge–Vertex Transformer (EVT), which couples vertex states with direction-aware pair attention, and an aligned graph-to-path construction that separates reusable graph encoding from path processing. Second, we characterize the expressivity of this model by a path-indexed two-dimensional folklore Weisfeiler–Leman (2-FWL) profile. Every compatible continuous prediction can be uniformly approximated on compact domains using one parameter set shared across jointly bounded graph and path sizes. We also establish a strict hierarchy between selection, joint cache reading, and path-marked encoding. Third, we build a coherent benchmarking suite combining controlled synthetic tasks, temporal routing measurements from RIPE Atlas, and ThorAxe-Combo, a curated biological benchmark of splicing tasks. The experiments distinguish information access from finite-resource learning: the most elaborate reusable pipeline benefits synthetic composition tasks and Atlas prediction, while simpler pair processors lead on ThorAxe-Combo. Our benchmarks also assess the computational gains enabled by graph-cache reuse. Together, these contributions identify when relational path processing justifies its computational cost.

1 INTRODUCTION

Many prediction problems concern an ordered path within a larger graph: network measurements attach delays to Internet routes (RIPE NCC Staff, 2015; RIPE NCC, 2026), while genomic and evolutionary splicing graphs place related biological sequences in a shared context (Garrison et al., 2018; Liao et al., 2023; Zea et al., 2021). The same graph often supports many queries. Learning should therefore exploit both the relations along each path and the opportunity to reuse the surrounding graph representation.

Models must therefore preserve path order and repeated visits, use graph context invariantly to vertex relabeling, and answer multiple queries efficiently. Two strategies differ in when the path enters the computation. Labeling methods mark its vertices and, when relevant, their positions before encoding (Wang et al., 2025; Zhou et al., 2023; Xu & Zou, 2024) (Figure 1a). This produces query-specific states, including off-path context, and requires re-encoding for each query. Alternatively, encoding once gives reusable states for subsequent paths (Figure 1b). We compare two cache readers: selecting states between path occurrences, and jointly combining their relations to each graph vertex before aggregation (Figure 1c). Reusing an encoding spreads its cost across queries. These interfaces raise three questions: what information a path predictor can access, how finite models learn to use it, and how much computation can be saved by reusing graph encodings.

Prior work relates the expressive power of graph encoders to Weisfeiler–Leman refinement, including pair-based networks (Maron et al., 2019a) and Edge Transformers (ET) (Müller et al., 2024). It also explains how marking and aggregation affect expressivity (Wang et al., 2025; Zhou et al., 2023; Zhang et al., 2023; Feng et al., 2023) and connects distinguishability to approximation (Azizian & Lelarge, 2021). We extend these tools to characterize exactly what ordered selection preserves from a stable 2-FWL pair cache and establish relative approximation with parameters shared across bounded

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.

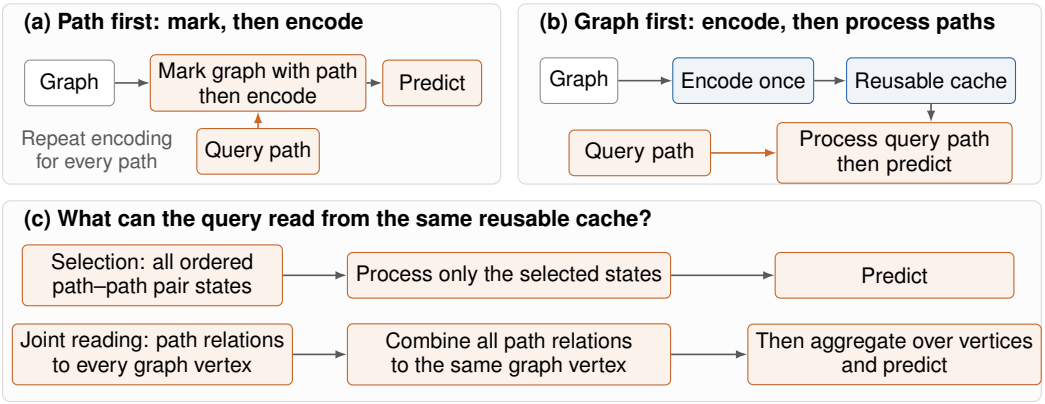


Figure 1: Two encoding strategies and three query interfaces. **(a)** Mark the graph with the ordered path, then encode it for that query. **(b)** Encode the graph once, then process each path from reusable states. **(c)** Within the reusable strategy, selection reads path–path states; joint reading combines all path relations to each graph vertex before aggregating over vertices. Blue operations are shared across queries; orange operations depend on the path. A two-stage selection model is detailed in Figure G.1.

graph and path sizes. Complementing the pooling-order analysis of Zhang et al. (2023), we establish a hierarchy, with strict separating examples, between ordered path-pair selection, joint reading of the same stable bidirectional 2-FWL cache, and occurrence-marked refinement (Proposition 4); further comparisons are in Appendix K.

We study the reusable selection interface with two-stage graph-to-path models that separate graph encoding, ordered selection, and path processing (Figure 1b–c). A graph encoder computes vertex and ordered-pair states before the query; a path processor receives their ordered selection, with position and repeated-vertex indicators. Our graph encoder, Edge–Vertex Transformer (EVT), combines triangular attention in four orientations with vertex–pair exchanges. Its triangular attention draws on ET (Bergen et al., 2021) and AlphaFold (Jumper et al., 2021), with different score and value constructions. We call the two-EVT variant TT, where T denotes triangular pair attention at each stage. Within the same interface, we compare it with variants using pair multilayer perceptrons (MLPs) or dense attention after selection, to study these operator choices at fixed model and training budgets (Section 2).

The biological component, ThorAxe-Combo, tests combinations of splicing choices rather than individual segments. Coordinated choices across distant positions (Tilgner et al., 2015; Santos-Rodriguez et al., 2026) motivate this graph–path task on annotated isoforms.

Contributions. Our contributions connect architecture, expressivity, and empirical design:

1. **EVT and reusable graph-to-path models.** We introduce a graph Transformer coupling vertices and direction-aware ordered pairs, together with an aligned occurrence-level selector and path processor. This construction caches graph context for many queries; TT applies EVT twice, while simpler variants isolate the role of each stage (Section 2).
2. **Expressivity across bounded graph and path sizes.** A path-indexed bidirectional 2-FWL profile exactly characterizes the idealized TT family's common kernel and uniform approximation class. The result uses one parameter set across all graph and path sizes up to fixed bounds. We establish a strict hierarchy of selection, joint reading, and marked encoding, and show that sufficiently expressive non-triangular path decoders can attain the selected profile (Section 3). This separates information limits from finite-resource operator choices.
3. **A coherent synthetic and real-data benchmarking suite.** Synthetic constructions connect theory to composition and context tests; curated RIPE Atlas measurements test temporal routing prediction; ThorAxe-Combo tests biological combinations on unseen genes (16,544 candidates from 1,072 genes). Shared splits, documented inputs, internal model selection, and same-weight reuse measurements support an extensive study of architectural tradeoffs, including cases where simpler models prevail (Section 4).

2 REUSABLE GRAPH-TO-PATH MODELS

We introduce EVT as a vertex–pair graph Transformer and combine it with an aligned caching mechanism that reuses graph encodings across ordered path queries.

An input consists of a graph G, with vertex set $V = [n] = \{1, \dots, n\}$, vertex features $X \in \mathbb{R}^{n \times d_X}$, and dense ordered-pair features $Y \in \mathbb{R}^{n \times n \times d_Y}$, together with an ordered query $P = (p_1, \dots, p_k) \in V^k$. We write $X = (x_i)$, $Y = (y_{ij})$; d_X, d_Y are the input feature dimensions. Here $s \in [k] = \{1, \dots, k\}$ is a path position and $p_s \in V$ is the vertex at that position. We call P a query path: any ordered vertex sequence is accepted, although applications may require consecutive vertices to be adjacent. The tensor Y stores features for all ordered vertex pairs, including self-pairs and pairs without an edge. These features encode adjacency and any edge attributes. We consider graphs with at most N vertices and paths with at most K positions. The same model parameters apply to all inputs with $n \le N$ and $k \le K$, producing a q-dimensional output at each of the k path positions. Predictions are invariant to consistent vertex relabeling but may depend on path order. Notation, input and output spaces, and relabeling symmetry are detailed in Appendices A and G.2.

The EVT block. EVT maintains vertex and ordered-pair states, also denoted X, Y at each layer. Each layer first applies vertex self-attention and pair triangular attention in parallel. We define the two oriented views of a pair state by

$$y_{ij}^{\rightarrow} := y_{ij}, \qquad y_{ij}^{\leftarrow} := y_{ji}. \tag{1}$$

To update pair (i, j), a triangular head combines y_{ir}^{α} and y_{rj}^{β} through an intermediate vertex r. Each head may choose its own pair of orientations. The choices $(\alpha, \beta) = (\rightarrow, \rightarrow), (\rightarrow, \leftarrow), (\leftarrow, \rightarrow), (\leftarrow, \leftarrow)$ correspond respectively to a directed chain, a common target, a common source, and a reverse chain. Aligned multilayer perceptrons (MLPs) exchange self-pair and endpoint information through AlMLP_1. Writing $\mathbf{1}_{\{E\}}$ for the indicator of a condition E, their updates are

$$\mathrm{AlMLP}_a(X, Y) := \left((\mathrm{MLP}_{X,a}(x_i, y_{ii}))_i,\ (\mathrm{MLP}_{Y,a}(y_{ij}, x_i, x_j; \mathbf{1}_{\{i=j\}}))_{ij} \right), \qquad a \in \{1, 2\}. \tag{2}$$

Each MLP acts on the concatenation of its listed inputs. We suppress its learnable parameter indices; parameters are shared across vertices or pairs, but not between branches or layers. A second step applies dense cross-attention between vertex states and the flattened list of pair states in both directions, followed by AlMLP_2, with the same form and independent parameters. All attention and MLP increments are added residually. Figure G.1a shows the layer diagram; operator definitions (attention, MLPs, normalization, and reshaping) are given in Appendix G.1. Experiments use fixed widths and depths and LayerNorm before attention and MLPs. The theory considers an idealized family with arbitrary finite capacity and no feature normalization (Definition H.4).

Two-stage construction and the TT variant. The two-stage family separates a reusable graph stage from a query-specific path stage. Its TT variant uses EVT at both stages (Figure G.1b). The experimental variants are defined together at the start of Section 4; each is trained independently.

With graph-stage parameters θ_{G}, an EVT first computes graph representations independently of the query:

$$(\widetilde{X}, \widetilde{Y}) = \mathrm{EVT}^{\mathrm{G}}_{\theta_{\mathrm{G}}}(X, Y). \tag{3}$$

The ordered selector indexes the cached states using the given path P, retaining its order and repeated visits. We call this exact occurrence-level pullback aligned graphical context: each path position and pair receives the representation of its graph vertex and ordered graph pair. We write $P \times P$ for all ordered pairs of path occurrences and $P \times V$ for all pairs of a path occurrence and a graph vertex:

$$\widetilde{Y}[P, P] := (\widetilde{y}_{p_s p_t})_{s,t \in [k]}, \qquad \widetilde{Y}[P, V] := (\widetilde{y}_{p_s u})_{s \in [k],\, u \in V}. \tag{4}$$

For fixed graph inputs and encoder weights, the cache is independent of all subsequent queries; a graph or weight update requires recomputing it. These arrays contain k^2 ordered path-occurrence pairs and $k \times n$ path-to-graph pairs, respectively; V includes the path vertices. Selection includes self-pairs and non-adjacent pairs, and may already encode off-path information acquired during graph encoding.

TT selects $\widetilde{X}[P] = (\widetilde{x}_{p_s})_{s\in[k]}$ and $\widetilde{Y}[P,P]$, then appends one-hot position codes $\phi(s) = (\mathbf{1}_{\{r=s\}})_{r=1}^{K} \in \mathbb{R}^K$ and $\psi(s,t) = \phi(s) \,\|\, \phi(t)$. Here $\|$ denotes concatenation. For the idealized theoretical family, shared injective maps ι_X and ι_Y preserve the selected states:

$$\widehat{x}^0_s = \iota_X(\widetilde{x}_{p_s}) \,\|\, \phi(s), \qquad \widehat{y}^0_{st} = \iota_Y(\widetilde{y}_{p_s p_t}) \,\|\, \psi(s,t) \,\|\, \mathbf{1}_{\{p_s=p_t\}}. \tag{5}$$

The position codes distinguish the visits; the equality indicator records that they refer to the same graph vertex. Second, an EVT with independent parameters θ_{P} refines the selected occurrence-indexed states:

$$(X^{\mathrm{P}}, Y^{\mathrm{P}}) = \mathrm{EVT}^{\mathrm{P}}_{\theta_{\mathrm{P}}}(\widehat{X}^0, \widehat{Y}^0). \tag{6}$$

The idealized model outputs q coordinates from each final vertex state; pair information reaches these outputs through vertex–pair exchanges (Equation (17)). In practice, learned affine maps transform the selected states and appended indicators into the second stage's hidden width. Task-specific prediction heads then read vertex states $Z = X^{\mathrm{P}}$, pair states $R = Y^{\mathrm{P}}$, or both (Appendix G.2). All operations commute with graph relabeling, giving Equation (21). Let n be the number of graph vertices, L_G and L_P the numbers of EVT layers in the graph and path stages, and d their common hidden width. For TT, the leading dense computational cost for Q query paths of lengths $k_1, \ldots, k_Q$ is $O(L_G n^3 d + \sum_{a=1}^{Q} L_P k_a^3 d)$, up to channel-mixing terms. The terminal cache costs $O(n^2 d)$; retaining all graph layers costs $O(L_G n^2 d)$. Directional strictness is established in Appendix J.1. Optional positional and streaming-memory extensions, not used in the reported experiments, are described in Appendices J.2 and J.3.

3 Expressivity of Reusable Graph-to-Path Models

We characterize the information available to reusable selection models, then compare their expressivity with joint cache reading and query marking. Throughout, graph sizes and path lengths satisfy $n \le N$ and $k \le K$. The theoretical families allow arbitrary finite capacity but use a single parameter set across all these sizes.

3.1 Exact expressivity and approximation from a selected cache

For a map f, its kernel is the indistinguishability relation $\ker(f) := \{(u,v) : f(u) = f(v)\}$. For a family $\mathcal{F}$, the common kernel $\mathrm{Ker}(\mathcal{F}) := \bigcap_{f\in\mathcal{F}} \ker(f)$ contains precisely the input pairs that no member can distinguish. Any uniformly approximable target must agree on these pairs. For the idealized TT family $\mathcal{H}_{N,K}$, we prove the converse for continuous targets on compact domains, thereby connecting distinguishability to approximation. This family uses the two EVT stages of Definition H.4, without feature normalization and with arbitrary finite widths and depths.

Bidirectional 2-FWL. The two-dimensional folklore Weisfeiler–Leman (2-FWL) refinement assigns query-independent pair colors c^r_{ij} after r rounds on the full graph by the following rule, where multisets retain multiplicities:

$$c^0_{ij} = (y_{ij}, y_{ji}, x_i, x_j, \mathbf{1}_{\{i=j\}}), \qquad c^{r+1}_{ij} = \left(c^r_{ij}, \left\{\!\left\{(c^r_{iu}, c^r_{uj}) : u \in [n]\right\}\!\right\}\right). \tag{7}$$

Colors retain exact input features in a common tuple–multiset representation, so they can be compared across graph sizes. Including both y_{ij} and y_{ji} matches EVT's recovery of reversed pair states through oriented triangular attention. Each c^r_{ij} determines the reversed-pair color c^r_{ji}; the multiset therefore determines all four attention orientations (Lemma H.7). With $R_N = 2N^2$, define

$$c^\star_{ij} := c^{R_N}_{ij}, \qquad C^\star_N(X,Y) := (c^\star_{ij})_{i,j\in[n]}. \tag{8}$$

This common depth gives a stable equality partition in every two-graph comparison with sizes at most N (Lemma H.6). At the graph stage, EVT has exactly this stable distinguishing power, matching direction-mixed ET with endpoint-initialized pair states (Proposition H.2); finite-depth comparisons are separate (Proposition H.3). We extend the stable characterization to ordered path selection and approximation with parameters shared over both bounded graph sizes and bounded path lengths.

Definition 1 (Path-indexed 2-FWL profile)**.** Let $\mathcal{D}_{\le N,\le K}$ contain the inputs (X,Y,P) with at most N vertices and K path positions. For an input with n vertices and k positions, its selected profile is

$$\Psi_{N,K}(X,Y,P) := (n, k, C^\star_N(X,Y)[P,P]) = \left(n, k, (c^\star_{p_s p_t})_{s,t\in[k]}\right). \tag{9}$$

The profile retains occurrence order and repeated-vertex equality, while remaining invariant to graph relabeling. Write $\mathcal{C}_\Psi(S)$ for the continuous, length-preserving predictions on $S \subset \mathcal{D}_{\leq N,\leq K}$, with q outputs per position, that agree on identical profiles. The uniform norm takes the supremum of Frobenius output errors over S (Appendix G.2).

Theorem 2 (Path-profile characterization and relative universality)**.** *On $\mathcal{D}_{\leq N,\leq K}$, the idealized TT family satisfies $\mathrm{Ker}(\mathcal{H}_{N,K}) = \ker(\Psi_{N,K})$. Moreover, for every compact $S \subset \mathcal{D}_{\leq N,\leq K}$, $\overline{\{H|_S : H \in \mathcal{H}_{N,K}\}}^{\|\cdot\|_{\infty,S}} = \mathcal{C}_\Psi(S)$. Equivalently, for each $F \in \mathcal{C}_\Psi(S)$ and $\varepsilon > 0$, one finite TT model satisfies $\|H - F\|_{\infty,S} < \varepsilon$ simultaneously over all sizes represented in S.*

Proof sketch. Rooted-color factorization gives the upper bound. Conversely, EVT realizes common separating color codes on each finite input set across sizes; diagonal indicators recover multiplicities despite attention normalization. A positioned decoder recovers the selected matrix and interpolates finite targets. On a compact domain, finitely many separators cover the pairs on which a continuous target differs appreciably; exact parallel evaluation and a shared readout yield one uniform approximant. Full proofs are in Appendices H.5, H.7 and H.8. □

Thus the selected profile is a sharp information boundary, not merely an upper bound on TT: every compatible continuous target is approximable. Capacity may depend on the target, compact domain, and accuracy; the theorem does not give the same guarantee for fixed-capacity models.

Remark 3 (No triangular path attention is needed for expressivity). Keep the idealized graph encoder, positioned pair MLPs, and global attention, but set every path-stage triangular increment to zero. The resulting subfamily $\mathcal{H}^{\mathrm{dec}}_{N,K}$ has the same kernel and compact approximation class as $\mathcal{H}_{N,K}$ (Corollary H.9). The construction can require substantial auxiliary width (Appendix H.9); triangular path attention can therefore still matter at finite model and training budgets.

3.2 An expressivity hierarchy of graph-to-path interfaces

Joint $P \times V$ reading uses $\Gamma_{N,K} = (n, k, \{\{(c^\star_{p_s u})_{s\in[k]} : u \in V\}\})$: it combines all path relations to the same graph vertex before pooling over vertices. Both Γ and Ψ read the same unmarked, reusable graph cache (Figure 1b–c). In contrast, marking annotates each vertex with all its occurrence positions before refinement (Figure 1a): append $m_P(i) = \sum_{s:p_s=i} e_s$ to x_i, where e_s is the s-th standard basis vector of $\mathbb{R}^K$. Writing $\bar{c}^\star$ for the resulting colors after R_N rounds, the marked profile is $\Psi^{\mathrm{mark}}_{N,K} = (n, k, (\bar{c}^\star_{p_s p_s})_{s\in[k]})$. We suppress N, K in profile subscripts when unambiguous.

For a profile A, let $\mathcal{F}_A$ be the family of all length-preserving predictions that factor through A. These are idealized information-access families with unrestricted readers, not prescribed finite neural architectures.

Proposition 4 (Expressivity hierarchy of access profiles)**.** *For stable bidirectional 2-FWL with occurrence-identifying marks,*

$$\mathcal{F}_\Psi \subseteq \mathcal{F}_\Gamma \subseteq \mathcal{F}_{\Psi^{\mathrm{mark}}}, \qquad \ker \Psi^{\mathrm{mark}} \subseteq \ker \Gamma \subseteq \ker \Psi. \tag{10}$$

Both inclusions are strict for $N \geq 16$ and $K \geq 3$, even on undirected binary graphs with constant vertex features and three-position simple paths. On queries with $k \leq 2$, $\mathcal{F}_\Gamma = \mathcal{F}_\Psi$ and $\ker \Gamma = \ker \Psi$. Marking is already strictly stronger than these two interfaces at $k = 1$ when $N \geq 16$.

Proof sketch. The diagonal color in coordinate t identifies the unique tuple of Γ belonging to $u = p_t$, recovering column t of Ψ. A stable marked diagonal recovers all marked path-to-vertex relations jointly; forgetting their marks yields Γ. For one or two positions, stable pair-color reconstruction also recovers Γ from Ψ. These reconstructions imply the family inclusions in the opposite order to the kernels. Rook–Shrikhande graph pairs witness strictness. The complete proof is in Appendix I.1. □

By Theorem 2, TT attains the selected interface's distinguishability and continuous compact approximation class. Idealized marked EVT and ET similarly attain the marked profile's kernel and all compatible finite lookup tables (Corollary I.2). The middle family is an information-level joint reader; the proposition does not assert that any particular fixed-capacity implementation realizes it, or that practical model parameterizations contain one another. The comparison requires stable refinement and occurrence-identifying marks, not merely binary path-membership flags.

Remark 5 (Cost of joint cache reading). Joint reading accesses kn states per query instead of the k^2 states used by selection: materializing them at width d requires $O(knd)$ rather than $O(k^2d)$ storage. Both reuse the $O(n^2d)$ graph cache, but joint reading retains an n-dependent query cost. Total work and peak working memory depend on the reader and its implementation.

3.3 Sharpness of the expressivity inclusions

We study sharpness through theoretical separating examples and a numerical realization of the gap between selection and joint reading.

Theoretical separation. The strict witnesses in Proposition 4 show that both additional forms of access can reveal genuinely new information. For a numerical test of the first inclusion, we use graphs from Latin squares of order seven: vertices are the 49 cells, and distinct cells are adjacent if they share a row, column, or symbol. Their degrees and common-neighbor counts satisfy the graph conditions below (Nakasora, 2006, Proposition 3.2).

Proposition 6 (Certified selection and aggregation separation)**.** *Let G be a simple undirected graph with 49 vertices, degree 18 at every vertex, and respectively 7 and 6 common neighbors for adjacent and distinct nonadjacent pairs. Give all vertices the same feature and use binary adjacency as the only pair feature. Let $\mathcal{Q}$ be a nonempty collection of ordered triangles $P = (p_1, p_2, p_3)$ with distinct vertices, balanced between the two labels $t(P) = |N_G(p_1) \cap N_G(p_2) \cap N_G(p_3)| \in \{0, 1\}$, where $N_G(v)$ is the neighbor set of v. For $N \geq 49$ and $K \geq 3$, the selected profile $\Psi_{N,K}$, the separate row histograms $(\{\{c^\star_{p_i u} : u \in V\}\})_{i=1}^3$, and the separate two-row histograms $(\{\{(c^\star_{p_i u}, c^\star_{p_j u}) : u \in V\}\})_{i<j}$ are constant on $\mathcal{Q}$. Any deterministic binary classifier based on these summaries therefore has accuracy exactly 50% on $\mathcal{Q}$, whereas a classifier using $\Gamma_{N,K}$ attains 100%.*

The proof is given in Appendix C, under "Detailed proof of Proposition 6".

Our certified graphs each supply 16 queries with $t = 1$ and 16 with $t = 0$. This balance is a query-selection condition, not a claim about all triangles or all Latin squares.

Numerical realization. We train two heads on identical $P \times V$ states from a randomly initialized, frozen EVT encoder. Pairwise pooling aggregates each pair of rows separately before combining their summaries; joint reading combines all three relations to each vertex before pooling. Only the heads are trained, on eight queries from one graph, for 1,000 AdamW updates at learning rate 10^{-3}, with three shared encoder seeds (17, 29, 43). We fix the protocol and final-step checkpoints during development, then generate 20 new graphs, certified pairwise non-isomorphic and non-isomorphic to the previous structures, giving 640 test queries.

On these 640 queries, joint reading reaches 100% accuracy for each of the three seeds, while pairwise pooling stays at its certified 50% ceiling (Table C.1 in Appendix C). This tests an aggregation distinction on a shared cache, not a parameter- or computation-matched architecture comparison. The new graphs have the same two label-specific joint relation histograms as the development graphs: the experiment tests transfer of this distinction across non-isomorphic structures of the same size, not extrapolation to new joint histograms. Marking's further strict gain is established theoretically, not by this numerical comparison. Protocols and certificates are in Appendix C.

4 Synthetic and Real-Data Benchmarking

The supplementary material provides software to reconstruct the reported numerical summaries and rerun the main experiments. It includes a reusable computational toolbox for graph–path models and a benchmarking toolbox for data preparation, training and evaluation.

Table 1 defines the graph and path stages of the compared models. In two-letter names, the first letter specifies graph-pair attention and the second path-pair attention: T is triangular and D is dense.

We contribute a three-part benchmarking suite that connects controlled mechanism tests to curated graph–path prediction tasks. Synthetic datasets separate composition of selected relations from use of graph context; an Internet-routing benchmark tests temporal prediction and computational reuse; ThorAxe-Combo tests generalization to unobserved combinations of splicing choices. Our contribution is the task design, curation and evaluation protocols, building on RIPE Atlas measurements and

Table 1: Graph and path stages of the compared variants. Trans.: Transformer on raw path features or vertex states; MLP: pair MLPs with row means. Marked EVT is query-specific. TD uses dense pair attention while retaining vertex states and vertex–pair exchanges. Task heads and ET orientations are in Appendices D.1.3 and F.2.

Variant	Graph	Path	Variant	Graph	Path
PathTransformer	None	Trans. (raw)	EVT-Path	EVT	Trans. (vertices)
TT	EVT	EVT	ET-MLP	ET	MLP
TD	EVT	EVT (dense)	ET-Twice	ET	ET
PP-MLP	EVT	MLP	MarkedEVTPath	Marked EVT	Trans. (vertices)

ThorAxe annotations, not new collection of these source data. Together, the tasks assess what finite models learn from reusable representations and when reuse saves computation.

4.1 Synthetic datasets

Synthetic data generation. For synthetic tasks, we generate weighted layered directed acyclic graphs (DAGs) with $n = 28$ vertices and four query paths of $k = 7$ vertices each. Inputs comprise $d_X = 8$ Gaussian nuisance vertex channels and $d_Y = 4$ pair channels: adjacency, positive edge weight and two Gaussian nuisance features (Appendix D.1.1). Each task uses 512 optimization, 128 validation and 256 test graphs, keeping a graph's queries together. Targets use training-only standardization and checkpoints use validation loss. Independently trained models use paired seeds: Task 1 has ten on the original dataset and three on each of two re-samplings, at 800 updates; Task 2 has five at 2,000 updates (Appendix D). Task 2 normalization uses a full 2,048-graph training pool, but optimization only its 512-graph subset. We report the coefficient of determination R^2 on held-out queries; higher is better, with zero corresponding to prediction by their mean target.

Task 1: composing selected relations. Task 1 averages products of three consecutive path-edge weights. With adjacency A, positive edge weights w and $B_{ij} = A_{ij}w_{ij}$, the target is

$$s_1(Y, P) = \frac{1}{k-3} \sum_{\ell=1}^{k-3} B_{p_\ell, p_{\ell+1}} B_{p_{\ell+1}, p_{\ell+2}} B_{p_{\ell+2}, p_{\ell+3}}. \tag{11}$$

This isolates the inductive bias for composing weights already available after selection. TT and TD retain the four-orientation EVT graph-encoder architecture, ordered selection, position codes and readout, changing the path-pair operator with a small feed-forward width compensation (Table 1 and Appendix D.2). TT improves on TD in every paired run across three independently generated graph datasets; seed counts and checkpoint-selection settings differ across datasets. A local MLP on triples of consecutive edge weights also captures much of the target variation.

Table 2: Task 1: test R^2 and paired TT–TD gains, mean ± sample standard deviation (SD) at 800 updates. Bold: best mean per dataset; –: not evaluated. TT/TD match widths and nearly match parameters (about 313,000 versus 4,481 for the local MLP); their path-pair attention costs are $O(k^3d)$/$O(k^4d)$.

Graph dataset	Seeds	TT	TD	TT–TD	Local MLP
Original	10	$\mathbf{.953 \pm .011}$	$.884 \pm .019$	$+.068 \pm .023$	–
Resampling 1	3	$\mathbf{.954 \pm .008}$	$.872 \pm .022$	$+.082 \pm .023$	$.892 \pm .023$
Resampling 2	3	$\mathbf{.912 \pm .009}$	$.862 \pm .009$	$+.050 \pm .009$	$.865 \pm .023$

Triangular path attention also helps with a dense graph encoder (DT versus DD; Table D.3); both contrasts persist at 3,200 updates (Table D.4).

Task 2: graph context, cache access and training resources. Task 2 averages $\log(1 +$ number of alternatives) over nonconsecutive path-position pairs, excluding the prescribed segment from the count. Exact dynamic programming supplies the labels (Appendix D.3). The task is motivated by variation-graph bubbles—alternative routes between shared endpoints (Garrison et al., 2018; Liao et al., 2023; Sibbesen et al., 2023). On DAGs, this target is determined by the stable selected profile: the question is how finite models exploit that information. Ten systems share the dataset manifest (the fixed graphs, queries and splits) and paired seeds, with parameter counts within 5% of TT but different widths and prediction interfaces. These include a raw-vertex PathTransformer, a graph

isomorphism network with edge features (GINE; Hu et al., 2019) followed by a path Transformer with a raw-feature bypass, chain-oriented and direction-mixed Edge Transformers, and the variants in Table 1 (Appendices B and D.1.3).

Table 3: Task 2: test R^2 over five paired seeds at 2,000 updates; brackets give the interquartile range (IQR). Bold: best overall; underlined: best reusable. PT: PathTransformer; GINE+raw: GINE-Path with raw bypass; ET-chain/ET-mix: faithful/direction-mixed Edge Transformer; Marked: MarkedEVTPath. Widths and prediction inputs differ.

Model	Median [IQR]	Mean	Model	Median [IQR]	Mean
PT	$-.013$ $[-.018, -.011]$	$-.016$	ET-Twice	$-.008$ $[-.018, -.002]$	$-.011$
GINE+raw	.190 [.184, .213]	.182	EVT-Path	.511 [.476, .572]	.538
ET-chain	$-.015$ $[-.022, -.005]$	$-.014$	PP-MLP	.537 [.455, .602]	.526
ET-mix	$-.005$ $[-.017, -.005]$	$-.010$	TT	.637 [.603, .643]	<u>.593</u>
ET-MLP	$-.006$ $[-.006, -.002]$	$-.006$	Marked	**.700** [**.664**, **.702**]	**.688**

TT has the highest median among reusable models, while MarkedEVTPath scores higher but re-encodes the graph per query (Figure 1a–b). TT leads the reusable-model budget sweep in median from 800 to 2,000 updates, although rankings vary across seeds (Appendix D.3.2). On frozen caches, broader joint $P \times V$ reading (path-to-all-vertex pair states) gives no gain over $P \times P$ selection (path-to-path pair states; mean R^2: 0.528 versus 0.530; Appendix D.3.1). ET fits training data but generalizes poorly under this protocol (Appendix D.3.2).

4.2 Atlas routing benchmark: assessing temporal prediction and graph reuse

We curate RIPE Atlas Internet-routing measurements (RIPE NCC Staff, 2015; RIPE NCC, 2026) into round-trip-time (RTT) prediction along observed responding-interface paths. Graphs contain up to 256 interfaces and paths up to 32 occurrences. A historical prior uses each interface's February–March 2026 median RTT, with that period's global median as fallback. Training uses April–July, validation August and the reserved test September. Unlabelled monthly topology is available: this is temporal prediction on known topology, not prediction of unseen network topology. We report mean absolute error (MAE) in milliseconds, averaging errors within each path and then across paths.

Table 4: Atlas September test: path-averaged MAE (ms, lower is better), mean ± SD over seeds 1–5 at validation-selected checkpoints. Bold: <u>lowest mean.</u>

System	MAE (ms)
Historical prior	12.809
PathTransformer	11.201 ± 0.089
PP-MLP	11.223 ± 0.069
TT	$\mathbf{10.940 \pm 0.102}$

PathTransformer, PP-MLP and TT predict corrections to the same prior. PathTransformer reads raw path-vertex and adjacent-edge features, including collective context, but has no learned graph encoder; PP-MLP and TT encode the whole graph (Appendix B.3). Architecture, learning-rate and checkpoint selection use 4,096 validation paths; capacities and search costs differ (Appendix E.1). We pool seeds 1–5 on the same 30,419 September paths, excluding development seed 0. Seeds 3–5 repeat the fixed configurations after the initial temporal evaluation, without further hyperparameter selection (Appendix E.2.1). TT has lower MAE in all five paired seeds, with mean gains of 0.261 ms over PathTransformer and 0.282 ms over PP-MLP (Table 4).

A separate, capacity- and optimization-matched comparison on three frozen PP-MLP caches finds no gain from joint reading: 11.050 ms versus 10.956 for selection (Appendix E.2.2). For identical trained weights and physical batches of eight, caching gives warm speedups of 2.80× for PP-MLP and 1.64× for TT at 32 paths, including cache construction (nine graph cases, tensor processing unit (TPU); prediction differences below 0.001 ms; Appendix E.2.3). Graph reuse thus saves inference time while preserving predictions; path processing remains per query. Synthetic timings likewise favor reusable TT over per-query marked encoding for multiple queries (Appendix D.4).

4.3 ThorAxe biology benchmark: assessing combinatorial splicing

ThorAxe represents protein isoforms as spliced-exon paths (Zea et al., 2021). Motivated by coordinated splicing choices (Tilgner et al., 2015; Santos-Rodriguez et al., 2026), we construct ThorAxe-Combo: do alternative blocks individually observed in a reserved clade (a group of related species)

occur together in one annotated isoform from that clade? For each gene–clade group, paths outside that clade define the visible graph. Two visible donor paths are split at shared spliced-exons; candidates recombine at least two alternative blocks. Focal positions in the first and last differing blocks are supplied to the head. Construction uses visible data, while eligibility additionally requires reserved-clade support for each chosen block and absence of the complete candidate from visible paths. The label is one if the candidate occurs in the reserved clade, zero otherwise: negatives are unobserved combinations, not established biological incompatibilities. Fixed upstream ThorAxe segmentation, visible sequence embeddings and path support supply the inputs. From 5,000 source genes, filtering retains 1,072 genes, 1,596 groups and 16,544 candidates, including 3,120 positives (Appendix F.1).

Five outer folds hold out entire genes, keeping their groups and queries together; three inner folds select learning rate and readout. Five graph systems approximately match parameters but differ in width; four offer three readouts, while GINE-Path offers vertex states only. Local references use focal/mean ESM Cambrian (ESM-C; EvolutionaryScale, 2024) embeddings or a path Transformer with/without raw junctions. They share candidates, splits and internal selection, but bypass the graph models' compressed input stem and have different capacities (Appendices F.2 and F.6). Area under the receiver operating characteristic curve (AUROC) and average precision (AP; higher is better for both metrics) weight gene–clade groups equally across outer tests; means and SDs aggregate five training seeds (Appendix F.3).

Table 5: ThorAxe-Combo: test mean ± SD over five seeds after internal learning-rate and readout selection. Graph systems (left) and local references (right) have different inputs and capacities. PT: PathTransformer; +J: raw junctions; ESM-C f+m: focal and mean embeddings. Random gives expected within-group ranking scores, averaged equally across groups.

Graph model	AUROC	AP	Reference	AUROC	AP
PP-MLP	$\mathbf{.702 \pm .018}$	$\mathbf{.676 \pm .013}$	PT	$.632 \pm .004$	$.618 \pm .003$
ET-MLP	$.690 \pm .012$	$.668 \pm .012$	PT+J	$.641 \pm .006$	$.626 \pm .003$
GINE-Path	$.643 \pm .001$	$.627 \pm .003$	ESM-C f+m	$.575 \pm .009$	$.577 \pm .005$
TD	$.638 \pm .018$	$.622 \pm .017$	Random	.500	.523
TT	$.635 \pm .023$	$.621 \pm .019$			

PP-MLP and ET-MLP reach 0.702 and 0.690 mean AUROC, ahead of TT's 0.635: triangular path processing is not uniformly advantageous. The ordered-sequence reference reaches 0.632 (0.641 with raw junctions), versus 0.575 for focal/mean embeddings. For all four pair-state models, adding a path-edge readout—the mean state of consecutive splice junctions—improves mean AUROC over the vertex-state readout (Table F.3). These independently trained comparisons jointly change pair-state access, vertex projections and selected learning rates; they do not isolate a single cause. An exploratory internal-validation pilot examines the vertex-input projection width (Appendix F.5); it is distinct from the outer-test results. The benchmark tests unseen genes within the eligible population, not unseen clades or homologous families (Appendix F.6).

5 Conclusion and Open Questions

EVT and its aligned cache separate query access from processing. Our theory identifies the selected information, proves relative uniform approximation with shared parameters across bounded graph and path sizes, and orders selection, joint reading, and marked encoding by expressivity. A second triangular stage is an inductive bias, not an information requirement. Our coherent benchmarking suite connects these distinctions to finite-resource learning: certified collisions exhibit a strict access gap, synthetic tasks show benefits of triangular composition, Atlas links predictive gains to graph reuse, and ThorAxe-Combo favors simpler pair processors and biologically aligned readouts. Rankings remain protocol-dependent: the theory allows arbitrary capacity, Atlas tests known topology in a reserved month, and biological negatives are unobserved combinations, not proven impossibilities.

Two questions follow. Can relational path processors approach the same accuracy with subcubic graph computation and smaller caches? And can task structure predict when to spend computation on triangular composition, broader cache access, or a simpler reader? Answering them would connect the information hierarchy to practical architecture selection at larger scales.

Reproducibility statement

The supplementary material includes figure-reproduction code, a computational toolbox for the models, and tools for dataset generation, retrieval and management. ThorAxe-Combo inputs are distributed through Figshare. Appendix L documents the commands, checkpoint inventories and reproduction coverage, including unavailable historical weights and curve data.

AI use statement

Generative AI tools, including ChatGPT/Codex, assisted with literature search and synthesis; research ideas and experimental design; formulating mathematical claims and developing and auditing proofs; implementing models and synthetic-data generators; experimental execution and result interpretation; and drafting and polishing the manuscript. Synthetic datasets were produced by executing the documented generators, and reported values were computed from executed experiments and archived predictions or metrics. The reproducibility materials document numerical and code checks and their scope. The authors are responsible for verifying claims, proofs, citations, code, protocols and numerical outputs, and remain solely responsible for the final content and any errors.

References

Josh Abramson et al. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630:493–500, 2024. doi: 10.1038/s41586-024-07487-w.

Waïss Azizian and Marc Lelarge. Expressive power of invariant and equivariant graph neural networks. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=lxHgXYN4bwl.

Gonzalo Benegas, Sanjit Singh Batra, and Yun S. Song. DNA language models are powerful predictors of genome-wide variant effects. *Proceedings of the National Academy of Sciences*, 120 (44):e2311219120, 2023. doi: 10.1073/pnas.2311219120.

Leon Bergen, Timothy J. O'Donnell, and Dzmitry Bahdanau. Systematic generalization with edge transformers. In *Advances in Neural Information Processing Systems*, volume 34, pp. 1390–1402, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/0a4dc6dae338c9cb08947c07581f77a2-Abstract.html.

Deng Cai and Wai Lam. Graph transformer for graph-to-sequence learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7464–7471, 2020. doi: 10.1609/aaai.v34i05.6243.

Jean-Baptiste Cordonnier and Andreas Loukas. Extrapolating paths with graph neural networks. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pp. 2187–2194, 2019. doi: 10.24963/ijcai.2019/303. URL https://doi.org/10.24963/ijcai.2019/303.

Hugo Dalla-Torre, Liam Gonzalez, Javier Mendoza-Revilla, et al. Nucleotide Transformer: Building and evaluating robust foundation models for human genomics. *Nature Methods*, 22:287–297, 2025. doi: 10.1038/s41592-024-02523-z.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*, volume 35, pp. 16344–16359, 2022. doi: 10.52202/068431-1189. URL https://papers.neurips.cc/paper_files/paper/2022/hash/67d57c32e20fd0a7a302cb81d36e40d5-Abstract-Conference.html.

Moshe Eliasof, Eldad Haber, and Eran Treister. pathGCN: Learning general graph spatial operators from paths. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 5878–5891. PMLR, 2022. URL https://proceedings.mlr.press/v162/eliasof22a.html.

EvolutionaryScale. ESM Cambrian: Revealing the mysteries of proteins with unsupervised learning. Model announcement, 2024. URL https://www.evolutionaryscale.ai/blog/esm-cambrian. Accessed 26 September 2026.

Jiarui Feng, Lecheng Kong, Hao Liu, Dacheng Tao, Fuhai Li, Muhan Zhang, and Yixin Chen. Extending the design space of graph neural networks by rethinking folklore Weisfeiler-Lehman. In *Advances in Neural Information Processing Systems*, volume 36, pp. 9029–9064, 2023. doi: 10.52202/075280-0397. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1cac8326ce3fbe79171db9754211530c-Abstract-Conference.html.

Rui Feng, Chen Luo, Qingyu Yin, Bing Yin, Tuo Zhao, and Chao Zhang. CERES: Pretraining of graph-conditioned transformer for semi-structured session data. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 219–230. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.naacl-main.16. URL https://aclanthology.org/2022.naacl-main.16/.

Erik Garrison, Jouni Sirén, Adam M. Novak, Glenn Hickey, Jordan M. Eizenga, Eric T. Dawson, William Jones, Shilpa Garg, Charles Markello, Michael F. Lin, Benedict Paten, and Richard Durbin. Variation graph toolkit improves read mapping by representing genetic variation in the reference. *Nature Biotechnology*, 36:875–879, 2018. doi: 10.1038/nbt.4227.

Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019. URL https://arxiv.org/abs/1905.12265.

Md Shamim Hussain, Mohammed J. Zaki, and Dharmashankar Subramanian. Global self-attention as a replacement for graph convolution. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 655–665, 2022. doi: 10.1145/3534678.3539296.

Md Shamim Hussain, Mohammed J. Zaki, and Dharmashankar Subramanian. Triplet interaction improves graph transformers: Accurate molecular graph learning with triplet graph transformers. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 20768–20792. PMLR, 2024. URL https://proceedings.mlr.press/v235/hussain24a.html.

John Jumper et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596:583–589, 2021. doi: 10.1038/s41586-021-03819-2.

Yeonjoon Jung and Sungsoo Ahn. Triplet edge attention for algorithmic reasoning, 2023. URL https://arxiv.org/abs/2312.05611. Second Learning on Graphs Conference (LoG 2023), Extended Abstract.

Nicolas Keriven and Gabriel Peyré. Universal invariant and equivariant graph neural networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://arxiv.org/abs/1905.04943.

Jinwoo Kim, Tien Dat Nguyen, Seonwoo Min, Sungjun Cho, Moontae Lee, Honglak Lee, and Seunghoon Hong. Pure transformers are powerful graph learners. In *Advances in Neural Information Processing Systems*, volume 35, pp. 14582–14595, 2022. doi: 10.52202/068431-1060. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/5d84236751fe6d25dc06db055a3180b0-Abstract-Conference.html.

Alexander Kozachinskiy, Felipe Urrutia, Hector Orellana, Tomasz Steifer, Germán Pizarro, Matías Fuentes, Francisco Meza Vásquez, Cristian Buc Calderon, and Cristóbal Rojas. Strassen attention, split VC dimension and compositionality in transformers. In *Advances in Neural Information Processing Systems*, volume 38, pp. 9665–9696. Curran Associates, Inc., 2025. doi: 10.52202/085713-0327. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/0df1738319f8c6e15b58cb16ea3cfa57-Abstract-Conference.html.

Wen-Wei Liao, Mobin Asri, Jana Ebler, et al. A draft human pangenome reference. *Nature*, 617:312–324, 2023. doi: 10.1038/s41586-023-05896-x.

Shuqi Lu, Zhifeng Gao, Di He, Linfeng Zhang, and Guolin Ke. Data-driven quantum chemical property prediction leveraging 3D conformations with Uni-Mol+. *Nature Communications*, 15(1): 7104, 2024. doi: 10.1038/s41467-024-51321-w.

Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019a. URL https://proceedings.neurips.cc/paper/2019/hash/bb04af0f7ecaee4aae62035497da1387-Abstract.html.

Haggai Maron, Heli Ben-Hamu, Nadav Shamir, and Yaron Lipman. Invariant and equivariant graph networks. In *International Conference on Learning Representations*, 2019b. URL https://openreview.net/forum?id=Syx72jC9tm.

Haggai Maron, Ethan Fetaya, Nimrod Segol, and Yaron Lipman. On the universality of invariant networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4363–4371. PMLR, 2019c. URL https://proceedings.mlr.press/v97/maron19a.html.

Gaspard Michel, Giannis Nikolentzos, Johannes F. Lutzeyer, and Michalis Vazirgiannis. Path neural networks: Expressive and accurate graph neural networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 24737–24755. PMLR, 2023. URL https://proceedings.mlr.press/v202/michel23a.html.

Luis Müller, Daniel Kusuma, Blai Bonet, and Christopher Morris. Towards principled graph transformers. In *Advances in Neural Information Processing Systems*, volume 37, pp. 126767–126801, 2024. doi: 10.52202/079017-4026. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/e5419147e53eba322cf12aff266a66f2-Abstract-Conference.html.

Hiroyuki Nakasora. Mutually orthogonal Latin squares and self-complementary designs. *Mathematical Journal of Okayama University*, 48:21–31, 2006. URL https://www.math.okayama-u.ac.jp/mjou/mjou48/_03_nakasora.pdf.

Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Michael Wornow, Callum Birch-Sykes, Stefano Massaroli, Aman Patel, Clayton Rabideau, Yoshua Bengio, Stefano Ermon, Christopher Ré, and Stephen Baccus. HyenaDNA: Long-range genomic sequence modeling at single nucleotide resolution. In *Advances in Neural Information Processing Systems*, volume 36, pp. 43177–43201, 2023. doi: 10.52202/075280-1872. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/86ab6927ee4ae9bde4247793c46797c7-Abstract-Conference.html.

Omri Puny, Derek Lim, Bobak Kiani, Haggai Maron, and Yaron Lipman. Equivariant polynomials for graph neural networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 28191–28222. PMLR, 2023. URL https://proceedings.mlr.press/v202/puny23a.html.

PyTorch Team at Meta. Helion: A high-level DSL for performant and portable ML kernels. PyTorch Blog, 2025. URL https://pytorch.org/blog/helion/. Accessed 26 September 2026.

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. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/5d4834a159f1547b267a05a4e2b7cf5e-Abstract-Conference.html.

RIPE NCC. RIPE Atlas: Region meshes. Official documentation and measurement archives, 2026. URL https://atlas.ripe.net/docs/tools-and-code/region-meshes. Accessed 26 September 2026.

RIPE NCC Staff. RIPE Atlas: A global internet measurement network. *The Internet Protocol Journal*, 18(3):2–26, 2015. URL https://www-static.ripe.net/static/rnd-ui/atlas/static/page/InternetProtocolJournal-18-3.pdf.

Krzysztof Rusek, José Suárez-Varela, Paul Almasan, Pere Barlet-Ros, and Albert Cabellos-Aparicio. RouteNet: Leveraging graph neural networks for network modeling and optimization in SDN. *IEEE Journal on Selected Areas in Communications*, 38(10):2260–2270, 2020. doi: 10.1109/JSAC.2020.3000405.

Gabriela Santos-Rodriguez, Akanksha Srivastava, Agin Ravindran, Favour O Oyelami, Chi Kin Ip, Pallavi Gupta, Jeanette Villanueva, Helen E King, Abigail Grootveld, Alexandra Sneddon, et al. Isoform-specific m6A deposition and coordinated splicing shape mammalian transcriptome evolution. *Nature Communications*, 17:5075, 2026. doi: 10.1038/s41467-026-72124-1.

Jonas A. Sibbesen, Jordan M. Eizenga, Adam M. Novak, Jouni Sirén, Xian Chang, Erik Garrison, and Benedict Paten. Haplotype-aware pantranscriptome analyses using spliced pangenome graphs. *Nature Methods*, 20:239–247, 2023. doi: 10.1038/s41592-022-01731-9.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/j.neucom.2023.127063.

Hagen Tilgner, Fereshteh Jahanbani, Tim Blauwkamp, Ali Moshrefi, Erich Jaeger, Feng Chen, Itamar Harel, Carlos D Bustamante, Morten Rasmussen, and Michael P Snyder. Comprehensive transcriptome analysis using synthetic long-read sequencing reveals molecular co-association of distant splicing events. *Nature Biotechnology*, 33(7):736–742, 2015. doi: 10.1038/nbt.3242.

Philippe Tillet, H. T. Kung, and David Cox. Triton: An intermediate language and compiler for tiled neural network computations. In *Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages*, pp. 10–19, 2019. doi: 10.1145/3315508.3329973.

Luyu Wang, Yujia Li, Ozlem Aslan, and Oriol Vinyals. WikiGraphs: A Wikipedia text–knowledge graph paired dataset. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing*, pp. 67–82. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.textgraphs-1.7. URL https://aclanthology.org/2021.textgraphs-1.7/.

Xiyuan Wang, Pan Li, and Muhan Zhang. Improving graph neural networks on multi-node tasks with the labeling trick. *Journal of Machine Learning Research*, 26(23):1–44, 2025. URL https://jmlr.org/papers/v26/23-0560.html.

Tuo Xu and Lei Zou. Rethinking higher-order representation learning with graph neural networks. In *Proceedings of the Second Learning on Graphs Conference*, volume 231 of *Proceedings of Machine Learning Research*, pp. 38:1–38:25. PMLR, 2024. URL https://proceedings.mlr.press/v231/xu24a.html.

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

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J. Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/1912.10077.

Diego Javier Zea, Sofya Laskina, Alexis Baudin, Hugues Richard, and Elodie Laine. Assessing conservation of alternative splicing with evolutionary splicing graphs. *Genome Research*, 31(8):1462–1473, 2021. doi: 10.1101/gr.274696.120.

Bohang Zhang, Guhao Feng, Yiheng Du, Di He, and Liwei Wang. A complete expressiveness hierarchy for subgraph GNNs via subgraph Weisfeiler-Lehman tests. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 41019–41077. PMLR, 2023. URL https://proceedings.mlr.press/v202/zhang23k.html.

Cai Zhou, Xiyuan Wang, and Muhan Zhang. From relational pooling to subgraph GNNs: A universal framework for more expressive graph neural networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 42742–42768. PMLR, 2023. URL https://proceedings.mlr.press/v202/zhou23n.html.

A NOTATION

Table A.1 collects the main symbols. Where a symbol has several uses, its context is specified explicitly. Graph vertices and path occurrences are distinct index sets, even when the path has no repeated visits.

Table A.1: Main notation for the reusable interface, its theoretical profiles, and the experiments.

Symbol	Meaning
Graph inputs and ordered paths	
$G,\ V=[n]$	Input graph and its vertex set; $[n]=\{1,\dots,n\}$.
n, N	Number of vertices in one input and the maximum number allowed in the bounded-size theoretical domain.
$P=(p_1,\dots,p_k),\ k, K$	Ordered query sequence in V^k, its number of vertex occurrences, and the maximum allowed number of positions. Each $p_s \in V$; repeated vertices are allowed.
$i, j, u;\ s, t$	Graph-vertex indices; path-occurrence indices. The vertex visited at position s is p_s. Some proofs also use $u=(X,Y,P)$ for a complete input.
$X=(x_i),\ Y=(y_{ij})$	Vertex and ordered-pair features or states. At input, $X \in \mathbb{R}^{n\times d_X}$ and $Y \in \mathbb{R}^{n\times n\times d_Y}$. Pair features include self-pairs and pairs without an edge.
$d_X, d_Y;\ q$	Input feature dimensions; output coordinates per path position in the theoretical model.
$P\times P$	Shorthand for occurrence-indexed pair-state access: $\widetilde{Y}[P,P]=(\widetilde{y}_{p_s p_t})_{s,t\in[k]}$, with k^2 positions, preserving order and repeated visits (Equation (4)).
$P\times V$	Access to $\widetilde{Y}[P,V]=(\widetilde{y}_{p_s u})_{s\in[k],\,u\in V}$, with kn positions, including graph vertices on the path. Joint reading combines all relations to the same u before invariant aggregation over vertices.
$\mathbf{1}_{\{p_s=p_t\}},\ \mathbf{1}_{\{s=t\}}$	Indicators of visiting the same graph vertex and of being the same path occurrence, respectively.
Encoded states, operators, and dimensions	
$\widetilde{X}, \widetilde{Y}$	Reusable graph states computed before the query is selected; $\widetilde{X}[P], \widetilde{Y}[P,P]$ denote their ordered selection.
$\widehat{X}^0, \widehat{Y}^0$	Selected vertex and pair states, augmented with position and repeated-vertex information.
$\phi(s)=e_s,\ \psi(s,t)$	Protected one-hot occurrence tag and ordered pair-position tag $\psi(s,t)=\phi(s)\,\|\,\phi(t)$ in the idealized ordered selector.
$X^{\mathrm{P}}, Y^{\mathrm{P}};\ Z, R$	Final path-stage vertex and pair states; Z, R are their practical prediction-head notation.
$\theta_{\mathrm{G}}, \theta_{\mathrm{P}};\ L_G, L_P$	Independent graph/path-stage parameters and numbers of macro-layers.
$d;\ d_a, d_v$	Hidden width; query/key dimension and value dimension per attention head, respectively. The symbol d has a separate color-code meaning in the decoder proof below.
$\mathrm{Att}, \mathrm{TriAtt}^{\alpha,\beta}$	Standard attention and triangular pair attention. Each $(\alpha,\beta)\in\{\rightarrow,\leftarrow\}^2$ chooses two oriented pair views, with $y^{\rightarrow}_{ij}=y_{ij}$ and $y^{\leftarrow}_{ij}=y_{ji}$.
$\mathrm{AlMLP},\ \mathrm{flat},\ \mathrm{unflat}$	Aligned vertex–pair MLP exchange; reshaping between an $m\times m$ pair array and its m^2-token list.
$\mathrm{Norm},\ \mathrm{PNorm}(h;\tau)$	Tokenwise feature normalization; normalization of learned state h while retaining fixed tags τ. See Definition H.4 for the theoretical conventions.
$\|,\ \odot,\ \{\{\cdot\}\}$	Concatenation, element-wise product, and a multiset retaining multiplicities.
Colors, information profiles, and function classes	
$c^r_{ij},\ R_N,\ c^\star_{ij}$	Bidirectionally initialized 2-FWL pair color after r rounds; common terminal depth $R_N=2N^2$; terminal color $c^\star_{ij}=c^{R_N}_{ij}$.
$C^\star_N,\ \mathfrak{C}_N$	Full matrix of terminal unmarked pair colors; their common formal color space across graph comparisons.

Continued on the next page

Table A.1 (continued)

Symbol	Meaning
$\Psi_{N,K}$	Selected profile $\big(n, k, (c^\star_{p_s p_t})_{s,t\in[k]}\big)$, retaining the ordered path-pair color matrix.
$\Gamma_{N,K}$	Joint-read profile $\big(n, k, \{\{(c^\star_{p_1 u}, \dots, c^\star_{p_k u}) : u \in V\}\}\big)$. Each tuple combines relations to the same vertex u.
$m_P(i),\ \bar{c}^\star_{ij}$	Occurrence mark $m_P(i) = \sum_{s:p_s=i} e_s$ assigned before graph encoding; terminal pair color after marked refinement.
$\Psi^{\mathrm{mark}}_{N,K}$	Marked profile $\big(n, k, (\bar{c}^\star_{p_s p_s})_{s\in[k]}\big)$, using the ordered selected marked diagonals.
$\mathcal{D}_{n,k},\ \mathcal{D}_{\le N,\le K}$	Input triples with fixed sizes (n, k), and their disjoint union over $n \le N,\ k \le K$.
$\mathcal{O}_K,\ S$	Output space $\bigsqcup_{k=1}^{K} \mathbb{R}^{k\times q}$; a finite or compact input set, as specified in each result.
$\ker f,\ \mathrm{Ker}(\mathcal{F})$	Pairs of inputs receiving the same output under f; common indistinguishability relation $\bigcap_{f\in\mathcal{F}} \ker f$ for a function family.
$\mathcal{H}_{N,K},\ \mathcal{H}^{\mathrm{dec}}_{N,K}$	Two-EVT family of Definition H.4; its subfamily with all path triangular increments disabled.
$\mathcal{C}_\Psi(S),\ \|\cdot\|_{\infty,S}$	Continuous, length-preserving targets constant on profile classes; uniform output error on S, using the Frobenius norm within each output component.
$\eta(c) \in \mathbb{R}^d,\ U_{st},\ B$	Color code of dimension d, explicit $K^2(1+d)$-coordinate auxiliary vector, and its globally broadcast mean in Equations (88) and (89). This d is chosen for the proof's color codes and is not fixed by an experimental hidden width.
Experimental notation and contextual uses	
A, w, B	Adjacency, positive edge weights, and weighted adjacency $B_{ij} = A_{ij}w_{ij}$ in the synthetic tasks.
$Q;\ B$	In cost statements, number of query paths sharing a graph cache (Section 2); physical batch size.
D, A, N; t	Diagonal, adjacent, and non-adjacent relation colors in the Latin-graph experiment; common-neighbor count $t \in \{0, 1\}$ for the three query anchors.
S, U, B, T	Selected pair colors, separate row histograms, separate paired-row histograms, and the joint three-row histogram in Appendix C.
(n, d, λ, μ)	Strongly regular graph parameters: vertex count, degree, and common-neighbor counts for adjacent and distinct non-adjacent vertices. Here d denotes degree.
g_R	Mean consecutive-pair state $(k-1)^{-1}\sum_{s=1}^{k-1} R_{s,s+1}$ used by biological path-edge readouts, in addition to focal vertex states and path context.
TT, TD, DT, DD	In the factorial comparison, the first letter gives the graph pair operator and the second the path pair operator: T is triangular attention and D is dense pair self-attention.

B Inputs and Access in the Evaluated Models

The tables separate the raw observations given to each model from the encoded states read by its path processor and prediction head. Selected states can already contain off-path information: PP-MLP, for example, applies its MLPs after a learned graph encoder. Here $X[P]$ denotes raw path-vertex features; $\widetilde{X}[P]$ and $\widetilde{Y}[P, P]$ denote states selected from the graph cache. Z, R denote vertex and pair outputs of the path processor, when present. All means exclude padding. Positions refer to occurrence indices; repeated-vertex indicators refer to equality of the underlying graph vertices.

B.1 SYNTHETIC COMPARISONS AND CERTIFIED READERS

System	Graph observations	Query access	Query information
PathTransformer (Task 2)	None beyond $X[P]$	Raw path vertices	Ordered positions
GINE-Path (Task 2)	X; directed edges and their attributes	Selected $\widetilde{X}[P]$ and raw $X[P]$ bypass	Ordered positions
Edge Transformer, faithful / direction-mixed	X, Y via endpoint lifts	Self-pair proxies, then path Transformer	Ordered positions
EVT-Path	X, Y	Selected $\widetilde{X}[P]$	Ordered positions
PP-MLP, ET-MLP	X, Y	Complete $P \times P$; pair MLPs and row means	Positions, vertex equality
TT, TD, DT, DD; ET-Twice	X, Y	Complete $P \times P$; second pair processor	Positions, vertex equality
MarkedEVTPath	X, Y and path-position marks	Query-conditioned graph states, then path Transformer	Marks before encoding; positions
Local MLP (Task 1 control)	No graph encoding	Windows of three raw path-edge weights	Window order

Table B.1: Synthetic input access. Graph encoders can use off-path vertices and relations; the first and last rows have no such access. No historical timing prior, biological clade indicator, or extra biological focal marks is supplied. Models grouped in one row share access rules; their architectures and training settings are specified in the task sections.

In Task 1, the TT–TD intervention retains the same graph-encoder architecture and selected observations, with separately trained weights and a small FFN adjustment. The factorial controls vary both operators. Canonical Task 2 rankings instead compare complete systems with different widths and readout interfaces (Appendix D.1.3).

The Task 2 frozen-cache readers both retain the complete ordered selected array. Their added branch combines $(\widetilde{y}_{p_s u})_{s\in[k]}$ for each common graph vertex u before summation. The selection control restricts u to distinct vertices visited by the path; the joint reader uses all valid graph vertices. Both receive graph and path counts. Their equal-sized heads are trained from scratch on the same frozen cache within each seed. They are distinct from the canonical PP-MLP head.

The certified collision readers use constant vertex features and binary adjacency, a frozen random EVT cache, and ordered anchor relations. Their pairwise and joint aggregations retain different numbers of anchor relations before pooling over graph vertices. Exact color profiles certify the examples and are not network inputs. These readers differ from the Task 2 readers and have different parameter counts (Appendix C).

B.2 THORAXE-COMBO

System	Observations	Learned graph context	Final prediction inputs
PP-MLP, ET-MLP, TD, TT	964 vertex channels; seven pair channels	All visible vertices and pairs	Focal Z, mean Z, clade; optional focal-pair or path-edge R
GINE-Path	Same vertex/pair channels; preserved directed adjacency	Directed message passing on visible edges	Focal Z, mean Z, clade; no R branch
ESM-C focal + mean	960 sequence channels at path vertices	No learned graph encoder	Focal embeddings, path mean, clade
PathTransformer, with/without raw junctions	964 path-vertex channels; optional seven adjacent pair channels	Path attention only	Focal Z, mean Z, clade; optional mean of locally projected junctions

Table B.2: Inputs for the retained biological comparisons. Focal positions are supplied to the head; the graph encoder remains unmarked. Graph-based systems use the 8/4-channel stem; local references bypass it. Readout choices and widths are specified in Appendix F.2.

All graph models receive visible-species and visible-path support attributes through their pair input. GINE uses those attributes on present edges, while preserving the discrete adjacency channel through its stem. Its vertex-only final head therefore still contains learned graph context. The ESM-C reference uses embeddings averaged over visible instances and species during preprocessing. All systems receive the same focal indices and clade identity, and predict directly, without an additive historical prior. Within each pair-state graph-model family, the three readout conditions are trained independently from input projections to prediction head.

B.3 ATLAS PROFILE MODELS

For Atlas, PathTransformer receives raw vertex features and locally transformed adjacent junction attributes. PP-MLP and TT receive the same graph observations and encode the whole graph before selection. At vertex-level targets, the final head reads an occurrence state and its normalized ordinal coordinate. These models differ from the synthetic and biological implementations carrying similar model names.

Every profile model predicts a correction to a shared context prior:

$$\widehat{t}_{qs} = b_{qs} + \alpha f_\theta(G, P_q)_s, \qquad \alpha = 10 \text{ ms}.$$

The prior b is read from the historical graph's vertex statistics; it is also represented among the raw features. Atlas uses historical RTTs and observed snapshot topology. Thus PathTransformer has no learned graph encoder but still receives collective context and the prior. Missing labels and padding are masked. There are no biological focal marks or clade embeddings in these adapters. The main model comparison evaluates complete, independently trained systems selected on validation (Appendix E). The matched-reader control instead holds PP graph states fixed and varies which vertices its context branch can read, with identical inputs, trainable capacity, and selection rules (Appendix E.2.2).

C CERTIFIED CACHE SELECTION AND AGGREGATION

Latin-square graphs provide queries with identical selected 2-FWL profiles and different labels. Their structural diversity lets us test the readouts on new, non-isomorphic graphs.

Data and exact summaries. We use graphs constructed from Latin squares of order seven (Nakasora, 2006). Each graph is a strongly regular graph with parameters $(n, d, \lambda, \mu) = (49, 18, 7, 6)$: it has 49 vertices, every vertex has 18 neighbors, adjacent vertices have exactly 7 common neighbors, and distinct non-adjacent vertices have exactly 6 common neighbors. Vertices have constant scalar features, and binary adjacency is the only pair attribute. For an ordered triangle $P = (p_1, p_2, p_3)$, the target t is the number of vertices adjacent to all three anchors:

$$t = |N(p_1) \cap N(p_2) \cap N(p_3)|,$$

where $N(v)$ denotes the set of neighbors of vertex v. We retain 32 queries per graph, balanced between 16 triangle paths with $t = 1$ and 16 with $t = 0$. The confirmation cohort contains 20 new test graphs and 640 queries. The earlier 48-graph cohort is used for development; only eight queries from one of its graphs are used to fit the readouts.

Joint unmarked 2-FWL refinement over all 68 development and confirmation graphs yields a stable partition into three common color classes: diagonal D, adjacent A, and non-adjacent N. For each graph, write $c^\star_{uv} \in \{\mathrm{D}, \mathrm{A}, \mathrm{N}\}$ for the resulting stable color of pair (u, v).

We compare four ways of retaining the colors associated with a query P. S keeps the selected pair colors between its three anchors. U counts colors separately for each anchor over all graph vertices. B counts pairs of colors that share the same graph vertex, separately for each pair of anchors. T counts joint color triples for all three anchors before discarding that shared vertex index:

$$\begin{aligned}
\mathsf{S}(P) &= (c^\star_{p_i p_j})_{i,j=1}^{3}, \\
\mathsf{U}(P) &= \left(\left\{\left\{c^\star_{p_i u} : u \in V\right\}\right\}\right)_{i=1}^{3}, \\
\mathsf{B}(P) &= \left(\left\{\left\{(c^\star_{p_i u}, c^\star_{p_j u}) : u \in V\right\}\right\}\right)_{i<j}, \\
\mathsf{T}(P) &= \left\{\left\{(c^\star_{p_1 u}, c^\star_{p_2 u}, c^\star_{p_3 u}) : u \in V\right\}\right\}.
\end{aligned}$$

Detailed proof of Proposition 6. The main-text proposition applies separately within every graph in these balanced cohorts. In the notation above, the optimal deterministic binary classification accuracies for functions of $\mathsf{S}, \mathsf{U}, \mathsf{B}, \mathsf{T}$ are respectively $1/2, 1/2, 1/2, 1$.

Proof. The degree and common-neighbor counts make every refinement round depend only on whether a pair is diagonal, adjacent, or nonadjacent. Thus the stable colors are precisely the three classes $\mathrm{D}, \mathrm{A}, \mathrm{N}$ used above.

For every query triangle, the selected 3×3 block has D on the diagonal and A off the diagonal, so S is constant.

Regularity gives the same row histogram for every anchor p_i: the row $(c^{\star}_{p_i u})_{u \in V}$ has multiplicities $(\mathrm{D}:1, \mathrm{A}:18, \mathrm{N}:30)$, so U is constant.

The strongly regular intersection counts give the same pairwise histogram for every adjacent anchor pair. Indeed, for adjacent anchors, DA and AD each occur once, AA occurs $\lambda = 7$ times, AN and NA each occur $d - 1 - \lambda = 10$ times, and NN occurs $n - 2d + \lambda = 20$ times. Thus B is constant. The pairwise counts and the three-anchor counts derived below are summarized here:

Two-row pattern	DA	AD	AA	AN	NA	NN
Multiplicity	1	1	7	10	10	20

Three-row pattern	Each multiplicity
DAA, ADA, AAD	1
AAA	t
AAN, ANA, NAA	$6 - t$
ANN, NAN, NNA	$4 + t$
NNN	$16 - t$

For three anchors, DAA, ADA, and AAD each occur once, while AAA occurs t times. Each pair has λ common neighbors, including the third anchor and the t neighbors shared by all three. Thus AAN, ANA, NAA each occur $\lambda - 1 - t = 6 - t$ times. The one-adjacency and zero-adjacency counts are respectively $d - 2\lambda + t = 4 + t$ and $n - 3d + 3\lambda - t = 16 - t$. Thus T distinguishes the two labels because its AAA count recovers t, whereas S, U, and B are constant across the retained queries. Since labels are balanced within every graph, any deterministic classifier based on these summaries has accuracy $1/2$. In contrast, T gives optimal accuracy 1. □

The bound applies to each of the three summaries that discard joint three-anchor information. Joint aggregation can use binary operations, provided they retain the shared vertex index. For example, one may first compute $r(u) = A_{p_1 u} A_{p_2 u}$ and then $t = \sum_u r(u) A_{p_3 u}$. The limitation therefore concerns aggregation schemes that discard the shared index u before the information from all three anchors has been combined.

The full set of 1,029 triangles in each graph is not assumed to be balanced, so its majority-class baseline need not be $1/2$.

For the reported experiment, an independent script recomputed targets from adjacency and compared complete tuple signatures exactly. These signatures certify the dataset; the networks receive only the graph features described above.

Learned readouts of a frozen cache. For each seed, we randomly initialize an EVT encoder with two blocks, width 32, feed-forward network (FFN) width 64, four direction-balanced heads, LayerNorm, and zero dropout. Its weights remain fixed, and both readouts use the same cached pair states $\widetilde{Y}$. Training only the readouts isolates their aggregation schemes on identical representations.

The joint readout combines the relations of all three anchors to each vertex u, then sums over vertices. The pairwise readout first combines the relations of two anchors to each vertex, sums separately for

each anchor pair, and finally combines the three summaries. In terms of the cached states,

$$\ell_{\mathrm{joint}}(P) = \beta + \sum_{u \in V} \phi([\widetilde{y}_{p_1 u}; \widetilde{y}_{p_2 u}; \widetilde{y}_{p_3 u}]),$$
$$h_{ij} = \sum_{u \in V} f([\widetilde{y}_{p_i u}; \widetilde{y}_{p_j u}]), \qquad (i,j) \in \{(1,2),(1,3),(2,3)\},$$
$$\ell_{\mathrm{pair}}(P) = g([h_{12}; h_{13}; h_{23}]).$$

Here $[\cdot;\cdot]$ denotes concatenation, β is a learned scalar bias, and ϕ, f, and g are MLPs with rectified linear unit (ReLU) hidden activations. The logits ℓ are converted to class probabilities by a sigmoid.

Since each pair state has 32 coordinates, ϕ has widths $96 \to 64 \to 32 \to 1$, f has widths $64 \to 64 \to 32$, and g has widths $96 \to 32 \to 1$. The joint and pairwise readouts contain 8,322 and 9,377 parameters, respectively; neither parameter counts nor operation counts are matched. Final affine outputs are initialized at zero. Both readouts access the same three full rows of pair states.

Only eight fixed training queries from a single graph are used for optimization. Each step uses all eight queries and a graph permutation transporting the frozen cache and anchors. Paired readouts share the permutation stream. Training uses AdamW for 1,000 steps with learning rate 10^{-3}, zero weight decay, and gradient clipping at 1. The six final readout checkpoints and three encoders are fixed before generating the confirmation cohort, which is used only for testing.

Fresh confirmation protocol. We record the protocol and hashes of the code and weights before generation. Independent randomized row matchings generate reduced Latin squares, using a fixed seed and a limit of 10,000 attempts. We retain the first 20 candidates that supply at least 16 queries of each label and have distinct structural invariants from all previously retained or excluded graphs. The exclusion set contains 107 structures from the earlier manifests, candidate pool, and reconstructed rejected generation attempts. Generation stops after 365 attempts; sampling is non-uniform and conditioned on structural novelty and class coverage, with no model-dependent filtering. Exact structural signatures certify non-isomorphism, and a common color identification verifies the profiles and labels before inference. All queries and vertex permutations of the new graphs are used only for testing.

Results and interpretation. Across three seeds, the joint readout achieves 100% accuracy on each of the 20 new graphs, while pairwise pooling remains at its certified 50% ceiling.

Table C.1: Accuracy (%) on 640 queries from 20 new graphs, for three shared encoder seeds.

Seed	Pairwise pooling	Joint three-row
17	50	100
29	50	100
43	50	100

Re-encoding each test graph under three vertex permutations preserves both accuracies. Raw accuracy at a zero-logit threshold and accuracy assigning half credit when $|\ell| \le 10^{-6}$ agree here; all pairwise-readout logits lie within this tolerance. The results show that jointly reading $P \times V$ relations from frozen EVT representations recovers information unavailable through $P \times P$ selection. As the multiplicity formulas above show, the joint histogram T depends only on t: each label has the same joint relation histogram across development and confirmation graphs. The confirmation therefore tests the same information-access contrast on new, non-isomorphic structures of the same size. The standalone supplement retains the confirmation logits and labels in `figures/data/cache-access-predictions.csv.gz`, with provenance digests in `figures/data/provenance.json`. The command `python reproduce.py figures` recomputes the canonical-query accuracies in `outputs/figures/cache-access-fresh.csv`; this is numerical reconstruction from recorded predictions, not new checkpoint inference. The reported generation, certification, and frozen-readout protocol above is not supplied as a complete standalone experiment driver. Its trained weights and graph arrays are not bundled, and reconstructing the accuracy table does not rerun the structural certificates or recover those artifacts (Appendix L.1).

D SYNTHETIC EXPERIMENTS

We first give the shared experimental setup, then collect each task's definition, results, and controls: Task 1 tests path composition, and Task 2 predicts path alternatives. The cache timing study follows these two tasks.

D.1 SHARED EXPERIMENTAL SETUP

This section specifies the synthetic inputs, shared training protocol, model definitions, and parameter accounting. Each task section then gives its target, results, and controls together. The standalone supplement exposes these models through `models/synthetic.py`, with dataset generation and loading in `datasets/synthetic.py`; see Appendices L.3 and L.4. The scientific protocols below describe the reported experiments, while Appendix L.1 distinguishes reconstructing their numerical summaries from performing new fits.

D.1.1 GRAPH SAMPLING AND INPUT CHANNELS

The generator has seven layers of four vertices. Edges can join only consecutive layers, in the forward direction. Each candidate edge is initially drawn independently with probability 0.40. Within each consecutive-layer block, every empty outgoing row is repaired by adding an edge to a uniformly chosen vertex in the next layer; every remaining empty incoming column is then repaired with a uniformly chosen source. Proposals whose undirected skeleton is disconnected are rejected. Thus 0.40 is the initial sampling probability, not the final edge density.

Before task-specific query selection, all eight vertex channels are drawn independently from $\mathcal{N}(0,1)$. They are nuisance features, with no layer labels or target-derived statistics. For every ordered pair, the four input channels, in order, are

$$y_{ij} = \big(A_{ij},\ A_{ij}w_{ij},\ A_{ij}\eta_{ij}^{(1)},\ A_{ij}\eta_{ij}^{(2)}\big), \qquad w_{ij} = \log(1 + e^{g_{ij}}) + 0.01,$$

where $g_{ij}, \eta_{ij}^{(1)}, \eta_{ij}^{(2)}$ are independent standard normal draws, independent of the vertex-feature draws and initial adjacency. Thus the second channel is the positive edge weight $B_{ij} = A_{ij}w_{ij}$; the last two are Gaussian edge nuisance features. All four channels vanish on nonedges, including self-pairs. These input arrays are shared by the queries of a graph; position codes and query marks are added by the model interfaces, not included in these channels.

For Task 1 and Task 2, four distinct full-length paths are sampled by random walks: the starting vertex is uniform in the first layer, and each next vertex is uniform among the current vertex's successors. Duplicate paths are rejected; this is not uniform sampling over all full paths. Finally, a seed-determined random permutation is applied consistently to vertices, features and queries. Both tasks use noiseless labels computed from the resulting graph and path.

D.1.2 DATA AND OPTIMIZATION

Unless a task-specific control states otherwise, the predictive studies use AdamW, a cosine learning-rate schedule from 3×10^{-4} to 0.05 of that value, weight decay 10^{-4}, gradient clipping at 5, training batch size 64, evaluation batch size 256, minimum validation improvement 10^{-4}, and validation-only checkpoint selection. Additional Task 1 and frozen-cache Task 2 controls specify their settings in Appendices D.2 and D.3.1. Target standardization uses training labels only. Each graph supplies four queries of fixed length $k = 7$, using the sampler above. The optimization, validation, and test sets contain 512, 128, and 256 graphs, respectively. Task 2 uses a larger normalization pool, detailed in Appendix D.3. The settings that differ by study are listed below; no test metric enters stopping or model selection.

Table D.1: Optimization protocol for the original Task 1 dataset and Task 2. "Patience" counts validation evaluations after the minimum-update threshold. MSE denotes mean squared error.

Setting	Task 1	Task 2
Optimization seeds	10	5
Maximum updates	800	2000
Minimum updates	400	2000
Warm-up updates	80	80
Validation interval	40	100
Patience	10	100
Objective	scalar MSE	scalar MSE

Every attention block uses four heads, linear input projections, two graph or pair layers, and two path/occurrence layers where that stream exists. The practical EVT implementation uses LayerNorm and separate pair initialization; TT adds absolute endpoint one-hot codes and a scalar relative offset in its second stage. The endpoint-aware rotary position embedding (RoPE) construction in Appendix J.2 is a theoretical extension and is not used by these checkpoints. Capacity matching changes only the hidden/FFN width: PathTransformer 112, GINE-Path 104, the two single-pass Edge Transformer path models 80, ET-MLP 88, ET-Twice 72, EVT-Path and MarkedEVTPath 40, PP-MLP 44, and TT 32.

D.1.3 Models and parameter matching

PathTransformer uses only the ordered path. GINE-Path applies a standard message-passing graph encoder before the same path stack; the main-table variant concatenates the raw selected-vertex features to the GINE embeddings before path projection, so message passing cannot irreversibly erase the raw path signal. The faithful Edge Transformer uses endpoint-lifted pair tokens and classical chain-oriented triangular attention; its direction-mixed counterpart allocates heads across all four contractions in Equation (98). EVT-Path reads the path from one graph EVT. PP-MLP and TT use the same complete $P \times P$ selection rule and positional fields, with separately trained graph encoders; PP-MLP replaces the second EVT by pointwise pair MLPs and masked row means. MarkedEVTPath injects the ordered path before global EVT refinement and is therefore strictly path-conditioned and non-cacheable. Two pair-only controls complete the corresponding two-by-two comparison: ET-MLP applies a balanced global Edge Transformer, the same complete ordered selection, and pointwise pair MLPs with masked row means; ET-Twice instead applies a second balanced Edge Transformer to the selected pair array. Thus ET-MLP is the pair-only analogue of PP-MLP, while ET-Twice is the pair-only analogue of TT; both retain ReLU in their ET blocks. ET-MLP uses GELU in the pair and vertex MLPs after selection. Its balanced, width-88 encoder differs from the chain-only, width-80 biological ET-MLP configuration (Appendix F.2). TT uses the balanced allocation, with one of its four triangular heads assigned to each contraction in Equation (98). Widths and feed-forward dimensions are predeclared. The ten models in the canonical Task 2 comparison are within 5% of TT's parameter count. The matched Task 1 intervention and its small FFN compensation are detailed in Appendix D.2.

Task 2 comparison. All ten systems share one exact manifest and paired optimization seeds under the protocol above. Prediction scores are reported in Table 3; Table D.2 gives the corresponding parameter and resource measurements. Interpretations and task-specific controls follow in Appendix D.3.

Standalone software and verification. The numerical supplement is self-contained; paths and commands in these experimental appendices are relative to its extracted root. Installation and coverage are described in Appendices L and L.1. `experiments/synthetic/recipes.json` records the supported Task 1 and Task 2 training configurations, and `datasets/resources/synthetic/catalog.json` records input-generation recipes and content fingerprints. Synthetic graph, query, target, and split arrays are generated locally rather than bundled. The controlled Task 2 recipe preserves its complete normalization pool and fixed optimization subset. No trained model weights are distributed; historical checkpoint inventories under `experiments/resources/` are metadata, not executable checkpoints.

`figures/data/` contains compact historical measurements, numerical references, and provenance digests. From the supplement root, `python reproduce.py verify` checks the distributed file

Table D.2: Parameter counts and computational measurements for Task 2. Time and resident set size (RSS) are isolated CPU training-step measurements; floating-point operation counts (FLOPs) are per query, in millions (MFLOPs). Fwd/bwd denote forward/backward passes. Prediction scores are in Table 3.

Model	Parameters	Fwd MFLOPs/q	Fwd+bwd MFLOPs/q	ms/update	RSS (MiB)
PathTransformer	312,795	**4.27**	**12.75**	**11.0**	**404**
GINE-Path + raw bypass	313,581	4.56	13.34	20.7	574
ET faithful	312,717	66.36	198.41	366.7	1372
ET direction-mixed	312,717	66.36	198.41	370.3	1358
ET-MLP	313,247	94.13	281.67	382.7	1391
ET-Twice	312,869	75.28	225.24	394.9	1373
EVT-Path	312,531	65.10	172.15	304.1	1413
PP-MLP	313,687	71.25	213.68	343.6	1435
TT (balanced)	312,753	48.00	140.69	313.0	1390
MarkedEVTPath	312,811	258.54	682.95	1076.7	2695

manifest; `python reproduce.py figures` reconstructs the available numerical tables and plots under `outputs/figures/`. These commands do not fit models or run checkpoint inference. In particular, summary reconstruction does not regenerate every historical prediction or certificate. Task 1 records retain the CPU comparisons on three graph datasets and the factorial and longer-budget float32 Metal Performance Shaders (MPS) controls. The original Task 2 per-update histories are unavailable; the corresponding historical optimization plot cannot be reconstructed from final scores. Fresh training is a separate operation (Appendix L.5), produces new checkpoints and histories, and may differ numerically across software versions and hardware platforms.

D.2 Task 1: mechanism, graph-operator, and training-budget controls

Task 1 tests how models compose selected path weights. Its target (Equation (11)) averages four products of three consecutive edge weights on a seven-occurrence path. All required weights are available after selection.

The three graph datasets in Table 2 use CPU fits; the factorial and longer-budget controls below use separate float32 MPS fits.

Matched intervention. TT and its dense-path control both use width 32, four heads, two graph and two path layers, separate pair initialization, pre-LayerNorm and zero dropout. Graph encoder, ordered selection, absolute and relative positions, cross-attention structure and scalar readout are retained. Replacing triangular attention removes one value projection per path layer; the dense control widens the path FFN from 128 to 130 in compensation, giving 312,457 parameters against 312,753. The comparison thus changes the attention operator together with a small FFN adjustment.

On the original graph dataset, the two models use seeds 0–9, the same manifest and sampling order, and 800 completed updates under the shared protocol (Appendix D.1.2). All ten paired gains are positive (two-sided sign-test $p = 0.002$). Table 2 reports means and sample standard deviations across seeds within each dataset; gain statistics are computed from the paired differences.

Factorial control. The graph and path pair operators are varied independently inside the same coupled vertex–pair architecture. We write T for balanced triangular attention and D for dense self-attention over all pair tokens; the first letter specifies the graph stage and the second the path stage. All four TT/TD/DT/DD variants retain the matched intervention's width, depth, positions, and readout architecture; each dense stage uses the same FFN compensation. D retains the EVT vertex stream and cross-attentions, which distinguishes it from the pair-only Edge Transformer baseline. The two additional PP-MLP controls, PP_T and PP_D, use the corresponding graph operator followed by pointwise pair MLPs and row means. They use width 44 to match total capacity, with graph FFN width 176 or 179, respectively. These controls match total parameter counts by using wider states than the attention variants.

These runs use the same manifest and optimizer settings as the CPU intervention, but evaluation batches contain 64 examples instead of 256. Mean paired differences are $+0.068$ for TT–TD and $+0.097$ for DT–DD, positive on all ten seeds: the path stage carries a benefit under either graph operator. TT and DT have similar means (difference $+0.004$).

Table D.3: Task 1 factorial control: test R^2 over seeds 0–9 at 800 updates and common learning rate 3×10^{-4}. These are float32 MPS fits; Table 2 reports the CPU comparisons.

Variant	Graph	Path	Width	Parameters	Mean R^2	IQR
TT	Triangular	Triangular	32	312,753	**.952**	[0.944,0.960]
TD	Triangular	Dense	32	312,457	.884	[0.867,0.898]
DT	Dense	Triangular	32	312,457	.948	[0.931,0.961]
DD	Dense	Dense	32	312,161	.851	[0.842,0.863]
PP_T	Triangular	MLPs / row means	44	313,687	.750	[0.733,0.761]
PP_D	Dense	MLPs / row means	44	313,459	.573	[0.609,0.720]

Learning rates $\{5 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}\}$ were screened on validation with seed 0 alone, with a smaller-rate tie-break and no test-based selection; all six variants selected the predeclared common rate 3×10^{-4}. The ten-seed summary includes the screening seed.

Fourfold training-budget control. The completed longer-budget check uses seeds $1, 2, 3$, the same manifest, and the common learning rate 3×10^{-4}, without reselection. All models are initialized afresh and trained for exactly 3200 updates. The cosine schedule is stretched to the new horizon; warm-up remains 80 updates and validation is still evaluated every 40 updates. The intervention therefore changes both update count and learning-rate schedule.

Table D.4: Task 1 budget sensitivity on seeds 1–3 in both columns. Mean test R^2 from validation-selected checkpoints. The 800-update column uses the corresponding three-seed subset of Table D.3. Bold marks the best score at each budget.

Variant	800 updates	3200 updates	Change
TT	.956	**.984**	+.028
TD	.870	.895	+.025
DT	**.961**	.982	+.021
DD	.852	.855	+.003
PP_T	.727	.873	+.146
PP_D	.538	.847	+.309

The two contrasts behave differently under a longer budget. The path-operator contrast persists: TT–TD moves from $+0.086$ to $+0.089$ and DT–DD from $+0.109$ to $+0.127$ on these same three seeds, positive throughout. The gaps to the pointwise controls shrink: TT–PP_T falls from $+0.229$ to $+0.112$, while DD–PP_D almost closes, from $+0.313$ to $+0.008$. The longer-budget result rests on three seeds; three positive signs give a two-sided sign test at $p = 0.25$.

Resources and statistical scope. Each comparison approximately matches parameter counts and uses equal update counts. FLOPs, wall time, and peak memory can differ. At fixed width d, dense attention on k^2 occurrence-pair tokens forms $O(k^4 d)$ attention interactions, versus $O(k^3 d)$ for triangular attention. Both operators also use $O(k^2 d^2)$ projections. The corresponding graph-stage terms are $O(n^4 d)$ and $O(n^3 d)$.

The TT–TD prediction gain therefore accompanies a lower asymptotic pair-attention cost. At the experimental $n = 28, k = 7, d = 32$, the forward operation counts in Table D.5 are also lower for TT: by 4.3% for the complete path encoder and 0.9% for the full model with four queries per graph. The shared graph encoder dominates the total.

Table D.5: Task 1 TT–TD forward operation counts in MFLOPs per query. The path column includes the two path-encoder layers and input projections. Totals include graph encoding and the scalar head, with Q queries sharing each graph. Parameters and FFN widths match the Task 1 intervention.

Model	Path encoder	Total, $Q = 1$	Total, $Q = 4$
TT	**9.641**	**163.052**	**47.995**
TD	10.079	163.490	48.433

Counts use PyTorch's `FlopCounterMode` for linear projections and matrix products, with two FLOPs per multiply–add. They exclude softmax, normalization, elementwise operations, data movement, and backward computation; these counts measure arithmetic work. Random weights suffice for this shape-based count. The standalone supplement retains the recorded counts in `figures/data/measurements.json`; `python reproduce.py figures` converts

them to the MFLOP values in `outputs/figures/t1-compute.csv` without training. This reconstructs the recorded arithmetic measurements; it does not run a new profiler.

The factorial and longer-budget comparisons share the original graph/query manifest and fixed absolute and relative position codes. The following control varies the sampled graphs while retaining these position codes.

Two additional graph generations and a local control. The two resamplings in Table 2 use the same weighted-DAG generator, each with 512 training, 128 validation, and 256 test graphs and four queries per graph. Graph fingerprints have no overlap within or between these datasets or with the original manifest. These are new samples of the same distribution. TT and TD retain the matched architectures above. A local MLP applies a shared $3 \to 64 \to 64 \to 1$ network with Gaussian error linear unit (GELU) activations to each window of three consecutive raw path-edge weights and averages the four outputs. It receives no products or off-path states. Its windowing and averaging are informed by the target; its 4,481 parameters are not matched to TT's 312,753 or TD's 312,457.

On each resampling, all three models use seeds 0–2 and exactly 800 CPU updates at initial learning rate 3×10^{-4}, with 80 warm-up updates and cosine decay to 5% of that rate. Other optimizer settings follow Appendix D.1.2. This control uses one CPU thread, evaluation batches of 32, validation every 40 updates, and selection on any strict validation-MSE improvement (threshold zero). Each dataset's training labels determine its target standardization. The original comparison instead uses evaluation batches of 256 and a validation-improvement threshold of 10^{-4}. We summarize the datasets separately to retain their variation and avoid giving the ten-seed dataset more weight than either three-seed resampling. The local MLP was evaluated only on the resamplings.

TT exceeds TD on every paired seed in both datasets. The local MLP also captures much of the target variation, showing that Task 1 can be learned with a small model given the appropriate local decomposition. The TT–TD result shows a benefit of triangular composition within the matched architecture and training budget, while the local MLP offers another effective solution. All these comparisons use the fixed position codes described above. The standalone supplement reconstructs the original, resampling, factorial, and longer-budget summaries from `figures/data/measurements.json`; their outputs are `t1-main.csv`, `t1-factorial.csv`, and `t1-budget.csv` under `outputs/figures/`. These are aggregates of historical measurements, not checkpoint replay.

Fresh Task 1 fits. After installing the supplement (Appendix L.2), the following commands run the supported studies from its root. Each output directory must be new.

```
python reproduce.py synthetic --task T1 \
  --output outputs/t1-original
python reproduce.py synthetic --task T1 --replica 1 \
  --output outputs/t1-replica1
python reproduce.py synthetic --task T1 --replica 2 \
  --output outputs/t1-replica2
python reproduce.py synthetic --task T1 --study factorial \
  --device mps --output outputs/t1-factorial
python reproduce.py synthetic --task T1 \
  --study long-budget --device mps \
  --output outputs/t1-long-budget
```

The commands generate missing graph arrays from the fixed recipes, retain the different evaluation and checkpoint-selection rules, and save new validation-selected weights, predictions, and optimization histories. The factorial and longer-budget recipes use the predeclared learning rate; they do not rerun the separate seed-0 learning-rate screen. CPU and CUDA are also supported through `--device`, but do not reproduce the original MPS arithmetic exactly. Add `--inspect` to print a recipe without training, or `--smoke` for a one-update integration check that is not a benchmark result. Full options are documented in `experiments/synthetic/README.md`.

D.3 TASK 2: PATH ALTERNATIVES AND REUSE

Target. Let $N_G(u, v)$ count directed paths in the DAG. Since the prescribed segment is one such path, $N_G(p_s, p_t) - 1$ counts alternatives. Dynamic programming gives the exact label

$$s_2(Y, P) = \frac{2}{(k-1)(k-2)} \sum_{s=1}^{k-2} \sum_{t=s+2}^{k} \log N_G(p_s, p_t). \tag{12}$$

Compatibility with the selected profile. The adjacency channel is part of the initial pair color. Hence the stable pair color $a = c^\star_{ij}$ determines A_{ij}. By Lemma H.7, it also determines the multiset $M(a) = \{\{(c^\star_{iu}, c^\star_{uj}) : u \in V\}\}$, including multiplicities. Set $f_1(c^\star_{ij}) = A_{ij}$ and recursively define

$$f_{\ell+1}(a) = \sum_{(b,c)\in M(a)} f_\ell(b) f_1(c).$$

Induction gives $f_\ell(c^\star_{ij}) = (A^\ell)_{ij}$. In a DAG, every directed walk is a path of length at most $n-1$, so $N_G(i, j) = \sum_{\ell=1}^{n-1} f_\ell(c^\star_{ij})$. The selected profile contains n, k, and these ordered pair colors; it therefore determines Equation (12). The question for Task 2 is how well a finite trained encoder and reader recover this information.

Target normalization. The controlled Task 2 manifest contains 2048 training graphs; optimizer updates use its fixed 512-graph subset, while normalization uses the full pool (mean 0.8260, standard deviation 0.3753, rounded). The additional 1536 graphs contribute training-label statistics only. All models and training budgets use these same statistics; validation and test labels remain excluded from preprocessing.

Prediction and computational tradeoff. TT has the highest median among reusable models, while MarkedEVTPath scores higher but recomputes its graph encoding for every query (Table 3). The budget sweep below assesses how the reusable-model comparison varies with training duration and seed.

GINE-Path without the raw-input bypass reaches median $R^2 = 0.239$, versus 0.190 for the bypass variant used throughout the main comparison.

D.3.1 SELECTED VERSUS JOINT ACCESS TO A FROZEN CACHE

To isolate the effect of access to off-path pair states, we train one width-44 PP-MLP model per seed (0–2) for 2,000 updates, then freeze its validation-selected EVT graph encoder. Both readers use this same cache within a seed. The manifest, 512-graph optimization subset, and target normalization pool are those of the main Task 2 comparison. Encoder training uses the shared optimizer settings, with 200 warm-up updates, validation every 100 updates, evaluation batches of 32, and zero minimum-improvement threshold. All runs use one CPU thread.

Both readers retain a flattened, ordered selection of all path-vertex and $P \times P$ states. An additional branch first combines the ordered tuple $(\widetilde{y}_{p_s u})_{s=1}^{k}$ of cached pair states using a shared MLP, then sums over u. The selected-only control restricts this sum to distinct path vertices; the joint reader sums over all graph vertices. The two branches are concatenated with $\log(1+n)$ and $\log(1+k)$, constant across this dataset, before scalar prediction. Thus the intervention changes which vertices contribute to the joint branch, while preserving the full ordered selection.

Each reader has 186,049 trainable parameters: the flattened-selection and joint branches have widths $64 \to 32$, followed by a $66 \to 64 \to 1$ readout, with GELU hidden activations. Readers share initial weights and minibatch draws within each seed, then train separately for 800 updates. AdamW uses weight decay 10^{-4}, gradient clipping at 5, and batches of 64 queries sampled with replacement. We screen constant learning rates $\{5 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}\}$, selecting checkpoints every 40 updates on validation MSE. Each reader's learning rate minimizes mean validation MSE across the three seeds; both select 10^{-4} before test evaluation. Encoder and reader parameters are fitted in separate stages, unlike the end-to-end models in Table 3.

Table D.6: Task 2 frozen-cache comparison: test R^2 after validation-only learning-rate selection. The last column gives mean $\pm$ sample standard deviation over three seeds; the last row contains paired differences.

Reader	Seed 0	Seed 1	Seed 2	Mean $\pm$ SD
$P \times P$ control	**.465**	**.509**	.616	$\mathbf{.530 \pm .078}$
Joint $P \times V$	.461	.504	**.617**	$.528 \pm .081$
Joint minus control	$-.004$	$-.005$	$+.002$	$-.002 \pm .003$

The readers achieve similar scores: the paired change is negative on two seeds and slightly positive on the third. This result concerns a finite cache trained for selected-only prediction. Although broader access could help such a reader, the hierarchy leaves its fixed-budget performance open. The control computes off-path contributions before masking them, so these runs compare predictive performance rather than optimized reader costs. The per-seed scores are retained in `figures/data/measurements.json`; `python reproduce.py figures` reconstructs `outputs/figures/t2-frozen-readers.csv`. This numerical summary does not rerun encoder fitting, reader selection, or checkpoint inference. The supplement does not provide a complete standalone driver for this two-stage frozen-reader control, and its historical weights are not bundled (Appendix L.1).

D.3.2 Optimizer-budget sensitivity

EVT-Path, PP-MLP, and TT are trained for $400, 800, 1200, 1600$, and 2000 updates, crossed with the same five seeds. Data, model configurations, and the other Task 2 settings in Appendix D.1.2 remain fixed. Each horizon is a separate fit with its own cosine schedule and a validation-selected checkpoint.

All ten models in the Task 2 comparison and all runs in this budget study use the same frozen manifest, whose generation recipe and content fingerprint are included in the supplement; its arrays are generated locally. The 2000-update entries for EVT-Path, PP-MLP, and TT reuse the fits reported in Table 3.

Table D.7: Controlled Task 2 optimizer-budget sensitivity. Values are median test R^2 [min,max] over all five seeds; the last column counts paired seeds on which balanced TT exceeds PP-MLP. Bold marks the best median at each budget.

Updates	EVT-Path	PP-MLP	TT	TT > PP-MLP
400	-0.000 [-0.015,0.091]	**0.036 [-0.005,0.176]**	0.000 [-0.005,0.023]	1/5
800	0.359 [0.303,0.452]	0.354 [0.300,0.427]	**0.421 [0.123,0.534]**	3/5
1 200	0.466 [0.345,0.588]	0.476 [0.391,0.551]	**0.542 [0.377,0.649]**	3/5
1 600	0.505 [0.395,0.662]	0.509 [0.406,0.593]	**0.615 [0.455,0.645]**	4/5
2 000	0.511 [0.435,0.697]	0.537 [0.411,0.624]	**0.637 [0.419,0.661]**	4/5

Performance is low at 400 updates; TT has the highest median from 800 updates onward. Rankings nevertheless vary across seeds: TT exceeds EVT-Path on $2/5, 3/5, 3/5, 3/5$ paired seeds at 800–2000 updates, while the corresponding counts against PP-MLP appear in the table. Score ranges overlap at each horizon.

Optimization diagnostic. The recorded curves in Figure D.1 show continuing improvement beyond 800 updates for the stronger graph models. Both faithful and direction-mixed ET reduce training error while validation error remains high.

The rendered historical figure is retained in the article, but its original step-by-step numerical histories are unavailable in the supplement. Final scores and the independent budget sweep do not determine these curves. The default figure command reports this limitation and does not synthesize replacement trajectories. Actual new or recovered histories can be supplied using `--histories PATH`; `--require-paper-curves` fails when the required histories are absent. Its coverage check requires all ten canonical Task 2 models, seeds 0–4, and 2,000 updates, and does not certify identity with the historical runs. Supplied histories are plotted separately from the archived measurements; see `figures/README.md`.

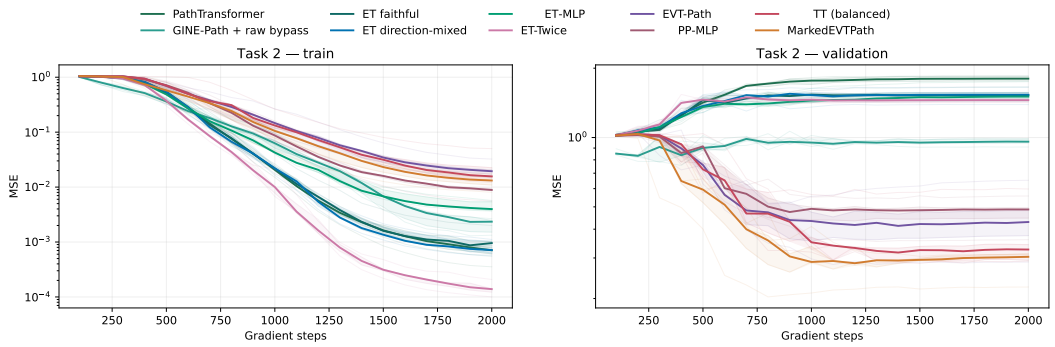


Figure D.1: Task 2 training and validation MSE over 2000 updates, from the archived training curves. The training panel shows recorded minibatch losses.

Bounded ET optimization controls. The canonical Task 2 comparison uses a shared initial learning rate (LR) of 3×10^{-4}. The following additional controls examine classical and direction-mixed ET on Task 2, using training and validation data only. We first screen initial learning rates $\{10^{-4}, 3 \times 10^{-4}, 10^{-3}\}$ for classical and direction-mixed ET, each over seeds 0–1 and 2,000 updates: twelve CPU fits and 24,000 updates in total, with equal search budgets for the two orientations. Both retain their 312,717 parameters, the Task 2 manifest and 512-graph optimization subset. AdamW uses weight decay 10^{-4}, zero dropout, 200 warm-up updates and cosine decay to 5% of the initial rate. Training batches contain 64 queries; evaluation batches contain 32. Checkpoints minimize validation MSE, checked every 100 updates with no minimum-improvement threshold; every fit completes the full budget.

An input audit independently recomputes the targets and checks normalization, padding, vertex relabeling, the mean readout, and gradients through both encoder stages. All checks pass. We then fit the first 16 optimization graphs (64 queries), with the same architectures and seeds 0–1. These four small-set fits use 800 updates, AdamW at a constant rate of 10^{-3}, zero weight decay and dropout, batches of 16, and clipping at 5. Checkpoints minimize MSE on all 64 training queries, evaluated every 100 updates. All four training R^2 scores exceed 0.99998.

Mean validation MSE in the LR screen selects 10^{-4} for both orientations. At this rate, we separately increase weight decay to 10^{-2} or set dropout to 0.1, retaining weight decay 10^{-4} in the latter condition. Dropout applies to all dropout modules, including graph and path attention, residual and feed-forward branches, and the head. These eight fits retain the LR screen's other settings and 2,000-update budget; the four existing fits at 10^{-4} supply the reference. The three controls together use 24 fits and 43,200 updates, divided equally between the two ET orientations.

Table D.8: Task 2 ET controls. Top: validation R^2, mean $\pm$ sample SD over two seeds; bold marks the best setting for each ET variant, including ties at displayed precision. Base setup uses weight decay 10^{-4} and zero dropout. Bottom: training R^2 on the 16-graph subset, reported per seed and rounded to six decimals. The five-seed test comparison is unchanged (Table 3).

Validation R^2: mean $\pm$ SD over seeds 0–1			
Setting	Initial LR	Classical ET	Direction-mixed ET
Base setup	10^{-4}	$\mathbf{-0.0059 \pm 0.0011}$	$\mathbf{-0.0065 \pm 0.0000}$
Base setup	3×10^{-4}	-0.0071 ± 0.0044	-0.0075 ± 0.0026
Base setup	10^{-3}	-0.0106 ± 0.0030	-0.0071 ± 0.0020
Weight decay 10^{-2}	10^{-4}	$\mathbf{-0.0059 \pm 0.0011}$	$\mathbf{-0.0065 \pm 0.0000}$
Dropout 0.1	10^{-4}	-0.0068 ± 0.0001	-0.0068 ± 0.0011
Training R^2: 16 graphs, 64 queries			
Seed	Constant LR	Classical ET	Direction-mixed ET
0	10^{-3}	0.999999	**1.000000**
1	10^{-3}	0.999990	**1.000000**

All twenty full-fit validation R^2 values remain negative. Increasing weight decay or dropout brings no improvement over the selected LR setup. The LR fits end with recorded minibatch MSE between 8.2×10^{-5} and 0.034. Together with near-perfect small-set fitting, these checks support a generaliza-

tion difficulty within the tested protocol that persists across the explored rates and regularizers. This sequential validation study covers the two ET models on Task 2; the main test table retains the common protocol. ET-MLP and ET-Twice separately evaluate full-pair path interfaces (Appendix D.1.3); their different widths and operators make them comparisons of complete systems. The retained validation and small-set numerical records are in `figures/data/measurements.json`. The figure command reconstructs `outputs/figures/t2-et-validation.csv` and `outputs/figures/t2-et-small-training.csv`; it does not rerun the input audit, learning-rate screen, regularization fits, or small-set fits. These auxiliary controls and the multi-horizon Task 2 budget sweep are covered by archived numerical summaries, not dedicated end-to-end retraining commands in the standalone supplement.

Fresh main-comparison fits. The standalone main Task 2 recipe runs all ten models and five paired seeds:

```
python reproduce.py synthetic --task T2 --output outputs/t2
python reproduce.py figures --histories outputs/t2 \
  --require-paper-curves
```

The training output directory must be new. The first command generates the fixed `s4-budget` inputs if needed, preserves the 2,048-graph normalization pool and 512-graph optimization subset, and writes new validation-selected checkpoints, test predictions, and histories. The second command uses those actual histories for a separate optimization plot and checks their coverage; it does not recover the historical curve data. Adding `--inspect` to the training command prints its recipe without fitting. The optional `--models`, `--seeds`, and `--smoke` switches restrict or shorten a run; such partial runs do not satisfy the strict ten-model, five-seed curve requirement. Further details are in `experiments/synthetic/README.md` and Appendix L.5.

D.4 Cache scaling and amortization

The timing study fixes 64 path queries of length $k = 7$ and varies $n \in \{28, 56, 112\}$ and the number $Q \in \{1, 4, 16, 64\}$ of distinct paths per immutable graph. Balanced TT and capacity-matched MarkedEVTPath use fixed random weights on an Apple M4 Max CPU (14 cores, 36 GB RAM), with each worker restricted to four PyTorch threads. Every cell aggregates ten fresh processes, each with two warm-ups and ten timed repetitions. TT time is measured as one call containing graph encoding and path processing for all queries; separate timings for cache construction and processing with an existing cache are retained only as diagnostics and their medians are never added to form the reported total. MarkedEVTPath performs its ordinary path-conditioned forward for every query. A second TT measurement processes paths in the same four-query chunks as MarkedEVTPath. Both cached variants are checked against the ordinary prediction before timing. The release records latency IQRs across processes, peak process memory (RSS), and exact retained-cache bytes separately.

Table D.9: Cache scaling. All times are end-to-end milliseconds/query, median [IQR] over ten fresh-process medians. Each TT total is a directly timed call covering cache construction and all path predictions. Native processes cached path queries in batches of up to 64; symmetric uses chunks of up to four queries for both models. Speedups divide MarkedEVTPath by the corresponding TT total. Bold marks the lowest time in each row.

n	Q	TT native	TT symmetric	MarkedEVTPath	cache/graph	native	symmetric
		milliseconds/query, median [process IQR]			MiB	speedup	
28	1	**4.480 [4.466,4.543]**	5.033 [4.976,5.160]	5.263 [4.872,5.364]	0.099	1.17×	1.05×
28	4	**1.264 [1.172,1.325]**	1.670 [1.610,1.714]	4.896 [4.699,5.044]	0.099	3.87×	2.93×
28	16	**0.491 [0.486,0.507]**	0.978 [0.965,1.003]	4.579 [4.564,5.102]	0.099	9.32×	4.68×
28	64	**0.304 [0.287,0.346]**	0.737 [0.701,0.866]	5.346 [5.158,5.530]	0.099	17.61×	7.25×
56	1	**14.568 [14.528,15.135]**	14.976 [14.925,15.433]	17.745 [17.618,18.146]	0.390	1.22×	1.18×
56	4	**3.764 [3.735,3.804]**	4.200 [4.179,4.234]	19.330 [18.854,19.755]	0.390	5.14×	4.60×
56	16	**1.113 [1.092,1.118]**	1.538 [1.522,1.562]	18.156 [17.853,18.669]	0.390	16.31×	11.81×
56	64	**0.473 [0.466,0.475]**	0.878 [0.873,0.881]	17.812 [17.746,17.953]	0.390	37.62×	20.29×
112	1	114.433 [113.796,117.680]	**113.203 [111.362,116.981]**	123.610 [122.921,124.604]	1.545	1.08×	1.09×
112	4	**28.547 [28.048,28.998]**	29.087 [28.555,29.516]	123.381 [122.531,124.000]	1.545	4.32×	4.24×
112	16	**7.268 [7.229,7.331]**	7.805 [7.766,7.843]	123.439 [122.591,123.811]	1.545	16.98×	15.81×
112	64	**2.060 [2.050,2.076]**	2.602 [2.578,2.622]	123.588 [122.604,124.256]	1.545	59.98×	47.49×

Under explicitly symmetric chunking, the ratio of median MarkedEVTPath to TT latency at $Q = 1$ ranges from $1.05\times$ to $1.18\times$. At $n = 28$, however, the fresh-process IQRs overlap substantially,

so this smallest case indicates similar latency. The $Q = 1$ process-IQRs separate at $n = 56$ and $n = 112$, and cache reuse gives large gains for $Q \geq 4$. Native-throughput values from $Q = 4$ onward are labelled amortization plus batching: at $n = 112$, they reach $4.32\times, 16.98\times, 59.98\times$ for $Q = 4, 16, 64$, versus $4.24\times, 15.81\times, 47.49\times$ under symmetric chunking. The symmetric column isolates reuse from the advantage of larger query batches and preserves a large gain. The exact retained cache grows from 0.099 to 0.390 and 1.545 MiB per graph as n doubles, consistent with dense ordered-pair storage. These CPU measurements quantify how repeated queries trade cache memory for lower inference time.

Numerical reconstruction. The per-process timing and cache-size records are retained in `figures/data/measurements.json` in the standalone supplement. `python reproduce.py figures` divides each recorded process time by its query count, then recomputes medians, IQRs, and speedup ratios in `outputs/figures/synthetic-reuse.csv` and renders the associated cache reuse plot. These outputs describe the original CPU measurements, not timings of the machine running the reconstruction command. No training or new timing campaign is performed; the distribution does not include a dedicated end-to-end driver for this fresh-process study (Appendix L.1).

E Atlas: Protocol, Temporal Evaluation, and Reuse

Atlas evaluates round-trip-time prediction along observed network paths. We first describe model selection on 4,096 validation queries, then report the temporal test, matched readers of frozen graph states, and the cost of reusing the graph encoding. The protocols and measurements below describe the original experiments. The standalone software, its data and model interfaces, and the distinction between numerical reconstruction and new model execution are documented in Appendix L; Atlas-specific entry points appear at the end of this section.

E.1 Protocol and selected configurations

The initial screen compared width-64 models at three common learning rates, seed 0, and 2,000 updates. Further architecture and learning-rate selection on validation retained PathTransformer, PP-MLP, and TT. These configurations were trained at seeds 0–5 for up to 8,000 updates; seed 0 also participated in selection. AdamW uses weight decay 0.01, gradient-norm clipping at 1, constant learning rate, validation every 500 updates, and early stopping after eight checks without improvement, following at least 2,000 updates. Effective query batches contain at most 16 paths. Loss and MAE average observed targets within each query and then weight queries equally. Masks exclude missing targets and padding. PathTransformer uses the same path self-attention blocks as in Task 2, with task-specific input projections and prediction head. Here it combines each path vertex with locally projected incoming and outgoing path-edge attributes before attention; Task 2 uses only vertex features. Neither variant learns a graph encoder. Atlas inputs also contain collective context features. All profile models add their prediction to a common historical prior; Appendix B.3 specifies this parametrization. PP-MLP and TT reuse EVT graph states, with different path processors. Selected configurations are listed in Table E.1. Models share the update ceiling; parameter counts and total search costs differ. In the result tables, bold marks the lowest MAE within each comparison, or the highest training throughput.

Task	Model	Width	Layers G/P	Heads G/P	LR	Params (k)
Atlas	PathTransformer	64	–/2	–/4	3e-04	101.2
Atlas	PP-MLP	64	2/–	4/–	1e-04	448.7
Atlas	TT	64	2/2	4/8	3e-04	778.9

Table E.1: Selected configurations. G/P denotes graph/path layers or heads; parameter counts are in thousands. Dashes denote absent attention stacks. These settings are selected for Atlas; other tasks use their own configurations.

All eighteen historical runs have complete training histories and checkpoint-prediction records in the experiment archive; these are not a claim that model weights or full prediction arrays are bundled

with the supplement. The main comparison pools seeds 1–5; seed 0 remains documented below. Standard deviations describe variation across training seeds.

E.2 ATLAS: KNOWN-TOPOLOGY LATENCY PROFILES

RIPE Atlas region-mesh measurements supply observed responding-interface paths and round-trip times (RTTs) (RIPE NCC Staff, 2015; RIPE NCC, 2026). Each region/month/destination-address graph retains up to 256 interfaces; paths have up to 32 occurrences. Snapshot topology and transition counts use unlabelled routes from that snapshot. The prior at each responding interface is the median of its valid February–March 2026 RTT observations, pooled across probes and targets. An interface absent from this history receives the median of all valid RTT observations in those two months. This fixed reference is shared by every model and also evaluated on its own. Training uses April–July and validation August. September was reserved for the later test. Probe/region/month blocks define the resampling units. The task predicts RTTs on known snapshot topology. Missing replies are masked. Responding interfaces may come from different packet routes, and RTT differences combine outward and return travel times rather than measuring one-way link delays.

At 3,000 updates, mean validation MAEs are close: 11.948 ± 0.068 ms for PathTransformer, 12.030 ± 0.036 for PP-MLP, and 11.899 ± 0.127 for TT. This update was chosen retrospectively as a common validation checkpoint. All nine Atlas runs reached 8,000 updates. At that endpoint, all three paired differences favor TT, whose mean error is about 3.5% lower than the two comparators (Table E.2). Its warm training throughput is lower.

Model	8k final MAE (ms)↓	Best checkpoint MAE (ms)↓	Warm train queries/s
PathTransformer	11.606 ± 0.180	11.366 ± 0.127	**295.9**
PP-MLP	11.614 ± 0.095	11.550 ± 0.058	74.5
TT	$\mathbf{11.202 \pm 0.264}$	$\mathbf{11.004 \pm 0.060}$	44.7

Table E.2: Atlas validation, mean $\pm$ seed SD over seeds 0–2. The fixed 8,000-update endpoint is distinct from the best validation checkpoint. Warm throughput is recorded training queries/s, not inference or measured cache-amortization speedup.

For selected checkpoints, resampling 688 whole probe blocks in 4,000 paired bootstrap draws, after averaging query errors over the three fitted seeds, gives improvements of 0.362 ms over PathTransformer (conditional 95% interval $[0.150, 0.649]$) and 0.546 ms over PP-MLP ($[-0.014, 1.398]$). These intervals describe resampling variation for the fitted models on the development validation set. They exclude training-seed and model-selection uncertainty. The temporal test below evaluates the selected systems with all choices fixed.

E.2.1 TEMPORAL TEST AND POOLED INITIALIZATION REPEATS

The first temporal evaluation used the selected checkpoints at seeds 0–2. Before evaluating September, all nine models' saved 4,096-query validation predictions were replayed exactly on scored entries. Architectures, learning rates, validation-selected checkpoints, and the 8,000-update endpoint were fixed before this evaluation. Full August validation comprises 30,838 paths and 218,474 labelled occurrences in 701 probe–region–month blocks; September comprises 30,419 paths and 213,556 labelled occurrences in 734 blocks.

Seeds 3–5 repeat the same configurations and training rules, with fresh initializations and sampling. They reuse September after its initial evaluation, with no further hyperparameter selection. We pool seeds 1–5 in Table E.3, excluding development seed 0 by its role in selection. This measures variation across trained instances on the same temporal test. The per-seed results, including seed 0, appear in Table E.4. All runs reached 8,000 updates; eight of the nine new runs still show late validation improvements, so this is a fixed-budget comparison rather than a convergence claim.

At selected checkpoints, TT's paired difference is -0.2824 ± 0.0806 ms against PP-MLP and -0.2609 ± 0.0362 ms against PathTransformer (mean $\pm$ seed SD), favoring TT in all five seeds. These describe initialization variation for the selected complete systems, whose architectural choices differ. The original block-bootstrap intervals were computed separately on three-seed panels; they are

System	Selected	8k endpoint
PathTransformer	11.201 ± 0.089	11.234 ± 0.128
PP-MLP	11.223 ± 0.069	11.253 ± 0.070
TT	$\mathbf{10.940 \pm 0.102}$	$\mathbf{10.953 \pm 0.118}$

Table E.3: Atlas September test: unclipped query-macro MAE in ms, mean $\pm$ sample SD over seeds 1–5. Selected denotes the checkpoint chosen on the 4,096-path August subset; 8k is the fixed training endpoint. All systems predict corrections to the common historical prior (MAE 12.809 ms).

Seed	PathTransformer	PP-MLP	TT
0 (development)	11.1079	11.2398	**10.8768**
1	11.2580	11.2115	**11.0263**
2	11.2351	11.1639	**10.9261**
3	11.0732	11.2263	**10.8284**
4	11.2918	11.3375	**11.0625**
5	11.1480	11.1748	**10.8585**

Table E.4: Atlas test MAE at validation-selected checkpoints. Seeds 1–5 enter the pooled comparison; seed 0 participated in model development and is shown for completeness. The same seed identifies independently trained models with matched sampling within each repetition.

not confidence intervals for this five-seed aggregate. We report seed variation for the pooled panel, without claiming uncertainty coverage over test blocks or model selection.

E.2.2 Selection and joint reading on identical frozen caches

We freeze the three original PP-MLP graph encoders (seeds 0–2), trained end to end for RTT prediction. They use width 64, two four-orientation graph layers, four heads, learning rate 10^{-4}, and the optimizer settings above. Their selected checkpoints are updates 6,500, 8,000, and 7,500. Both readers consume the same fingerprinted vertex and pair tensors; their weights start from scratch while graph encoders remain fixed.

Each reader has 133,065 trainable parameters and two branches. The first reads the ordered $P \times P$ selection. The second combines relations to all ordered path occurrences for each candidate vertex before aggregating across candidates. Selection restricts candidates to the distinct visited vertices; joint reading uses every valid graph vertex. Both directed pair states, vertex states, occurrence positions, and vertex-equality indicators are supplied, with padding masks and explicit candidate counts. Thus the selection control has the same additional nonlinear branch as the joint reader; it is a new matched reader, not the original PP-MLP head.

The common search uses three learning rates $(10^{-4}, 3 \cdot 10^{-4}, 10^{-3})$ and two weight decays $(0, 0.01)$, with 3,000 updates per reader and encoder: 36 fits. Mean endpoint validation MAE over the three encoders selects $3 \cdot 10^{-4}$ and 0.01 for both readers. Two new reader initializations (31 and 32) per encoder then run for 8,000 updates, producing 12 fits with matched sampling. The primary checkpoint minimizes MAE on the fixed August subset. Full August and the previously evaluated September test are scored only after this selection.

Reader	August selected	September selected	September 8k
Selection ($P \times P$)	$\mathbf{10.706 \pm 0.059}$	$\mathbf{10.956 \pm 0.045}$	$\mathbf{10.953 \pm 0.036}$
Joint ($P \times V$)	10.733 ± 0.044	11.050 ± 0.054	10.997 ± 0.049

Table E.5: Frozen Atlas caches: query-macro MAE (ms). We average two reader initializations within each encoder, then report mean $\pm$ SD across the three encoders. Selected and 8k distinguish validation selection from the fixed endpoint. The encoder checkpoints were selected for the original PP-MLP task, before fitting these readers.

Joint reading gives no gain at the tested budget (Table E.5). Its selected-checkpoint difference from selection is $+0.0935$ ms, with conditional 95% block-bootstrap interval $[-0.0021, 0.2050]$; at 8k the difference is $+0.0436$ ms ($[0.0061, 0.0992]$). The recorded intervals use 10,000 resamples of the 734 whole test blocks after averaging fitted readers; they exclude training and selection uncertainty. Cache

hashes, branch gradients, and paired sampling were checked. Eleven confirmation runs still show late validation improvements. The result concerns exploitation of these finite PP representations; the strict information hierarchy permits, but does not require, a predictive gain on this task.

E.2.3 SAME-MODEL INFERENCE REUSE ON ATLAS

For each selected seed-0 model, we compare three executions of identical weights: graph encoding independently for every query, sharing within each physical batch, and a persistent cache shared across all query batches. The completed panel uses physical batches $B \in \{1, 4, 8\}$ and $Q \in \{1, 4, 16, 32\}$ distinct paths where feasible ($B \leq Q$). Nine graphs support all query counts: they contain 182–256 vertices and 33–48 distinct paths, with path lengths 3–16. Three additional graphs with paths of length 18–19 provide low-reuse controls at $Q = 1$. Graphs are padded to 256 vertices. The earlier two-case, batch-one measurements remain in the evidence archive; the table below uses the completed nine-graph panel.

Measurements use one TPU v6e, FP32, seven synchronized warm repetitions per case and mode, and include initial cache construction. The panel covers 84 graph/query/batch cells and 840 execution-mode cells, with no compilation in the 5,880 warm repetitions. Maximum prediction discrepancy across paired executions is below 0.001 ms. Times are medians across graphs of their within-case median times; speedups are medians of paired case ratios.

Model	B	Independent	Within batch	Persistent	I/P	W/P
PP-MLP	1	434.34	434.24	**84.83**	5.13×	5.12×
PP-MLP	4	113.50	109.22	**29.82**	3.82×	3.66×
PP-MLP	8	57.96	54.94	**20.71**	2.80×	2.66×
TT	1	753.10	753.52	**401.56**	1.88×	1.88×
TT	4	194.41	188.86	**108.88**	1.79×	1.73×
TT	8	99.44	95.61	**60.67**	1.64×	1.58×

Table E.6: Atlas same-weight inference at $Q = 32$ distinct paths, nine graphs. Warm device time in ms includes graph encoding and path processing, excluding compilation and host transfers. Physical batch B is identical across modes. I/P and W/P compare independent and within-batch execution with persistent reuse, respectively. Bold marks the lowest time for each model and batch size. Each graph cache stores 16.0625 MiB of FP32 tensors.

At $B = 8$, persistent reuse takes 0.647 ms per path for PP-MLP and 1.896 ms for TT. TT benefits from reuse but its costlier path stage leaves a smaller speedup. Including input transfer and output collection raises the corresponding $Q = 32$ totals to 44.36 and 79.78 ms; a path-only execution that avoids materializing unused graph states takes 21.26 ms. These are synchronized request latencies, not asynchronous throughput. Separate cold-process probes take 26.09 s for persistent PP-MLP and 27.09 s for TT at this shape, mostly compilation. Their recorded allocator peaks are 59.52 and 72.83 MiB; tensor cache size and allocator accounting measure different quantities from total accelerator memory.

Full 4,096-query validation replays match the stored TPU predictions. A separate CPU diagnostic leaves three of 57 PP predictions outside its replay tolerance (maximum gap 0.007612 ms); this portability discrepancy is distinct from same-device execution parity. The measurements concern inference; differentiable training has separate recomputation costs (Appendix D.4).

E.3 STANDALONE DATA, MODEL, AND REPRODUCTION INTERFACES

All paths and commands here are relative to the extracted supplement, after installation as in Appendix L.2. The supplement includes the numerical records under `figures/data/`, original Atlas source and training recipes under `experiments/atlas/historical/`, and historical checkpoint identities under `experiments/atlas/resources/`. It contains neither trained Atlas weights nor complete graph/path or prediction arrays.

Data and reusable models. `python reproduce.py data fetch atlas` retrieves 256 checksum-pinned public RIPE Atlas archives and reconstructs the benchmark without the development repository. `python reproduce.py data verify atlas` checks the reconstructed

arrays against the fixed content identities, including path order, masks, priors, labels, and temporal splits. An existing raw-data cache can instead be passed to `python reproduce.py data build atlas --raw PATH --root data/atlas`. The preparation protocol and checksums are in `datasets/resources/atlas/`; see also Appendix L.3.

Atlas graph/path shards are loaded by `datasets.AtlasDataset`. The factory `build_atlas_model` in `models/` exposes PathTransformer, PP-MLP, and TT. The returned model predicts an occurrence-level residual; evaluation multiplies it by 10 and adds the historical RTT prior to obtain milliseconds. Graph encoding and path queries can be called separately for reuse while inputs and weights remain fixed. The public factory's default configuration is not the validation-selected architecture: use the recorded recipe (Appendix L.4).

New training and checkpoint evaluation. The following commands inspect the selected TT recipe, train a new instance, and explicitly score its complete September partition:

```
python reproduce.py atlas-train \
  --run-id atlas_confirmation_tt_s1
python reproduce.py atlas-train \
  --run-id atlas_confirmation_tt_s1 \
  --device cpu --seconds 10800 \
  --output outputs/atlas-tt-s1 --execute
python reproduce.py atlas-evaluate \
  --checkpoint outputs/atlas-tt-s1/best.pt \
  --partition test --device cpu \
  --output outputs/atlas-test-tt-s1
```

Without `--execute`, recipe inspection loads no data or weights. The training adapter preserves the original optimizer and validation rules; run identifiers use `simple`, `pp`, or `tt` and seeds 0–5. Seeds 3–5 reuse the selected settings. The explicit wall-clock limit is at most 10,800 seconds; a CPU may reach fewer than 8,000 updates, unlike the completed historical runs. New fits therefore require their own completed budget and selection records, and must not be relabelled as historical weights. The pooled comparison still excludes development seed 0.

Numerical reconstruction and historical replay. `python reproduce.py figures` reconstructs the Atlas development, temporal, frozen-reader, and reuse tables from bundled measurements, including seed variation and paired timing ratios. This neither reruns model selection nor regenerates conditional bootstrap intervals or TPU timings. The frozen-reader and reuse controls are included at this numerical-record level, not as complete new training or timing campaigns.

`python reproduce.py atlas-replay inspect` lists the nine original seed-0–2 run identities without inference. Historical validation replay additionally requires separately supplied `best.pt` and `best_validation_predictions.npz` files via `--checkpoint-root`. The adapter verifies their recorded hashes and the canonical reconstructed dataset before scoring the 4,096-query cohort; it reports cross-backend differences using the original tolerances. Use `atlas-evaluate` for new checkpoints, rather than comparing them as recovered historical states. The historical TPU parity and CPU discrepancy reported above remain distinct from the supplement's software integration checks. `VALIDATION.md` records those checks and their limits; Appendices L.1 and L.5 summarizes reproduction coverage and the complete training interfaces.

F ThorAxe-Combo: Benchmark Details

F.1 Candidate construction, cohorts, and graph inputs

We use the fixed spliced-exon segments and protein-isoform paths produced by ThorAxe (Zea et al., 2021). For each gene–clade group, paths observed in the designated clade are reserved for candidate eligibility and target annotation; paths observed outside that clade define the visible graph and supply two donor paths. A gene may contribute several clade groups. The donor paths are split at shared spliced-exons, and candidates recombine the intervening alternative blocks, each potentially containing several spliced-exons. Candidate paths and focal positions are determined from these visible donor paths before reserved-clade annotations are applied.

Candidates recombine at least two differing blocks, each observed in a protein-isoform path from the reserved clade. The two focal positions correspond to the first and last differing blocks; the number of recombined blocks may exceed two. The complete candidate path must be absent from the visible paths. Its label is 1 if it is observed in the reserved clade and 0 otherwise. A negative label records absence from the available annotations; biological compatibility remains open.

Table F.1: ThorAxe-Combo cohort drawn from 5,000 source genes, retaining visible graphs with at most 128 vertices and groups containing both classes. Counts cover the union of the five outer test folds.

Property	ThorAxe-Combo
Eligible genes	1,072
Gene × clade groups	1,596
Candidates	16,544
Positive / negative candidates	3,120 / 13,424
Median candidates per group	5
Groups with one positive	899

Graph and sequence features. The input graph is rebuilt from visible protein-isoform paths, with directed edges between consecutive spliced-exons. For each spliced-exon, precomputed 960-dimensional ESM-C 300M embeddings (EvolutionaryScale, 2024) are averaged over visible instances within each species, then across species. Four flags indicate non-boundary, start, stop, and embedding availability, giving 964 vertex channels. Seven pair channels encode forward and reverse adjacency, visible-species and visible-path support fractions, intron presence, normalized log nucleotide gap, and a self-pair indicator. Support fractions concern adjacent edges. Graph-based models use an input stem reducing vertex and pair features to 8 and 4 channels; GINE also receives the compressed vertex features and directed adjacency.

F.2 Models, readouts, and parameter budgets

The graph-based comparison evaluates thirteen encoder-readout conditions: three readouts for PP-MLP, ET-MLP, TD, and TT, plus the vertex-state readout for GINE-Path. TT uses triangular pair attention at both stages. TD keeps triangular graph attention and uses dense self-attention over the k^2 path-pair tokens, while retaining the EVT vertex stream and cross-attentions; its path-pair work is $O(k^4 d)$ against TT's $O(k^3 d)$. GINE-Path applies directed GINE message passing followed by a path Transformer.

PP-MLP and ET-MLP use an EVT and an Edge Transformer graph encoder, respectively. After selection, a shared MLP transforms each pair state and a second MLP combines its row mean with the vertex state:

$$R_{st} = \phi(\widehat{Y}^0_{st}), \qquad Z_s = \rho\Big([\widehat{X}^0_s; k^{-1}\textstyle\sum_t R_{st}]\Big),$$

on the selected, positioned inputs $\widehat{X}^0, \widehat{Y}^0$. The biological ET-MLP uses chain-only triangular attention at width 80, whereas synthetic ET-MLP uses four orientations at width 88 (Appendix D.1.3). Both use the same endpoint-lifted ET encoder implementation, with ReLU in its graph blocks and GELU in the pair and vertex MLPs after selection; their MLP widths and task heads differ. EVT uses the balanced four-orientation allocation. All graph encoders have two layers, and the attention-based path processors have two layers and four heads.

These graph-model readouts include focal vertex states, the mean vertex state along the path, and a learned clade embedding. These vertex states already contain graph context; the pair-based readouts additionally access pair states directly. The vertex-state-only readout projects the focal vertex branch to 64 channels. The other readouts project focal vertices and pair features to 32 channels each. Writing the focal positions as (a, c), the focal-pair readout uses $[R_{ac}; R_{ca}]$; the path-edge readout uses $g_R = (k-1)^{-1}\sum_{s=1}^{k-1} R_{s,s+1}$ over the entire valid path. These consecutive pairs are splice junctions in the biological graph; the configuration files retain the name `junction` for this readout. Pooling covers the full path and excludes padding. GINE-Path is evaluated with the vertex-state readout, its available output in these runs.

Table F.2: Trainable graph-to-path parameter budgets for path-edge readouts, except GINE-Path's vertex-state readout. Other readouts have slightly different counts. Precomputed ESM-C embeddings are inputs to these predictors.

Model	Hidden width	Parameters
PP-MLP	32	351079
ET-MLP	80	349679
GINE-Path	104	341963
TD	32	351079
TT	32	351375

These comparisons approximately match total parameter counts while allowing differences in latent width, capacity allocation, FLOPs, and memory.

ESM-C focal-and-mean baseline. This baseline reads the two focal 960-dimensional ESM-C embeddings, the mean ESM-C embedding over the full candidate path, and a learned 16-dimensional clade embedding. Learned projections map the concatenated focal embeddings and the path mean to 64 channels each; a $144 \to 64 \to 1$ classifier produces the prediction. The projections and classifier use LayerNorm and GELU. Valid boundary positions and missing embeddings contribute zeros to the path mean; padding is excluded. Inputs are limited to the embeddings and clade identity, bypassing the $964 \to 8$ stem. Its 200,033 trainable parameters belong to the prediction head, compared with roughly 350,000 for the graph-based models. ESM-C embeddings remain fixed.

Local sequence references. Two further references use a path Transformer (two layers, four heads, width 64, feed-forward width 256, rotary positions) on the ordered 964 raw vertex channels, with pointwise LayerNorm and a linear projection to width 64. They bypass the graph models' eight-channel input stem. The final head concatenates the two focal states, their full-path mean, a 64-channel junction branch, and a 16-channel clade embedding; its dimensions are $272 \to 64 \to 1$, with LayerNorm and GELU. In `sequence_z`, the junction branch is zero. In `sequence_junction`, a shared affine $7 \to 64$ map transforms the raw attributes of consecutive path pairs, then their masked mean enters the final head. These junctions undergo no graph propagation. The two conditions share module shapes and paired initializations, apart from the additional affine map, and are trained separately. They provide local-input references with their own parameter budgets; several architectural choices differ from the graph-based models.

F.3 Grouped evaluation, selection, and optimization

Five outer folds evaluate prediction on unseen genes by holding out whole gene identifiers, keeping all groups and candidates from a gene together. Within each outer fold and fixed encoder–readout condition, learning rates $\{10^{-5}, 3{\times}10^{-5}, 10^{-4}, 3{\times}10^{-4}, 10^{-3}, 3{\times}10^{-3}\}$ are compared on three inner partitions and two seeds. Selection maximizes mean inner macro-AUROC, breaking learning-rate ties toward the smaller value. The graph-based entries in Table 5 select both the readout and learning rate within each outer fold; Table F.3 fixes the readout and selects only its learning rate. Both decisions use inner validation only. There are 2,340 inner cells and 325 final fits across the thirteen conditions, with five seeds for final evaluation. Each of the three sequence references uses the same candidates, gene partitions, learning-rate grid, and selection criterion, with its readout fixed: 180 inner fits and 25 final fits over five seeds (540 inner and 75 final fits in total).

Among the 80% of genes outside the outer test, a separate partition is retained for checkpoint selection: approximately 64% of all genes train weights, 16% select the checkpoint, and 20% form the outer test.

AdamW uses the selected constant learning rate, no warm-up, weight decay 0.01, and global gradient clipping at 1. Each update processes one gene–clade group, with up to 16 queries. Graph-based models use micro-batches of four and evaluation chunks of eight; both sizes are one for the three sequence references. Training lasts at most 6,000 updates, with validation every 200 updates and early stopping allowed from update 1,000 after eight evaluations without improvement. Validation macro-AUROC selects the checkpoint, which may precede update 1,000. Early stopping and sampling can produce different completed update counts and numbers of passes over the cohorts.

Metrics and aggregation. AUROC and non-interpolated Average Precision (AP) are computed within each gene–clade group. For each seed, we average over all groups from the five outer tests with equal group weights, then report the mean and sample standard deviation of the five seed scores. This gives each group equal weight, regardless of its size or outer fold, and evaluates each seed separately. SD measures variability across training seeds. Under a uniformly random ranking, conditional on each group's candidate and positive counts, expected AP averages 0.5234, against an expected AUROC of 0.5.

F.4 Reference results and readout comparisons

Table F.3: Encoder–readout comparison: mean outer-test AUROC over five seeds. The head is fixed in each column and its learning rate selected internally. All columns include focal vertices, mean path-vertex context, and clade identity; "+" identifies the added relational branch. A dash denotes an untested condition. Bold compares encoders within each readout.

Model	Vertices	+ Focal pairs	+ Path edges
PP-MLP	0.632	0.618	**0.702**
ET-MLP	**0.654**	**0.630**	0.690
GINE-Path	0.643	–	–
TD	0.619	0.607	0.638
TT	0.620	0.612	0.635

Moving from vertex-state-only to path-edge readouts raises PP-MLP and ET-MLP by 0.070 and 0.036 AUROC, positive in all five fold averages and all five seed averages. These comparisons also change vertex projection width, capacity allocation, the selected learning rate, and end-to-end gradients. Inner selection chooses path-edge readouts for PP-MLP/ET-MLP/TD/TT in every fold.

The uncertainty analysis resamples whole genes within each outer fold 20,000 times, keeping predictions matched across models, with a sensitivity analysis resampling the five matched seeds. Intervals condition on the fitted models, splits, and selection decisions, without retraining or multiple-comparison correction. The AUROC difference between PP-MLP and TT is $+0.066$ with gene-and-seed interval $[0.044, 0.091]$.

Path processing across synthetic and biological tasks. The two MLP path-processor controls have the lowest mean scores in the six-model Task 1 comparison at 800 updates. Both improve substantially with longer training (Tables D.3 and D.4). On ThorAxe-Combo, PP-MLP has the highest graph-model mean AUROC, 0.702, after internal selection. PP-MLP's advantage over TT also depends on the readout: the respective scores are 0.632 versus 0.620 with vertex states and 0.702 versus 0.635 with path-edge states (Table F.3).

Task 1 requires products of three consecutive edge weights, explicitly testing composition after path selection. Biological prediction uses sequence features and observed path support already processed by a relational graph encoder. A plausible explanation is that the signal available in these encoded relations benefits more from direct path-edge readout than from another stage of triangular composition under the evaluated budgets. This interpretation remains a hypothesis. Full path length and annotation coverage may also matter: focal positions identify only part of a longer path, and labels record observed support. Separating these effects would require targeted controls.

F.5 Vertex-input projection width: an internal-validation pilot

We compare projections from 964 vertex-input channels to either 8 or 32 channels for PP-MLP and TT. Each model retains its width-32 encoder, path-edge readout, and four-channel pair-input projection; widening the vertex-input projection increases its parameter count by 23,928. All systems are retrained independently. The pilot uses inner folds 0 and 1 of outer fold 0, two training seeds, and learning rates $\{10^{-5}, 10^{-4}, 3\times10^{-4}, 10^{-3}\}$ per configuration, for 64 fits. The update ceiling and checkpoint-selection and early-stopping rules follow Appendix F.3. For each model and projection width, we select the learning rate with the highest mean validation macro-AUROC across the two folds and two seeds. All pilot scores come from inner validation.

Table F.4: ThorAxe-Combo vertex-input projection pilot: mean validation macro-AUROC over two inner folds and two seeds at the separately selected learning rate; bold marks the better projection width for each model. The selected rate is 10^{-4} for both PP-MLP widths and 3×10^{-4} for both TT widths. These are exploratory validation scores, distinct from the outer-test results in Table 5.

Model	Projection to 8 channels	Projection to 32 channels
PP-MLP	**0.706**	0.684
TT	0.613	**0.623**

Increasing the projection width slightly improves TT's selected validation AUROC, while reducing that of PP-MLP. PP-MLP remains ahead at both widths. The supplement retains the per-fit numerical scores under `figures/data/`; `python reproduce.py figures` reconstructs the selected-rate summaries in `thorax-projection-validation.csv`. This pilot is covered as a numerical control, not as a complete new training campaign exposed by the standalone interface.

F.6 Scope and reproducibility

The evaluation concerns unseen genes within the eligible population: all four reserved clades—birds, Laurasiatheria, primates, and rodents—also occur among training genes, and homologous families may recur. ThorAxe segmentation and alignments are fixed upstream. Excluding a clade rebuilds the visible graph and features from the remaining paths. Candidate eligibility uses reserved-clade support annotations.

The standalone supplement reconstructs the biological tables and seed summaries from group metrics, inner selection records, and final-fit identities under `figures/data/`. These records cover the graph systems and all three sequence references, including ESM-C. The command `python reproduce.py figures` recomputes learning-rate/readout selection from the archived inner scores and then aggregates outer-test group scores. It does not retrain models, regenerate per-candidate predictions, or recompute the original conditional bootstrap intervals. Those intervals remain the historically computed values reported above.

Unlike the compact numerical records, the dataset toolbox includes the fixed candidate cohort, all 20 gene-split files, input hashes, and one real gene for a loading example. Complete biological inputs must be retrieved separately; no trained predictor checkpoints or complete per-candidate prediction arrays are distributed. The downloaded `node_esmc_embeddings.pt` files are input representations, not fitted prediction-head weights. Their recovered bytes are checksum-verified, but their identity with the historical training embedding snapshot has not been independently established. New training on these inputs is therefore not an exact replay of the archived scores.

Comparability and verification. All models use the same candidate cohort and gene splits, with hyperparameters selected on internal validation data. Reported metrics were checked against saved predictions in the historical audit. Those checks covered all 325 retained graph-model runs and the sequence references' 540 inner and 75 final records; they are not new checkpoint replays performed by the supplement's figure command. The historical reproduction audit separately checked recovered inputs, rescored individual predictions, and performed a small fresh training/checkpoint round trip. The current release's source identities, input inventory, and numerical provenance are retained in `experiments/thorax/reference/`, `datasets/resources/thorax/`, and `figures/data/provenance.json`, respectively. `VALIDATION.md` reports the standalone software checks, including source verification, model-interface tests, cohort/split integrity, and campaign-generation checks; these are not a new full nested-training campaign.

F.7 Standalone dataset and training interfaces

Commands below run from the extracted supplement after installation (Appendix L.2). The fixed cohort and pretrained input representations are obtained from the Figshare archive identified in `datasets/resources/thorax/figshare.json`:

```
python reproduce.py data fetch thorax --root data/thorax
python reproduce.py data verify thorax \
  --root data/thorax/Dataset_v2_torch
python reproduce.py thorax prepare \
  --data-root data/thorax/Dataset_v2_torch \
  --work outputs/thorax
```

Retrieval checks the upstream checksum list, eight archive hashes, and 4,288 extracted input files; allow approximately 16 GB for archives and extracted data. Installation and ordinary loading do not download inputs. Preparation preserves the 1,072-gene, 1,596-group, 16,544-candidate cohort and its 3,120 positives. It does not regenerate the original 5,000-gene source population or upstream ThorAxe, Ensembl, and ESM-C processing.

The public `datasets.ThoraxDataset` interface exposes outer and inner gene splits, typed graph/query batches, focal positions, and clade identities; test loading requires explicit opt-in through `allow_test=True`. `models.build_thorax_model` exposes the five graph families and the local `PT`, `PT+J`, and `ESM-C` references. Its graph readout names are `vertex`, `focal-pair`, and `junction`. The general toolbox is convenient for new experiments; the historical graph and sequence campaigns use separately preserved source snapshots and isolated adapters, not an interchangeable modern factory (Appendix L.4).

For graph systems, `python reproduce.py thorax-campaign` generates sequential command files for `--mode inner`, `select`, and `final-new`. Executing those files performs the 2,340 inner fits, inner-only learning-rate/readout selection, and 325 final fits described above. Individual fits are exposed by `python reproduce.py thorax train`; `--dry-run` validates their prepared protocol without fitting. The three sequence-reference campaigns are exposed separately by `python reproduce.py thorax-sequence-campaign`, with `prepare`, inner `commands`, `select`, and final `commands` phases covering 540 inner and 75 final fits. Both workflows write new checkpoints, predictions, and optimization records. The explicit `final-archived` alternative reuses recorded inner choices instead of running a new nested search; this distinction must be retained when reporting results.

`experiments/README.md` and `experiments/thorax/SEQUENCE.md` give the complete commands and selection checks. Missing historical weights are not supplied by these adapters or by the metadata inventories in `experiments/resources/`. Appendices L.1, L.3 and L.5 summarizes the separation between numerical reconstruction, data preparation, and new training.

G EVT Operators and Normalization

This appendix specifies the operations used by the proofs: normalization, MLP updates at each vertex or pair, tensor reshaping, and both attention operators. We suppress layer, branch, head, and parameter subscripts: every displayed occurrence of Norm, Att, TriAtt, MLP_X, and MLP_Y owns its trainable parameters unless sharing is explicitly stated. Parameters are shared over the relevant vertex or pair indices, not across the two EVT stages. The maps flat and unflat are parameter-free.

G.1 Overview of the EVT updates

Vertex and pair states exchange information through attention and shared MLP updates, as shown in the detailed appendix diagram (Figure G.1a). Each macro-layer consists of two steps. The first step updates vertices by self-attention and pairs by triangular attention in parallel, using the oriented views in Equation (1). Let m denote the number of vertices or path occurrences in this stage. For one head, W_Q, W_K are learned query/key projections of width d_a, and W_L, W_R are learned value projections of a common width. Tokens are column vectors in these formulas, $\odot$ is the element-wise product, and $\mathrm{softmax}_r(a) = e^{a_r} / \sum_{u=1}^{m} e^{a_u}$. For orientations $(\alpha, \beta) \in \{\rightarrow, \leftarrow\}^2$, with normalization and head indices suppressed,

$$\mathrm{TriAtt}^{\alpha,\beta}(Y)_{ij} = \sum_{r=1}^{m} \mathrm{softmax}_r\left(\frac{\langle W_Q y^{\alpha}_{ir}, W_K y^{\beta}_{rj}\rangle}{\sqrt{d_a}}\right)\left(W_L y^{\alpha}_{ir} \odot W_R y^{\beta}_{rj}\right), \tag{13}$$

Each head has independent projections and its own softmax over the intermediate vertex r. Its value combines both pair states through an element-wise product. For a scalar pair channel $A \in \mathbb{R}^{m\times m}$,

under Definition H.4, zero logits and identity value projections yield $m^{-1}A^2$, $m^{-1}AA^\top$, $m^{-1}A^\top A$, or $m^{-1}(A^\top)^2$: a directed chain, a common target, a common source, or a reverse chain, respectively. The first follows Edge Transformers (Bergen et al., 2021); the middle two match the bilinear cores of AlphaFold's triangle multiplication (Jumper et al., 2021). EVT uses these orientations in content-dependent attention heads, retaining cubic dense complexity at fixed total width (Equation (98)).

The residual updates in Equation (2) use the following pair of independent MLPs:

$$\mathrm{AlMLP}(X, Y) := \big((\mathrm{MLP}_X(x_i, y_{ii}))_i, (\mathrm{MLP}_Y(y_{ij}, x_i, x_j; \mathbf{1}_{\{i=j\}}))_{ij}\big)\,. \tag{14}$$

The independent MLPs share parameters over indices. A vertex receives its self-pair; a pair receives its own state and ordered endpoints. The pair MLP never reads y_{ji} directly. Reverse-pair access instead enters through oriented triangular attention. The second step applies dense cross-attention in both directions. Each vertex attends to all pair states, flattened into a list, and each pair attends to all vertex states. Both cross-attentions read the states produced by the first step and are followed by another AlMLP update with separate parameters. Every attention and MLP update is residual. The complete equations are given in Appendix G.6.

The practical implementation uses pre-LayerNorm. At the path-stage input, position and repeated-vertex indicators are injected once and mixed into the learned states. The theoretical conventions are given in Definition H.4.

G.2 Domains, symmetry, and practical interfaces

Learning on graphs with ordered paths. Vertex names may change without changing the prediction. Path positions specify the order of the query and remain fixed when vertices are renamed.

Fix maximum sizes $N, K \geq 1$ and an output width $q \geq 1$. For $1 \leq n \leq N$ and $1 \leq k \leq K$, let

$$\mathcal{D}_{n,k} := \mathbb{R}^{n \times d_X} \times \mathbb{R}^{n \times n \times d_Y} \times [n]^k, \qquad \mathcal{D}_{\leq N, \leq K} := \bigsqcup_{n=1}^{N} \bigsqcup_{k=1}^{K} \mathcal{D}_{n,k}. \tag{15}$$

Each component has the Euclidean topology on features and the discrete topology on $[n]^k$; $\mathcal{D}_{\leq N, \leq K}$ has the finite disjoint-union topology. An input $u = (X, Y, P) \in \mathcal{D}_{n,k}$ consists of vertex features $X = (x_i)$, dense ordered-pair features $Y = (y_{ij})$, and an ordered index sequence $P = (p_1, \ldots, p_k)$, which we call a path. Repetitions are allowed, and the architecture is defined for every $P \in [n]^k$: it does not require consecutive indices to be adjacent. In typical graph applications, P is nevertheless restricted to a valid walk. When y_{ij} contains an edge indicator A_{ij}, this is the subdomain defined by $A_{p_s p_{s+1}} = 1$ for every consecutive pair; all results below restrict automatically to that subdomain.

Here and throughout, "edge" in Edge–Vertex Transformer refers to the dense, relabeling-equivariant ordered-pair index set $[n]^2$, including diagonal and nonadjacent pairs, rather than only to the edges of a sparse graph. If an input graph has adjacency matrix A, then A_{ij} is a coordinate of y_{ij}, together with any pair or edge attributes; a declared null value represents edge-only attributes when $A_{ij} = 0$. Outputs belong to

$$\mathcal{O}_K := \bigsqcup_{k=1}^{K} \mathbb{R}^{k \times q}, \tag{16}$$

with the finite disjoint-union topology. Every architecture below is one map on $\mathcal{D}_{\leq N, \leq K}$: the same parameters act on every size component.

The idealized family reads final vertex coordinates as

$$H_{\theta_G, \theta_P}(X, Y, P)_s = \pi_{\mathrm{out}}(x_s^{\mathrm{P}}). \tag{17}$$

For a domain subset S, compatible predictions are

$$\mathcal{C}_\Psi(S) := \left\{ F \in C(S; \mathcal{O}_K) : \begin{array}{l} F \text{ is length preserving, and} \\ \ker(\Psi_{N,K}|_S) \subseteq \ker(F) \end{array} \right\} \tag{18}$$

Let $\mathfrak{C}_N$ be the common space of depth-R_N formal colors. The profile of Definition 1 takes values in $\bigsqcup_{n=1}^{N} \bigsqcup_{k=1}^{K} (\{n\} \times \{k\} \times \mathfrak{C}_N^{k \times k})$. For length-preserving F and G, the uniform norm is

$$\|F - G\|_{\infty, S} := \sup_{u \in S} \|F(u) - G(u)\|_{\text{Frobenius}}. \tag{19}$$

For a graph relabeling $\sigma \in \mathfrak{S}_n$, define

$$(\sigma \cdot X)_{\sigma(i)} = x_i, \qquad (\sigma \odot Y)_{\sigma(i)\sigma(j)} = y_{ij}, \qquad (\sigma P)_s = \sigma(p_s). \tag{20}$$

The target symmetry is

$$H(\sigma \cdot X, \sigma \odot Y, \sigma P) = H(X, Y, P). \tag{21}$$

Permuting path positions is not a symmetry unless separately imposed.

Flattening convention. For $Y \in \mathbb{R}^{m\times m\times d}$ and $Y^{\mathrm{flat}} \in \mathbb{R}^{m^2\times d}$, fix $\rho_m(i,j) = (i-1)m + j$ and define

$$[\mathrm{flat}(Y)]_{\rho_m(i,j)} = y_{ij}, \qquad [\mathrm{unflat}(Y^{\mathrm{flat}})]_{ij} = Y^{\mathrm{flat}}_{\rho_m(i,j)}. \tag{22}$$

Position codes. It is enough to choose any injective absolute-position map $\phi : [K] \to \mathbb{R}^{d_\phi}$ and to retain endpoint order through $\psi(s,t) = \phi(s) \,\|\, \phi(t)$. For the theory we use the canonical one-hot choice, where e_s is the s-th standard basis vector:

$$\phi(s) = e_s \in \mathbb{R}^K, \qquad \psi(s,t) = \phi(s) \,\|\, \phi(t). \tag{23}$$

Because $[K]$ is finite, every fixed injective ϕ is expressively equivalent: a biased ReLU MLP can map its finite image exactly to the one-hot codes. In practice, ϕ may therefore be one-hot, any fixed collision-free absolute encoding, or a learned lookup whose entries remain distinct. A symmetric pair code would generally lose orientation.

Practical encoder and task heads. The supplement's general encoder library uses the same complete ordered selection but a finite learned interface. Task adapters and preserved historical implementations are distinguished in Appendix L.4. With the canonical one-hot absolute code and the optional signed relative feature $\rho_K(t-s) = (t-s)/\max\{1, K-1\}$, it initializes the path stage as

$$\widehat{x}^{\mathrm{pr}}_s = W^{\mathrm{in}}_X[\widetilde{x}_{p_s}; e_s] + b^{\mathrm{in}}_X, \tag{24}$$

$$\widehat{y}^{\mathrm{pr}}_{st} = W^{\mathrm{in}}_Y[\widetilde{y}_{p_s p_t}; e_s; e_t; \rho_K(t-s); \mathbf{1}_{\{p_s=p_t\}}] + b^{\mathrm{in}}_Y. \tag{25}$$

For padded batches, a selected pair is valid exactly when both occurrences are valid. The appended bit $\mathbf{1}_{\{p_s=p_t\}}$ records vertex equality; the path-stage MLP separately receives $\mathbf{1}_{\{s=t\}}$, which records occurrence equality. For example, in $P = (u, v, u)$ the first bit is one at $(s,t) = (1,3)$ and the second is zero. Invalid occurrences and pairs are zeroed after projections and updates, and masked means divide by the number of valid occurrences. The positional codes and vertex-equality bit in Equation (25) are injected once and then treated as ordinary learned coordinates. This implementation preserves graph relabeling invariance and path order sensitivity; the theoretical interface is specified in Definition H.4.

The low-level `StrikeTwice` encoder returns graph and path states,

$$\mathrm{Enc}_\theta(X, Y, P) = (\widetilde{X}, \widetilde{Y}, Z, R), \qquad Z := X^{\mathrm{P}}, \quad R := Y^{\mathrm{P}}, \tag{26}$$

in a `PathEncoderOutput` object. In contrast, the public synthetic factory `build_synthetic_model` with identifier `TT` attaches a task-specific head and returns predictions. For scalar regression, let $k_{\max}$ be the padded path length and $m_s \in \{0,1\}$ the validity mask. The experiments use the exact masked mean

$$u = \frac{\sum_{s=1}^{k_{\max}} m_s Z_s}{\sum_{s=1}^{k_{\max}} m_s}, \qquad \widehat{y} = \mathrm{MLP}_{\mathrm{task}}(u), \tag{27}$$

Structured tasks may apply shared rowwise heads to Z_s and use the adjacent pair state $J_s = R_{s,s+1}$ for junction predictions. These task heads read the encoder outputs; the theoretical family instead uses the coordinate readout in Equation (17).

G.3 TOKENWISE NORMALIZATION

The practical reference implementation applies tokenwise feature normalization before every attention sublayer. It also normalizes the concatenation of all inputs to each aligned MLP. For $z \in \mathbb{R}^d$, let $\gamma, \beta \in \mathbb{R}^d$ be learned scale and shift vectors, $\varepsilon > 0$ a fixed stabilizer, and $\mathbf{1} \in \mathbb{R}^d$ the all-ones vector. LayerNorm is

$$\mathrm{LN}_{\gamma,\beta,\varepsilon}(z) = \gamma \odot \frac{z - \bar{z}\mathbf{1}}{\sqrt{d^{-1}\|z - \bar{z}\mathbf{1}\|_2^2 + \varepsilon}} + \beta, \qquad \bar{z} = d^{-1}\mathbf{1}^\top z, \tag{28}$$

while the common RMSNorm alternative is

$$\mathrm{RMSNorm}_{\gamma,\varepsilon}(z) = \gamma \odot \frac{z}{\sqrt{d^{-1}\|z\|_2^2 + \varepsilon}}. \tag{29}$$

We write Norm for the chosen normalization. In the practical implementation, the selector concatenates its position and graph-equality features once, the path input projection mixes them into an ordinary learned state z, and subsequent attention sublayers apply $\mathrm{Norm}(z)$. Likewise, the practical aligned pair branch applies $\mathrm{Norm}([y_{ij}; x_i; x_j; \mathbf{1}_{\{i=j\}}])$, rather than normalizing the four arguments separately.

For the theoretical envelope only, a token is viewed as learned state h together with fixed auxiliary coordinates τ, such as absolute-position or equality indicators, and we define

$$\mathrm{PNorm}(h; \tau) := \mathrm{Norm}(h) \parallel \tau. \tag{30}$$

For $z = h \parallel \tau$, abbreviate $\mathrm{PNorm}(z) := \mathrm{PNorm}(h; \tau)$. The normalization and fixed-tag conventions are specified in Definition H.4.

G.4 POINTWISE MLPS AND ALIGNED FUSION

For an element-wise activation σ, a depth-L pointwise MLP with learned matrices and biases $\theta = ((W_\ell, b_\ell))_{\ell=1}^L$ has the form

$$\mathrm{MLP}_\theta(z) = W_L\sigma(W_{L-1}\sigma(\cdots\sigma(W_1 z + b_1)\cdots) + b_{L-1}) + b_L. \tag{31}$$

The practical implementation uses GELU and applies one LayerNorm after concatenating the local inputs displayed below. For the theoretical interface of Definition H.4, write $z_a = h_a \parallel \tau_a$ and define

$$\mathrm{MLP}(z_1, \ldots, z_r; \tau) := \mathrm{MLP}_\theta(\mathrm{PNorm}(h_1; \tau_1) \parallel \cdots \parallel \mathrm{PNorm}(h_r; \tau_r) \parallel \tau), \tag{32}$$

where fixed arguments follow a semicolon and absent arguments are omitted. Each MLP in a residual block returns the width of its receiver stream. In AlMLP, the vertex and pair MLPs are independent; the two AlMLP occurrences in one EVT macro-layer are also independent unless parameters are explicitly tied.

G.5 FLATTENING PAIR STATES

The mutually inverse reshapes in Equation (22) use lexicographic order only to enumerate tokens, without adding positional information. Attention is equivariant to query-row permutations and invariant to simultaneous key/value-row permutations.

G.6 EXACT EVT MACRO-LAYER ORDER

With ℓ the macro-layer index, the two steps described in Appendix G.1 are, explicitly,

$$\bar{X}^\ell = X^\ell + \mathrm{Att}(X^\ell, X^\ell, X^\ell), \qquad \bar{Y}^\ell = Y^\ell + \mathrm{TriAtt}(Y^\ell), \tag{33}$$

$$(X^{\ell+\frac{1}{2}}, Y^{\ell+\frac{1}{2}}) = (\bar{X}^\ell, \bar{Y}^\ell) + \mathrm{AlMLP}(\bar{X}^\ell, \bar{Y}^\ell), \tag{34}$$

followed by

$$\bar{X}^{\ell+\frac{1}{2}} = X^{\ell+\frac{1}{2}} + \mathrm{Att}(X^{\ell+\frac{1}{2}}, \mathrm{flat}\, Y^{\ell+\frac{1}{2}}, \mathrm{flat}\, Y^{\ell+\frac{1}{2}}), \tag{35}$$

$$\bar{Y}^{\ell+\frac{1}{2}} = Y^{\ell+\frac{1}{2}} + \mathrm{unflat}\, \mathrm{Att}(\mathrm{flat}\, Y^{\ell+\frac{1}{2}}, X^{\ell+\frac{1}{2}}, X^{\ell+\frac{1}{2}}), \tag{36}$$

$$(X^{\ell+1}, Y^{\ell+1}) = (\bar{X}^{\ell+\frac{1}{2}}, \bar{Y}^{\ell+\frac{1}{2}}) + \mathrm{AlMLP}(\bar{X}^{\ell+\frac{1}{2}}, \bar{Y}^{\ell+\frac{1}{2}}). \tag{37}$$

The aligned updates use Equation (14), with independent parameters at each step.

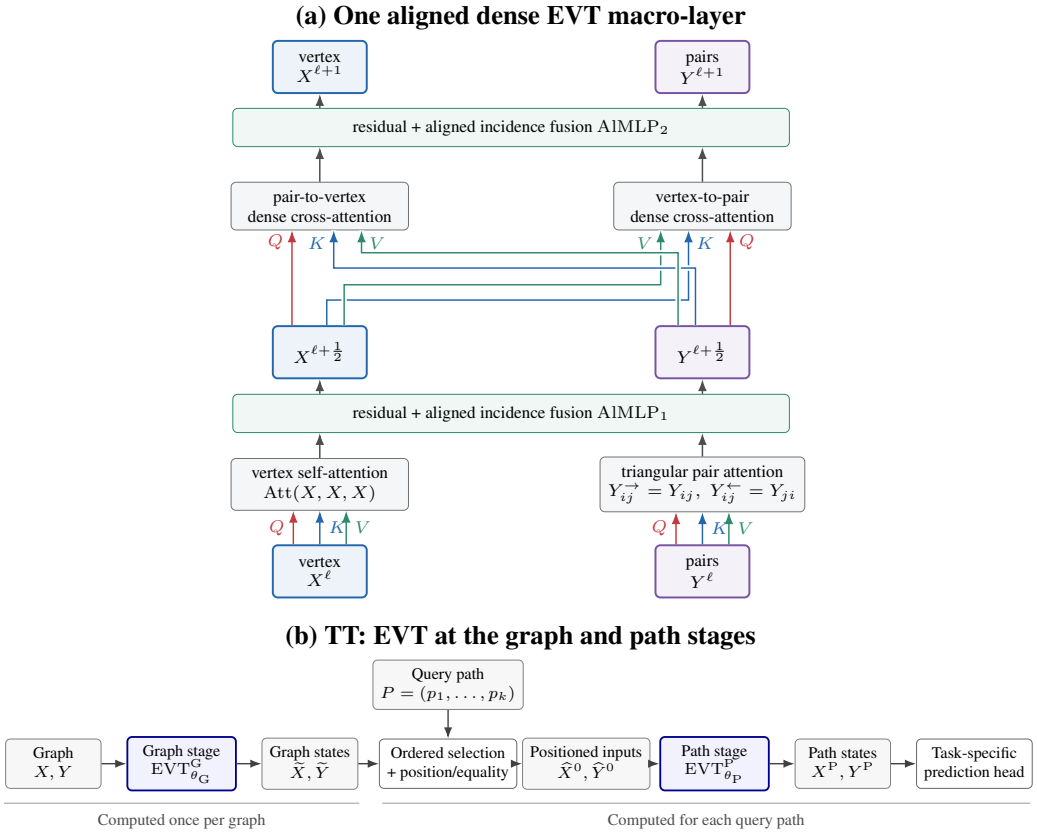


Figure G.1: Detailed EVT and TT architectures. **(a)** Vertex (blue) and pair (purple) streams update in parallel, fuse, exchange information by cross-attention, and fuse again. Red/blue/green arrows denote queries/keys/values; each sublayer is residual. **(b)** TT instantiates the selection branch of Figure 1b–c with independently parameterized EVT graph and path stages. Position and repeated-vertex indicators accompany the ordered selection.

G.7 Ordinary attention

Each displayed attention occurrence owns independent projections. For one head, write receiver tokens as $q_a = h_a^Q \parallel \tau_a^Q$, key tokens as $k_b = h_b^K \parallel \tau_b^K$, and value tokens as $v_b = h_b^V \parallel \tau_b^V$. Set $\widetilde{q}_a = \mathrm{PNorm}(h_a^Q; \tau_a^Q)$, $\widetilde{k}_b = \mathrm{PNorm}(h_b^K; \tau_b^K)$, and $\widetilde{v}_b = \mathrm{PNorm}(h_b^V; \tau_b^V)$. Here $Q = (q_a), K = (k_b), V = (v_b)$ denote token arrays, not the query count, path bound, or graph vertex set. The matrices W_Q, W_K project to query/key width d_a, and W_V projects values. Then

$$\mathrm{Att}(Q,K,V)_a = \sum_b \frac{\exp(\langle W_Q\widetilde{q}_a, W_K\widetilde{k}_b\rangle/\sqrt{d_a})}{\sum_c \exp(\langle W_Q\widetilde{q}_a, W_K\widetilde{k}_c\rangle/\sqrt{d_a})} W_V\widetilde{v}_b, \tag{38}$$

followed by concatenation of heads and an internal output matrix W_O. The operator Att returns only this increment; residual addition is explicit in Appendix G.6. Normalization follows Appendix G.3; for practical tokens, the fixed suffix τ is empty.

G.8 Triangular attention

For $m \times m$ pair states $Y = (y_{ij})$, where m is the current vertex or occurrence count, write $y_{ij} = h_{ij} \parallel \tau_{ij}$ and normalize each pair token once as $y_{ij}^{\mathrm{norm}} = \mathrm{PNorm}(h_{ij}; \tau_{ij})$. In agreement with Equation (1), set

$$y_{ij}^{\mathrm{norm},\rightarrow} := y_{ij}^{\mathrm{norm}}, \qquad y_{ij}^{\mathrm{norm},\leftarrow} := y_{ji}^{\mathrm{norm}}. \tag{39}$$

Each of the H heads $h \in [H]$ independently fixes a direction pair $(\alpha_h, \beta_h) \in \{\rightarrow, \leftarrow\}^2$ and defines

$$\begin{aligned} q_{ir}^h &= W_Q^h y_{ir}^{\mathrm{norm},\alpha_h}, & k_{rj}^h &= W_K^h y_{rj}^{\mathrm{norm},\beta_h}, \\ \ell_{irj}^h &= \langle q_{ir}^h, k_{rj}^h\rangle/\sqrt{d_a}, & v_{irj}^h &= (W_L^h y_{ir}^{\mathrm{norm},\alpha_h}) \odot (W_R^h y_{rj}^{\mathrm{norm},\beta_h}), \end{aligned} \tag{40}$$

and

$$T_{ij}^h = \sum_{r=1}^m \frac{e^{\ell_{irj}^h}}{\sum_{u=1}^m e^{\ell_{iuj}^h}} v_{irj}^h, \qquad \mathrm{TriAtt}(Y)_{ij} = W_O^{\triangle}[T_{ij}^1 \parallel \cdots \parallel T_{ij}^H]. \tag{41}$$

We write $\mathrm{TriAtt}^{\alpha,\beta}$ when all heads in a bank use the same pair and TriAtt for the general head-wise mixture. Because every head owns its denominator, the four direction classes are never forced to compete inside one softmax; their outputs interact only through $W_O^{\triangle}$ and subsequent pointwise maps. A layer may omit a class or assign several heads to it. For each intermediate index r, the value combines the same two oriented pair states used in the score. This joint value can encode the counts of paired colors used in the 2-FWL proof. With zero logits, a value depending on only one of these states summarizes only that state's distribution; it does not record which colors occur together. Values based on only one state, with nonconstant scores depending on both states, require a separate expressivity analysis. Like Att, TriAtt returns an increment and does not include the explicit residual connection.

H Detailed Expressivity Analysis

We first state the EVT–ET comparison and define the theoretical family. The proofs then establish this comparison, recover the complete selected profile from an occurrence state, and construct continuous readouts.

Supporting graph-stage results. Extending the ET analysis of Müller et al. (2024), direction-mixed EVT and direction-mixed ET with endpoint features share rooted distinguishing power and compact approximation closure (Proposition H.2). Reversed-pair access strictly extends the forward-tokenized chain-only restriction on directed inputs (Proposition J.1). Complementary finite-depth communication comparisons are given in Proposition H.3.

H.1 Information kernels and theoretical scope

Kernels as information boundaries. For a map f, let $\ker(f) := \{(u,v) : f(u) = f(v)\}$. For a family $\mathcal{F}$ of maps on a common domain, define its common kernel by

$$\mathrm{Ker}(\mathcal{F}) := \bigcap_{f\in\mathcal{F}} \ker(f) = \{(u,v) : f(u) = f(v) \text{ for every } f \in \mathcal{F}\}. \tag{42}$$

This equivalence relation consists exactly of the pairs that no member of the family can distinguish. An individual network may identify additional inputs; the common kernel records what the entire family can never separate. In the present length-preserving metric setting, every uniform limit of members of $\mathcal{F}$ must therefore take the same value on inputs that no family member distinguishes. Kernel equality with an explicit profile is the necessary information-level part of a universality theorem; the approximation statement below proves that it is also sufficient. For outputs in $\mathcal{O}_K$, equality is equality in the disjoint union, so different output lengths are distinct.

Why access to reversed pair states matters. If every head is restricted to the compositional variant $\mathrm{TriAtt}^{\rightarrow,\rightarrow}$, while pointwise pair maps continue to receive only y_{ij}, the corresponding refinement starts instead from

$$c_{ij}^{0,\mathrm{pure}} = (y_{ij}, x_i, x_j, \mathbf{1}_{\{i=j\}}). \tag{43}$$

For directed inputs this loses information: at rooted graph-stage pair readouts, the common kernel of the family allowing access to reversed pair states is strictly smaller than that of its pure restriction. In particular, on an explicit four-vertex directed example, two rooted pairs have the same stable pure color although the corresponding entries of $YY^\top$ differ. No member of either pure rooted ET/EVT family can therefore approximate even the coordinate map $(Y; i, j) \mapsto (YY^\top)_{ij}$ with uniform error below $1/2$ on that finite compact set. By contrast, on a scalar channel, one zero-logit $\mathrm{TriAtt}^{\rightarrow,\leftarrow}$ head computes $n^{-1}YY^\top$, and the bounded-size cardinality anchor lets the post-map recover the unnormalized product uniformly on compact sets. The strict kernel statement and witness are proved in Appendix J.1. The comparison concerns rooted graph-stage states, before path selection, with arbitrary head allocations.

Capacity and depth. Kernel equality guarantees that some finite family member separates each pair of distinct profiles. For each compact S, compatible target F, and $\varepsilon > 0$, Theorem 2 supplies a finite network whose resources may depend on S, F, ε, N, K. The theorem gives no quantitative bound on total capacity or guarantee of learning. The explicit construction uses the sufficient refinement bound $R_N = 2N^2$ before the additional setup, occurrence-decoding, gating, broadcast, and readout blocks. For the synthetic graph size $N = 28$, this bound alone is $1{,}568$ refinement stages, whereas the synthetic models use two graph and two path EVT macro-layers.

Consequences for EVT. The graph-stage proof also shows that direction-mixed EVT and the endpoint-lifted, direction-mixed pair-only ET have the same rooted common kernel—namely, the one induced by stable 2-FWL with bidirectional initialization—and the same uniform closure on every compact set. The native vertex stream therefore changes finite-depth and computational organization, not the unlimited-capacity information boundary. The precise rooted statement and complementary shallow comparisons are in Propositions H.2 and H.3.

The second triangular operator is also a resource choice. A positioned path decoder using pair MLPs and global pair-to-vertex attention can recover the complete selected profile without path triangular attention (Corollary H.9). This preserves the limiting function class but may require many auxiliary state coordinates.

Ordered selection and bounded sizes. Beyond the graph-level EVT–ET characterization, the theorem propagates the expressivity analysis through an ordered $P \times P$ selection and a second relational network. Shared parameters across bounded sizes are already covered by graph approximation frameworks (Keriven & Peyré, 2019; Azizian & Lelarge, 2021). Here, for fixed N, K, one parameter set acts on every component and, for each target, compact set, and accuracy, one network approximates simultaneously across all represented $n \le N$ and $k \le K$. Common safe depths, diagonal cardinality anchors, and one shared vector encoding of the finitely many colors make this possible without padding; see Appendix H.7.

Remark H.1 (Path-conditioned global refinement). Appendix I compares selected access, joint reading of the unmarked cache, and global refinement with occurrence marks, whose graph states depend on the query.

H.2 EVT–ET EXPRESSIVITY COMPARISON

Define the rooted pair and vertex domains

$$\mathcal{R}_N^Y := \bigsqcup_{n=1}^{N} \big(\mathbb{R}^{n\times d_X} \times \mathbb{R}^{n\times n\times d_Y} \times [n]^2\big), \qquad \mathcal{R}_N^X := \bigsqcup_{n=1}^{N} \big(\mathbb{R}^{n\times d_X} \times \mathbb{R}^{n\times n\times d_Y} \times [n]\big), \tag{44}$$

with their finite disjoint-union topologies. For $L \geq 0$, all four families below consist of maps to the same fixed space $\mathbb{R}^q$, with one parameter set shared over every $n \leq N$. The pair-only families are

$$\mathcal{E}_{N,L}^Y := \{(X, Y; i, j) \mapsto \pi(e_{ij}^L)\}, \quad \mathcal{E}_{N,L}^X := \{(X, Y; i) \mapsto \pi(e_{ii}^L)\}, \tag{45}$$

where e^0 is the endpoint tokenizer in Equation (47), each counted ET layer is one direction-mixed triangular-attention residual followed by one shared pairwise-MLP residual, and π selects designated coordinates. Each head is independently assigned its direction pair. The EVT families are

$$\mathcal{V}_{N,L}^Y := \{(X, Y; i, j) \mapsto \pi_Y(y_{ij}^L)\}, \quad \mathcal{V}_{N,L}^X := \{(X, Y; i) \mapsto \pi_X(x_i^L)\}, \tag{46}$$

where (X^L, Y^L) follows exactly L macro-layers Equations (33) to (37). The raw affine lifts and final coordinate projections are interfaces and do not count as layers. In particular, the ET tokenizer may fuse endpoint features before its first counted layer, whereas an EVT macro-layer used to write that fused token into pair workspace is counted.

For each rooted family, the lifts, MLPs, head dimensions, direction choices, normalization, and residual updates follow the architectural conventions of Definition H.4, with the input tokenizers and layer counts defined above. Unsubscripted families mean the arbitrary-depth unions $\mathcal{E}_N^\bullet = \bigcup_{L\geq 0} \mathcal{E}_{N,L}^\bullet$ and $\mathcal{V}_N^\bullet = \bigcup_{L\geq 0} \mathcal{V}_{N,L}^\bullet$.

The vertex stream does not raise the stable graph-information ceiling, but it can expose useful operations at a shallower or cheaper typed schedule. To compare it with a pair-only direction-mixed Edge Transformer, initialize ET by the pointwise endpoint lift used in prior ET analyses (Bergen et al., 2021; Müller et al., 2024):

$$e_{ij}^0 = \mathrm{Lift}(y_{ij}, x_i, x_j, \mathbf{1}_{\{i=j\}}, 1), \tag{47}$$

where Lift is an affine injection. An ET layer is a triangular-attention residual followed by a shared pairwise MLP. With this common interface, an EVT simulates an ET after one attention-disabled setup macro-layer writes the endpoint lift into pair workspace. Subsequent EVT blocks disable the vertex and cross branches and let the aligned pair MLP ignore its endpoint arguments. The depth shift disappears if Equation (47) is instead granted as common tokenization to both families; this is not the depth convention used below.

On a rooted pair input, set $\gamma_N^Y(X, Y; i, j) = c_{ij}^{R_N}$; on a rooted vertex input, set $\gamma_N^X(X, Y; i) = c_{ii}^{R_N}$.

Proposition H.2 (Rooted EVT–ET kernel and closure equivalence)**.** *For the rooted families above,*

$$\mathrm{Ker}(\mathcal{V}_N^Y) = \mathrm{Ker}(\mathcal{E}_N^Y) = \ker(\gamma_N^Y), \qquad \mathrm{Ker}(\mathcal{V}_N^X) = \mathrm{Ker}(\mathcal{E}_N^X) = \ker(\gamma_N^X). \tag{48}$$

Moreover, if S is a compact set of rooted pair or rooted vertex inputs and γ is the corresponding code, then the appropriate ET and EVT uniform closures both equal

$$\mathcal{C}_\gamma(S) := \{F \in C(S; \mathbb{R}^q) : \ker(\gamma|_S) \subseteq \ker(F)\}. \tag{49}$$

Proof sketch. The transpose-closure and stable-reconstruction lemma below shows that one pair color determines both endpoint colors, every oriented multiset over intermediate vertices, and the global pair-color multiset. Hence pair and vertex states factor through $c_{ij}^\star$ and $c_{ii}^\star$. Conversely, an oriented setup head exposes the reverse atomic state, after which triangular attention and finite-set MLP interpolation realize each 2-FWL refinement. Finite feature-bank compilation followed by Stone–Weierstrass gives the compact closure statement. The detailed construction, including one network shared over all $n \leq N$, is given in Appendix H.5. □

To isolate direct pair-to-vertex access, let $\mathcal{V}_{N,1}^{X,\mathrm{red}}$ use the following reduced typed schedule: one triangular pair residual and its pair MLP, followed by one global $Y \to X$ attention and its vertex MLP.

Vertex self-attention, the first vertex MLP, $X \to Y$ attention, and the final pair MLP are disabled. This is not a full EVT macro-layer. For the reduced comparison, the affine endpoint lift Equation (47) is common tokenization and costs no ET layer; one ET layer means exactly one triangular-attention residual followed by one pairwise-MLP residual. A reduced depth counts one execution of the displayed typed schedule.

Proposition H.3 (Finite-depth comparison)**.** *Writing a depth subscript on the rooted families,*

$$\mathcal{E}^{Y}_{N,L} \subseteq \mathcal{V}^{Y}_{N,L+1}, \qquad \mathcal{V}^{X,\mathrm{red}}_{N,1}|_S \subseteq \overline{\mathcal{E}^{X}_{N,3}|_S}^{\|\cdot\|_\infty} \quad \text{for every compact set } S \text{ of rooted vertex inputs.} \tag{50}$$

The depth-three bound is sharp for this reduced schedule: for $N \geq 7$, there is a two-point finite domain of rooted binary graphs on which a reduced one-layer EVT target has positive distance from every two-layer ET. On a second two-point rooted domain, a two-layer ET target separates the inputs whereas every reduced one-layer EVT agrees. Thus $\mathcal{V}^{X,\mathrm{red}}_{N,1}$ and $\mathcal{E}^{X}_{N,2}$ are incomparable after restriction to suitable finite domains.

Proof sketch. The first inclusion uses one attention-disabled EVT setup layer and then disables the extra branches. For the second, the three ET layers respectively compute the reduced pair update, row moments, and global moments at the diagonal root. A uniform separable approximation of the softmax kernel then recovers the pair-to-vertex update. Triangle density and two rooted leaves of a broom graph give the two strict directions. The layer schedule, uniform error argument, and finite domains are explicit in Appendix H.6. In particular, no three-layer compilation of a full EVT macro-layer is claimed. □

Ordinary vertex self-attention is not the source of the gap: two triangular heads with reversed query/value roles reproduce both endpoint updates exactly in one ET layer (Appendix H.6.2). EVT instead avoids storing and recomputing those shared endpoint results across all n^2 pair tokens.

H.3 Theoretical envelope

Definition H.4 (Theoretical cascade family)**.** The family $\mathcal{H}_{N,K}$ consists of the maps $\mathcal{D}_{\leq N,\leq K} \to \mathcal{O}_K$ obtained from Equations (3), (5) and (6) under the following conventions.

1. The graph and path stages have independent parameters and arbitrary finite macro-depths, vertex and pair widths, numbers of heads, and query/key and value dimensions. Every triangular head is independently assigned any $(\alpha_h, \beta_h) \in \{\to, \leftarrow\}^2$. Within each stage, one parameter set is shared over all its index sets and hence over every $n \leq N$ or $k \leq K$.
2. Raw vertex and pair interfaces are trainable injective affine lifts that append a constant coordinate and any finite number of zero workspace coordinates. At (i, j), the pair interface reads only y_{ij}; neither it nor AlMLP reads y_{ji} directly. The equality predicate is supplied to AlMLP as in Equation (14). Reverse-indexed tokens enter through $Y^{\leftarrow}$ pair states; receiver-local recovery of the reverse atomic state uses the diagonal-gated head in Equation (59). These interfaces, the selector, and the final fixed coordinate projection do not count as macro-layers. Each macro-layer is exactly Equations (33) to (37).
3. Every MLP is a biased ReLU network of arbitrary finite width and depth. Every displayed attention and MLP occurrence has independent parameters; its output may be identically zero, and designated workspace coordinates may be excluded from its residual increment.
4. Feature normalization is removed: every Norm is the identity. Fixed tags τ are retained outside the learned residual state; every residual increment is zero on their coordinates, and $\mathrm{PNorm}(h; \tau) = h \,\|\, \tau$. The selector uses $\phi(s) = e_s$ and $\psi(s, t) = \phi(s) \,\|\, \phi(t)$, and the output is read from q designated vertex coordinates after the path stage.

Thus a member of $\mathcal{H}_{N,K}$ is one map defined simultaneously on all bounded size components, not a collection of separately parameterized fixed-size networks. Any fixed injective ϕ on $[K]$ is equivalent in this envelope, because the finite interpolation primitive below decodes its image exactly to the one-hot tags.

The path-profile proof invokes Proposition H.2 as a graph-stage realization result: every first-stage EVT factors through stable 2-FWL colors, and one can be chosen to separate any finite collection of such colors. This is the only graph-stage expressivity fact required below.

H.4 Compilation primitives

Lemma H.5 (Compilation primitives and parallel groups of state coordinates)**.** *The families in Appendix H.2 and Definition H.4 have the following properties.*

1. *They can create constant and zero auxiliary coordinates, copy or protect designated coordinates exactly, disable any branch, pad depth with identity layers, and interpolate an arbitrary map exactly on a finite set of distinct token codes.*
2. *Any finite collection of networks from one of these families can be compiled into one wider network whose disjoint groups of coordinates reproduce all their designated states exactly. For cascades, graph-stage and path-stage depths are padded independently.*
3. *On a compact input set, one appended attention-disabled pointwise ReLU block can uniformly approximate any continuous function of the combined coordinates stored at the same rooted token.*

Proof. A biased ReLU MLP represents affine maps exactly by the identity $z = \mathrm{ReLU}(z) - \mathrm{ReLU}(-z)$. It therefore writes constants, copies coordinates, and leaves protected output rows at zero. To interpolate distinct codes $z_1, \dots, z_t$, choose a linear functional whose values on these codes are pairwise distinct and interpolate the desired outputs by a univariate continuous piecewise-affine function; its hinge representation is a finite biased ReLU network. Zeroing an attention output matrix, or both an MLP's final matrix and final bias, disables that residual. Zeroing every residual gives a legal identity layer.

For parallel compilation, first pad the component schedules to their maximum depths with these identity layers. Concatenate their input lifts and hidden states, assign each original attention head to its own group of coordinates, and make every output projection and pointwise MLP block diagonal. Thus each softmax is formed only from the queries and keys of its original network; neither residuals nor nonlinearities mix the coordinate groups. If an implementation requires a common head dimension D, zero-pad a head of original query dimension d_a and multiply its query projection by $\sqrt{D/d_a}$. Its score remains $\langle q, k\rangle/\sqrt{d_a}$ after division by $\sqrt{D}$; value coordinates are merely zero-padded. Hidden ReLU depths are equalized by identity layers on their nonnegative hidden activations. This constructs the joint state exactly, including when the original networks have different widths, head counts, and, for a cascade, graph and path depths.

Finally, the image of a compact input set under the combined state map is compact. Extend each coordinate of a continuous postcomposition from that image to an enclosing Euclidean box by the Tietze extension theorem, and apply the ReLU universal approximation theorem there. The resulting pointwise map is written into fresh zero coordinates by a final disabled-attention residual block. The exact direct sum is the architecture-specific closure needed by the usual compact relative-universality strategy for invariant and equivariant networks (Keriven & Peyré, 2019; Maron et al., 2019c; Azizian & Lelarge, 2021). □

H.5 Proof of the stable EVT–ET equivalence

Lemma H.6 (Uniform joint stabilization)**.** *Consider any two graph inputs of sizes $n, m \le N$, and compute their formal colors c^r and $\widetilde{c}^r$ by Equation (7). On the disjoint union $\Omega = [n]^2 \sqcup [m]^2$, let $\mathcal{P}_r$ be the partition whose blocks are equality classes of depth-r colors. Then $\mathcal{P}_{r+1}$ refines $\mathcal{P}_r$, and*

$$\mathcal{P}_{R_N} = \mathcal{P}_{R_N+1}, \qquad R_N = 2N^2. \tag{51}$$

Consequently, equality of two terminal colors, possibly from different graphs, implies equality of their terminal multiset over intermediate vertices:

$$c_{ij}^{R_N} = \widetilde{c}_{i'j'}^{R_N} \quad \Longrightarrow \quad \left\{\!\!\left\{ (c_{iu}^{R_N}, c_{uj}^{R_N}) : u \in [n] \right\}\!\!\right\} = \left\{\!\!\left\{ (\widetilde{c}_{i'v}^{R_N}, \widetilde{c}_{vj'}^{R_N}) : v \in [m] \right\}\!\!\right\}. \tag{52}$$

Proof. Because c_{ij}^{r+1} retains c_{ij}^r as its first component, equality at round $r+1$ implies equality at round r; hence $\mathcal{P}_{r+1}$ refines $\mathcal{P}_r$. The set Ω has $n^2 + m^2 \le 2N^2$ elements. Every strict refinement increases the number of blocks by at least one, so there can be at most $n^2 + m^2 - 1$ strict refinements. Once $\mathcal{P}_{r+1} = \mathcal{P}_r$, no later split is possible: on each $\mathcal{P}_r$-block the round-$r+1$ color is then constant, so replacing every round-r color in a multiset over intermediate vertices by its round-$r+1$ color is a

deterministic relabeling of partition classes. Thus the partition remains stationary. It follows that it is stationary by round $n^2 + m^2$, and therefore by the common round $R_N = 2N^2$.

If the colors on the left of Equation (52) are equal, their positions lie in one block of $\mathcal{P}_{R_N}$. Stationarity places them in one block of $\mathcal{P}_{R_N+1}$ as well. Equality of the literal round-$R_N + 1$ tuples then gives equality of their second components, which are exactly the two displayed multisets over intermediate vertices. □

Lemma H.7 (Transpose closure and stable-color reconstruction)**.** *For a terminal pair color $a = c^\star_{ij}$, the color a determines its reversed color $c^\star_{ji}$, its terminal multiset over intermediate vertices, the two endpoint diagonal colors, and the global multisets of terminal pair and diagonal colors.*

Proof. First define an involution on formal colors. At initialization,

$$\tau_0(y_{ij}, y_{ji}, x_i, x_j, \delta) := (y_{ji}, y_{ij}, x_j, x_i, \delta). \tag{53}$$

If a round-$r + 1$ color is written as (c, M), where $M = \{\{(a, b)\}\}$, set

$$\tau_{r+1}(c, M) := (\tau_r(c), \{\{(\tau_r(b), \tau_r(a)) : (a, b) \in M\}\}). \tag{54}$$

Induction in Equation (7) gives $c^r_{ji} = \tau_r(c^r_{ij})$: for each intermediate vertex, transposition reverses both ordered pairs and exchanges their positions in the tuple. Thus the terminal color determines its reversed color.

Let

$$M(a) := \left\{\left\{(c^\star_{ir}, c^\star_{rj}) : r \in [n]\right\}\right\}. \tag{55}$$

By Lemma H.6, $M(a)$ is well defined: the same formal color in any other graph instance has the same multiset over intermediate vertices. Its first and second projections give the row and column color multisets. More explicitly, let $\tau_\star := \tau_{R_N}$, $\epsilon(\rightarrow) = 0$, and $\epsilon(\leftarrow) = 1$. The multiset of paired colors read by direction pair (α, β) is

$$M^{\alpha,\beta}(a) = \left\{\left\{(\tau_\star^{\epsilon(\alpha)}(b_1), \tau_\star^{\epsilon(\beta)}(b_2)) : (b_1, b_2) \in M(a)\right\}\right\}. \tag{56}$$

Thus all four variants remain functions of one canonical multiset over intermediate vertices, with the two colors in each tuple associated with the same vertex. In $M(a)$, exactly one tuple has a diagonal first component, which identifies $c^\star_{ii}$, and exactly one has a diagonal second component, which identifies $c^\star_{jj}$. The global pair-color multiset is also recovered: the first projection of $M(c^\star_{ij})$ gives $\{\{c^\star_{ir} : r \in [n]\}\}$; for every recovered color $a = c^\star_{ir}$, the second projection of the well-defined $M(a)$ is $\{\{c^\star_{ur} : u \in [n]\}\}$. Their union with multiplicity over r is the global pair-color multiset, and its diagonal submultiset is then also known. □

Consequently, ordinary vertex attention, all four oriented triangular variants, both cross-attentions, and both aligned MLPs are functions of the receiver stable color and stable global multisets. Indeed, the reversed pair state at (r, i) is a function of $c^\star_{ri} = \tau_{R_N}(c^\star_{ir})$; the same argument applies when the other pair state read by the head is reversed. Induction over layers proves

$$x^\ell_i = a_\ell(c^\star_{ii}), \qquad y^\ell_{ij} = b_\ell(c^\star_{ij}). \tag{57}$$

For separation on a finite pair of inputs or finite set S, first make the bidirectional initial color locally available without changing AlMLP. In EVT, one attention-disabled setup macro-layer uses the original aligned pair MLP to store the forward atomic token

$$a_{ij} := (y_{ij}, x_i, x_j, \mathbf{1}_{\{i=j\}}, 1) \tag{58}$$

in protected pair workspace. Pair-only ET already has this token through Equation (47). In the next triangular layer, at receiver (i, j), a zero-logit $\mathrm{TriAtt}^{\leftarrow,\rightarrow}$ head reads the state at (u, i) and multiplies it by the diagonal indicator from (u, j), retaining only the term with $u = j$:

$$g_{ij} = \frac{1}{n} \sum_{u=1}^{n} a_{ui} \mathbf{1}_{\{u=j\}} = \frac{1}{n} a_{ji}. \tag{59}$$

The Hadamard value in Equation (40) implements the displayed gate, and the head writes into fresh workspace. The following original aligned pair MLP sees (a_{ij}, g_{ij}). Its last coordinate is $1/n$, so

finite-set interpolation recovers a_{ji} and maps every initial color occurring in S exactly to a linearly independent basis code $\eta_0(c)$. The ET pairwise MLP performs the same step. Uniformly on compact sets, division by $1/n \geq 1/N$ is a continuous pointwise operation and can be approximated arbitrarily well. Thus reverse access is realized by an oriented setup head without changing the pointwise pair interface.

At round r, suppose the state carries the corresponding basis code $\eta_r(c)$. In a triangular head choose zero queries and keys, and projections L, R such that

$$[L\eta_r(a)]_{a'b'} = \mathbf{1}_{\{a'=a\}}, \qquad [R\eta_r(b)]_{a'b'} = \mathbf{1}_{\{b'=b\}}, \qquad L\eta_r(a) \odot R\eta_r(b) = e_{(a,b)}. \tag{60}$$

Then the head returns

$$h^r_{ij} = \frac{1}{n}\sum_{u=1}^{n} e_{(c^r_{iu}, c^r_{uj})}. \tag{61}$$

The normalized histogram is injective jointly over all $n \leq N$. Indeed, let $D(c) \in \{0,1\}$ denote the initial diagonal-bit component retained by the color c. Then $D(c^r_{iu}) = 1$ singles out exactly $u = i$, so

$$\lambda(h) := \sum_{(a,b):D(a)=1} h_{ab} = \frac{1}{n}, \qquad \mathrm{mult}(a,b) = \frac{h_{ab}}{\lambda(h)}. \tag{62}$$

A finite-set ReLU MLP maps $(\eta_r(c^r_{ij}), h^r_{ij})$ to $\eta_{r+1}(c^{r+1}_{ij})$. One shared vector encoding handles the union of all colors appearing in the chosen pair or finite set, across all its size components. The resulting network still uses one parameter set on the whole bounded domain. This proves separation for the rooted ET pair family; diagonal selection gives the rooted vertex family, while the aligned pair MLP writes the code into EVT pair workspace and the aligned vertex MLP copies its diagonal instance into the EVT vertex stream. Together with Equation (57), it proves the kernel part of Proposition H.2.

Let S be a compact rooted input set. By the kernel equality, the relation $R := \ker(\gamma|_S)$ is also an intersection of kernels of continuous network maps, hence is closed. The quotient $Q = S/R$ is therefore compact Hausdorff. Network coordinates descend to continuous functions on Q and separate its points by the kernel equality. Let $\mathcal{A}$ be their scalar uniform closure. By Lemma H.5, constants belong to $\mathcal{A}$, and any two representatives can be evaluated exactly in separate coordinates at the same token before a terminal MLP approximates their sum or product. Hence $\mathcal{A}$ is a closed unital separating subalgebra of $C(Q)$. Stone–Weierstrass gives $\mathcal{A} = C(Q)$, and parallel compilation of the q coordinates proves Equation (49) for both ET and EVT. This quotient argument follows the standard relative-universality viewpoint for invariant networks (Keriven & Peyré, 2019; Maron et al., 2019c; Azizian & Lelarge, 2021); the exact parallel construction is supplied here by Lemma H.5.

H.6 Finite-depth comparison

At finite depth, two comparisons illustrate how the typed architecture changes access to vertex and pair states. Throughout this subsection, the affine endpoint lift is depth zero, every numbered ET layer contains one triangular-attention residual and its pairwise-MLP residual, and the reduced one-layer family follows exactly the pair-update–then–$Y \to X$ schedule stated before Proposition H.3. No unnumbered setup MLP is used. Both lower-bound domains use symmetric adjacency tensors and the same ordered endpoint tokenizer. Their bidirectional atomic color is therefore a deterministic relabeling of the forward color. The finite-round version of the transpose recursion Equations (53) and (54) then makes every oriented multiset over intermediate vertices at round r a function of the canonical compositional multiset over intermediate vertices. By induction, an L-layer direction-mixed ET state still factors through the round-L color used in the lower bounds. The reduced broom comparison has only one triangular pair layer, for which raw symmetry already suffices.

H.6.1 A three-layer upper bound for a reduced EVT readout

Fix a reduced readout and one of its pair-to-vertex heads; finitely many heads are evaluated in parallel using separate groups of state coordinates. Write z_{uv} for the pair state after the reduced triangular residual and pair MLP, and

$$q_i = Qx_i, \qquad k_{uv} = Kz_{uv}, \qquad v_{uv} = Vz_{uv}, \qquad \kappa(q,k) = \exp(\langle q,k\rangle/\sqrt{d_a}). \tag{63}$$

The exact cross-attention head is $A_i = N_i/D_i$, where

$$D_i = \frac{1}{n^2}\sum_{u,v}\kappa(q_i, k_{uv}), \qquad N_i = \frac{1}{n^2}\sum_{u,v}\kappa(q_i, k_{uv})v_{uv}. \tag{64}$$

The factor n^{-2} cancels in the quotient.

All queries, keys, and values reached from a compact rooted domain S, over its finitely many size components, lie in common compact sets Q_S, K_S, V_S. Hence for any $\eta > 0$ there are continuous scalar functions a_h, b_h such that

$$\sup_{Q_S\times K_S}\left|\kappa(q,k) - \sum_{h=1}^{H} a_h(q)b_h(k)\right| < \eta. \tag{65}$$

Define

$$B_h = \frac{1}{n^2}\sum_{u,v} b_h(k_{uv}), \qquad C_h = \frac{1}{n^2}\sum_{u,v} b_h(k_{uv})v_{uv}. \tag{66}$$

The following schedule computes approximations of these moments. By Lemma H.5, every quantity listed as protected occupies fresh residual coordinates. The triangular value is an elementwise product of two projected pair states; choosing one projection to return the constant-one channel passes the other projected state unchanged.

ET layer	Triangular heads	Pairwise MLP	Protected after the layer
1	Reproduce the reduced triangular pair residual in separate coordinates.	Reproduce its pair MLP exactly and, with extra hidden units, approximate $b_h(k_{uv})$ and $b_h(k_{uv})v_{uv}$ for every h.	z_{uv}, k_{uv}, v_{uv}, endpoint data and q_u, the root data (x_u, z_{uu}) on diagonal tokens, moments, and 1.
2	With zero scores, read the moment coordinates from the state at (u, r) and 1 from the state at (r, j). At every (u, j) this gives the row means $n^{-1}\sum_r b_h(k_{ur})$ and $n^{-1}\sum_r b_h(k_{ur})v_{ur}$.	Copy the row means into fresh channels; leave the query and root coordinates unchanged.	Row moments, q_u, endpoint data, and the diagonal root data.
3	With zero scores, read 1 from (i, r) and the row moments indexed by r from (r, j). Their averages are B_h, C_h at every (i, j).	At (i, i), approximate the quotient in Equation (67) and the complete reduced vertex residual and readout.	The designated diagonal output.

Table H.1: Three ET layers compiling the reduced pair-update–then–$Y \to X$ schedule. "Protected" means that subsequent attention output projections and residual MLP increments write zero into those coordinates.

The first MLP can reproduce the given pair MLP and approximate the additional moment maps simultaneously by concatenating hidden units. Thus nonlinear products $b_h(k)v$ require no extra layer. The last MLP sees q_i, the preserved reduced root state, and all global moments, and approximates

$$\frac{\sum_h a_h(q_i)C_h}{\sum_h a_h(q_i)B_h}. \tag{67}$$

It can therefore also apply the reduced vertex MLP in the same pointwise block.

For completeness, let $M_V = \sup_{V_S}\|v\|$, $d_0 = \min_{Q_S\times K_S}\kappa > 0$, and $A_0 = \sum_h \sup_{Q_S}|a_h|$. If the first MLP approximates each stored moment integrand within δ, the two normalized aggregations do not amplify that error. Consequently the final denominator and numerator differ from D_i, N_i by at most $\eta + A_0\delta$ and $M_V\eta + A_0\delta$, respectively, uniformly in $n \le N$ and in S. Choose $\eta + A_0\delta < d_0/2$; the approximate denominator is then at least $d_0/2$, so the quotient is uniformly stable. Uniform

approximation of the final continuous pointwise map uses the remaining error budget. More explicitly, for a requested error ε, first choose η, δ so that uniform continuity of the quotient and reduced readout makes their propagated error below $\varepsilon/2$, then choose the last MLP approximation below $\varepsilon/2$. This proves Equation (50). Width and head count may depend on S and the requested accuracy, but the three ET layers in Table H.1 do not; the result is neither an exact parameter identity nor a compilation of a full EVT macro-layer.

H.6.2 Vertex attention inside triangular attention

For one vertex head

$$A_i(X) = \sum_t \mathrm{softmax}_t \left(\langle Qx_i, Kx_t \rangle / \sqrt{d_a} \right) Vx_t, \tag{68}$$

the endpoint lift supplies x_i, x_t and a constant. In the first endpoint-simulation head choose

$$q_{it} = Qx_i, \quad k_{tj} = Kx_t, \quad v^{\mathrm{L}}_{it} = 1, \quad v^{\mathrm{R}}_{tj} = Vx_t. \tag{69}$$

Its output is $A_i(X)$, independent of j. Reversing the query/key and value roles means using a second head with

$$q_{it} = Kx_t, \quad k_{tj} = Qx_j, \quad v^{\mathrm{L}}_{it} = Vx_t, \quad v^{\mathrm{R}}_{tj} = 1. \tag{70}$$

Its output is $A_j(X)$, independent of i. The pair-only ET therefore implements both endpoint updates in one triangular layer, with two groups of heads but no depth depending on n. Its standard execution nevertheless forms n^3 scores indexed by the two endpoints and an intermediate vertex, and stores the endpoint results redundantly over n^2 pairs, whereas native vertex attention forms n^2 scores and stores n vertex states.

H.6.3 Sharpness and the reverse shallow gap

Taking a scalar designated output, we use the same explicit binary encoding in both witnesses. For a simple undirected graph G, set every vertex feature to $x_i = 1$, set $y_{ij} = A^G_{ij}$, and retain the diagonal bit supplied by Equation (47). Thus diagonal, edge, and nonedge tokens have the three distinct coordinate pairs $(A_{ij}, \mathbf{1}_{\{i=j\}}) = (0,1), (1,0), (0,0)$; the attention heads below read these coordinates directly, so no setup layer or finite-code MLP is hidden in the depth count. Write $\mathsf{I}(G, r)$ for this input with designated vertex root r.

For the first domain, let

$$G_0 = \{r\} \sqcup C_6, \qquad G_1 = \{r\} \sqcup (C_3 \sqcup C_3), \qquad S_\triangle = \{\mathsf{I}(G_0, r), \mathsf{I}(G_1, r)\}. \tag{71}$$

Both graphs have seven vertices. A uniform triangular head followed by its pair MLP computes

$$h_{ij} = \frac{1}{n} \sum_u A_{iu} A_{uj}, \qquad z_{ij} = A_{ij} h_{ij}. \tag{72}$$

The pair MLP gates by the binary variable A_{ij} exactly: for $0 \le h \le 1$, $Ah = \mathrm{ReLU}(h) - \mathrm{ReLU}(h - A)$. Uniform pair-to-vertex attention then broadcasts, at every root,

$$T_i(A) = \frac{1}{n^2} \sum_{i',j'} z_{i'j'} = \frac{\mathrm{tr}(A^3)}{n^3}. \tag{73}$$

Uniform $Y \to X$ attention gives Equation (73), hence $T|_{S_\triangle} \in \mathcal{V}^{X,\mathrm{red}}_{N,1}|_{S_\triangle}$.

For either graph and every nonroot u, the one-round colors (c^1_{ru}, c^1_{ur}) depend only on the facts that r is isolated and u has degree two. The root term is also identical. Therefore the multiset over intermediate vertices defining c^2_{rr} agree, so $c^2_{rr}(G_0) = c^2_{rr}(G_1)$. Induction over the two ET layers shows that every diagonal two-layer ET readout factors through this color and hence agrees on $S_\triangle$. Yet G_0 has no triangle while G_1 has two, so $T_r(G_0) = 0$ and $T_r(G_1) = 12/7^3$. Consequently

$$\inf_{E \in \mathcal{E}^X_{N,2}} \|T - E\|_{\infty, S_\triangle} \ge \frac{6}{7^3}. \tag{74}$$

For the reverse domain, let the connected broom be $G_{\mathrm{B}} = (V_{\mathrm{B}}, E_{\mathrm{B}})$, with $V_{\mathrm{B}} = \{a, b, c, v, w\}$ and

$$E_{\mathrm{B}} = \{\{v, a\}, \{v, b\}, \{v, w\}, \{w, c\}\}. \tag{75}$$

Set $S_{\mathrm{B}} = \{\mathsf{I}(G_{\mathrm{B}}, a), \mathsf{I}(G_{\mathrm{B}}, c)\}$. The leaf roots a, c have the same one-round diagonal color. Moreover, the unrooted input is the same in both elements of S_{B}, so its global multiset of one-round pair colors is identical. In any reduced readout, the pair phase is a function of the one-round pair color; the constant vertex query then turns global $Y \to X$ attention into a function of that global multiset; and the last vertex MLP additionally sees only the rooted self-pair state. Every member of $\mathcal{V}_{N,1}^{X,\mathrm{red}}$ therefore has the same value at a and c.

By contrast, the target

$$F_i = \sum_j A_{ij} \deg(j) \tag{76}$$

belongs to $\mathcal{E}_{N,2}^{X}|_{S_{\mathrm{B}}}$ under the stated depth convention. Indeed, a zero-score head in layer one stores $n^{-1}\sum_r A_{ur} = \deg(u)/n$ at every pair (u, v). At the diagonal (i, i), a zero-score layer-two head multiplies A_{ij} from the state at (i, j) by $\deg(j)/n$ from the state at (j, i), and averages over j, yielding F_i/n^2. Since $n = 5$ on S_{B}, the same layer's final linear MLP multiplies by 25. Thus no preliminary coding layer is used. Finally, $F_a = \deg(v) = 3$ and $F_c = \deg(w) = 2$, and hence

$$F|_{S_{\mathrm{B}}} \in \mathcal{E}_{N,2}^{X}|_{S_{\mathrm{B}}} \setminus \mathcal{V}_{N,1}^{X,\mathrm{red}}|_{S_{\mathrm{B}}}, \tag{77}$$

proving the reverse non-inclusion.

The two finite domains therefore isolate opposite shallow advantages without asserting an ordering between full macro-layer classes. Together with Table H.1, they prove the sharpness and incomparability claims in Proposition H.3.

H.7 Proof of the path-profile theorem

We first show that the positioned second EVT loses no information from the selected matrix. Define the symbolic initial occurrence color

$$d_{st}^{0} := (c^{\star}_{p_s p_t}, \phi(s), \phi(t), \mathbf{1}_{\{s=t\}}). \tag{78}$$

On a two-input comparison or finite set, a preparatory macro-layer disables attention and uses the biased aligned pair MLP to write a basis code $\eta_0(d_{st}^0)$ into pair workspace. Subsequent triangular layers realize the same symbolic 2-FWL recursion on the occurrence set.

Lemma H.8 (Two-round recovery of the positioned selection)**.** *For injective ϕ, every color d_{st}^2 determines the complete indexed matrix $(d_{uv}^0)_{u,v\in[k]}$ and the root indices (s, t).*

Proof. The multiset over intermediate vertices in d_{st}^2 contains (d_{su}^1, d_{ut}^1) for every u. Each first component retains $\phi(u)$, so the multiset is indexed and yields every d_{su}^1. For fixed u, the multiset over intermediate vertices inside d_{su}^1 contains (d_{sv}^0, d_{vu}^0) for every v. Its retained $\phi(v)$ indexes these entries and recovers d_{vu}^0 for all v. Varying u recovers the full matrix. Retained root tags recover s, t. □

By Proposition H.2, every graph-stage EVT vertex and pair state factors through its rooted stable 2-FWL color. Hence equal $\Psi_{N,K}$ makes all selected first-stage states equal, and deterministic second-stage updates preserve equality. Conversely, the finite simultaneous realization used to prove Proposition H.2 supplies one first-stage EVT that assigns basis codes to every stable graph color in the two compared inputs. The selected d^0 arrays then differ exactly when the profiles differ. The occurrence version of Equation (62), using the occurrence diagonal predicate, recovers k from a uniform triangular head. Two refinement layers realize the code of Lemma H.8. The aligned vertex MLP reads the self-pair state d_{ss}^2, copies its profile code into X_s, and writes a separating scalar into a designated vertex-output coordinate. Different k already gives different output components.

The same construction works with one parameter set over all bounded sizes: the graph code uses the common terminal depth $R_N = 2N^2$ justified by Lemma H.6, while the positioned occurrence decoder uses exactly the two refinements of Lemma H.8, uniformly for every $k \le K$. Finite-set

MLPs are defined on the union of all codes occurring in the comparison. No padding or explicit size embedding is needed. If the diagonal flags are removed, however, normalized attention no longer recovers multiplicity across sizes; one must then add size tags, sum attention, or validity-aware padding.

H.8 Universality proof

For finite S, the compact-density statement strengthens to exact interpolation:

$$\{H|_S : H \in \mathcal{H}_{N,K}\} = \mathcal{C}_\Psi(S). \tag{79}$$

Indeed, apply the finite simultaneous realization from Proposition H.2 to all stable graph colors occurring in S. By Lemma H.8, every terminal vertex root then has an injective code for $(\Psi_{N,K}(u), s)$. Since $F \in \mathcal{C}_\Psi(S)$ assigns the same output to inputs with the same profile, a shared ReLU MLP conditioned on $\phi(s)$ exactly interpolates $F(u)_s$. Necessity follows from the kernel identity proved in the preceding subsection.

For compact S, fix $F \in \mathcal{C}_\Psi(S)$ and $\varepsilon > 0$. For each length k, set

$$E_{\varepsilon,k} := \{(u,v) \in S_k^2 : \|F(u) - F(v)\|_{\mathrm{F}} \geq \varepsilon/2\}, \qquad S_k := S \cap \bigsqcup_{n \leq N} \mathcal{D}_{n,k}. \tag{80}$$

The finite union $E_\varepsilon = \bigcup_k E_{\varepsilon,k}$ is compact and disjoint from $\mathrm{Ker}(\mathcal{H}_{N,K})$. The open separator sets $\{(u,v) : H(u) \neq H(v)\}$, $H \in \mathcal{H}_{N,K}$, cover it; select a finite subcover $H_1, \ldots, H_m$, and append the zero output H_0 so the joint array shape records length. Let $A = (H_0, \ldots, H_m)$. Among inputs with the same value of A, the outputs of F differ by less than $\varepsilon/2$ in Frobenius norm.

The image $A(S)$ is compact. For each $a \in A(S)$, choose a representative $u_a \in A^{-1}(a)$, and set

$$U_a := \{u \in S_{k(a)} : \|F(u) - F(u_a)\|_{\mathrm{F}} < \varepsilon/2\}, \qquad V_a := A(S) \setminus A(S \setminus U_a). \tag{81}$$

Here $k(a)$ is the length recorded by the joint array shape. The bound on the variation of F within each preimage gives $A^{-1}(a) \subset U_a$. Since $S \setminus U_a$ is compact, V_a is an open neighborhood of a, and $A^{-1}(V_a) \subset U_a$. On each open-and-closed length component $A(S_k)$, choose a finite partition of unity subordinate to selected V_{a_j} with $k(a_j) = k$. These partitions define the continuous, length-preserving map $R(b) = \sum_{j:k(a_j)=k(b)} \rho_j(b) F(u_{a_j})$ on $A(S)$. Every sum thus combines matrices of the same size, and

$$\|F - R \circ A\|_{\infty,S} < \varepsilon/2. \tag{82}$$

By Lemma H.5, one wider cascade simulates the finite networks H_j exactly in disjoint groups of coordinates after padding the graph and path depths independently. Write $A_s = (H_0(u)_s, \ldots, H_m(u)_s)$. Compactness gives an $M \geq 1$ such that every coordinate z of every A_s, over $u \in S$, satisfies $|z| \leq M$. For $g \in \{0,1\}$, the biased ReLU gate

$$G_M(z,g) := \mathrm{ReLU}(z + Mg) - \mathrm{ReLU}(z - Mg) - Mg \tag{83}$$

satisfies $G_M(z,0) = 0$ and $G_M(z,1) = z$ throughout this interval. A terminal path MLP applies this gate coordinatewise with $g = [\phi(s)]_t$, thereby placing $(1, A_s)$ exactly in block $t = s$ of a fixed K-block workspace and zero in every other block. No product primitive is being assumed. A zero-query, zero-key vertex-attention head then broadcasts

$$B_k(A) := \frac{1}{k} \sum_{s=1}^{k} e_s \otimes (1, A_s) \tag{84}$$

to every occurrence. The leading occupancy coordinate in each block identifies the active positions and the factor $1/k$, so B_k is an injective continuous embedding of the finite disjoint union of joint output arrays: its nonzero blocks recover k, and division of each data block by its occupancy recovers A_s. A shared terminal MLP sees $(B_k(A), \phi(s))$ and uniformly approximates $R(A)_s$. More precisely, on the compact row-code set

$$\mathcal{B} := \{(B_{k(u)}(A(u)), \phi(s)) : u \in S,\ s \in [k(u)]\}, \tag{85}$$

define $g(B_{k(u)}(A(u)), \phi(s)) := R(A(u))_s$. Injectivity of B_k makes g well defined and continuous. Extend its coordinates by Tietze to a compact Euclidean neighborhood of $\mathcal{B}$, and approximate that extension by the terminal MLP with Euclidean error below $\varepsilon/(2\sqrt{K})$ per output row. The resulting matrix error is at most $\sqrt{k(u)}\,\varepsilon/(2\sqrt{K}) \leq \varepsilon/2$ in Frobenius norm. Together with Equation (82), the total error is below ε. Uniform limits remain continuous and length preserving, and still assign the same output to inputs with the same profile, giving the reverse inclusion.

H.9 A POSITIONED DECODER WITHOUT PATH TRIANGULAR ATTENTION

Let $\mathcal{H}^{\mathrm{dec}}_{N,K} \subseteq \mathcal{H}_{N,K}$ be the subfamily in which every triangular-attention increment in the path stage is identically zero. All other operations and conventions remain those of Definition H.4.

Corollary H.9 (Profile-preserving positioned decoder)**.** *On the full bounded-size domain,*

$$\mathrm{Ker}(\mathcal{H}^{\mathrm{dec}}_{N,K}) = \ker(\Psi_{N,K}). \tag{86}$$

For every compact $S \subset \mathcal{D}_{\le N,\le K}$,

$$\overline{\{H|_S : H \in \mathcal{H}^{\mathrm{dec}}_{N,K}\}}^{\|\cdot\|_{\infty,S}} = \mathcal{C}_\Psi(S). \tag{87}$$

For finite S, every target in $\mathcal{C}_\Psi(S)$ can be interpolated exactly with one path-stage macro-layer after a suitable graph-stage encoder.

Proof. Fix a finite set S. The simultaneous realization in Appendix H.5 supplies one graph-stage EVT carrying a distinct basis code $\eta(c) \in \mathbb{R}^d$ for every terminal graph color occurring in S. The injective selector preserves these codes together with the ordered tags e_s, e_t. With within-stream attention disabled, the first aligned pair MLP of a path macro-layer writes into fresh workspace the K^2-block vector

$$U_{st} = e_s \otimes e_t \otimes \big(1, \eta(c^\star_{p_s p_t})\big). \tag{88}$$

This is a prescribed map on finitely many distinct local codes and tags, so Lemma H.5 realizes it exactly by a shared ReLU MLP. All unused residual outputs are zero.

The following global $Y \to X$ attention in the same macro-layer reads these half-step pair states, as specified in Equation (35). Set its queries and keys to zero and let its value projection select U_{st}. Writing into fresh vertex coordinates, it broadcasts to every occurrence

$$B = \frac{1}{k^2} \sum_{s,t=1}^{k} e_s \otimes e_t \otimes \big(1, \eta(c^\star_{p_s p_t})\big). \tag{89}$$

Each active block has occupancy $1/k^2 \ge 1/K^2$, while inactive blocks are zero. Thus B determines k and the complete indexed matrix of codes. Each terminal color also determines n, since it retains the cardinality of its first-round multiset over intermediate vertices. Hence B determines $\Psi_{N,K}$ on this finite set. The last aligned vertex MLP receives B and the protected root tag $\phi(s)$; finite interpolation writes any prescribed compatible row $F(u)_s$ into the output coordinates. No path triangular-attention head is used in this construction.

Applying the construction to two inputs separates unequal profiles of the same length. Different lengths already give different output components. The reverse kernel inclusion follows from $\mathcal{H}^{\mathrm{dec}}_{N,K} \subseteq \mathcal{H}_{N,K}$ and Theorem 2, proving Equation (86).

For compact S, the compact argument of Appendix H.8 applies within this subfamily. Indeed, the kernel identity supplies a finite family of separators from $\mathcal{H}^{\mathrm{dec}}_{N,K}$ for Equation (80); the construction of neighborhoods of the preimages, followed by a partition of unity, gives Equation (82). Constants, identity padding, and exact parallel channel compilation preserve the condition that every path triangular increment is zero. Finally, the positioned readout in Equations (83) to (85) uses only path MLPs and vertex self-attention, which remain available. It therefore produces the same arbitrarily accurate uniform approximation within $\mathcal{H}^{\mathrm{dec}}_{N,K}$. The opposite closure inclusion follows from the subfamily inclusion and Theorem 2. □

The one-macro-layer statement concerns finite-set interpolation after a suitable graph encoder. The compact construction may require additional path layers to combine separate groups of features and apply the terminal readout.

Auxiliary width and experimental budgets. The explicit auxiliary state in Equation (88) has $K^2(1+d)$ coordinates, where d is the dimension of the color code η chosen for the finite set S, rather than the experimental hidden width. For illustration, setting $K = 7$ and $d = 32$ gives $7^2(1+32) = 1{,}617$ auxiliary coordinates, about 50.5 times width 32. This counts only the displayed workspace, in addition to retained states and tags; the interpolation MLPs also require capacity. It

is a sufficient allocation for this construction, not a minimal-width lower bound for all positioned decoders. The evaluated Task 1 TT and TD models use width 32, FFN widths 128 and 130, and 312,753 and 312,457 parameters, respectively (Appendix D.2); they do not instantiate this auxiliary workspace.

I Selection, Joint Reading, and Query Marking

For occurrence-indexed access to an unmarked pair cache, we compare the selected profile $\Psi_{N,K}$ with joint reading of the same unmarked cache and with refinement conditioned on the query. Use the common terminal depth $R_N = 2N^2$ and define occurrence marks by

$$m_P(i) := \sum_{s:p_s=i} e_s \in \mathbb{R}^K, \tag{90}$$

$$\bar{c}^0_{ij} = (y_{ij}, y_{ji}, x_i, x_j, \mathbf{1}_{\{i=j\}}, m_P(i), m_P(j)). \tag{91}$$

Writing the resulting terminal marked colors as $\bar{c}^\star$, set

$$\Psi^{\mathrm{mark}}_{N,K}(X,Y,P) := \left(n, k, (\bar{c}^\star_{p_s p_s})_{s\in[k]}\right), \tag{92}$$

$$\Gamma_{N,K}(X,Y,P) := \left(n, k, \left\{\!\left\{(c^\star_{p_1 u}, \dots, c^\star_{p_k u}) : u \in [n]\right\}\!\right\}\right). \tag{93}$$

In Γ, tuple coordinates retain occurrence order and the multiset retains multiplicities over all vertices, including the path vertices. This invariant reading of $P \times V$ has no additional vertex identifiers or access to pairs whose two endpoints lie outside the path.

The marked families $\mathcal{H}^{\mathrm{mark,ET}}_{N,K}$ and $\mathcal{H}^{\mathrm{mark,EVT}}_{N,K}$ follow the graph-stage conventions of Definition H.4. Both use global direction-mixed refinement and a tagged readout of selected diagonals. The ET endpoint lift includes both marks. EVT appends $m_P(i)$ to x_i; its pair lift still reads only y_{ij}, and its aligned pair MLP receives the marked endpoint states. Parameters are shared across bounded sizes, while graph states depend on P. We suppress N, K when unambiguous.

Lemma I.1 (Marked recovery and forgetting)**.** *A terminal marked color $\bar{c}^\star_{ij}$ determines $\left(\bar{c}^\star_{ip_t}, \bar{c}^\star_{p_t j}\right)_{t\in[k]}$. In particular, a selected diagonal determines its indexed row of selected pair colors and all selected diagonal colors, including repeated occurrences. At every depth, the unmarked color is a function of the marked color.*

Proof. By Lemma H.6, applied to marked inputs, the color $\bar{c}^\star_{ij}$ determines $\left\{\!\left\{(\bar{c}^\star_{iv}, \bar{c}^\star_{vj}) : v \in [n]\right\}\!\right\}$. Exactly one entry has a nonzero t-th coordinate in the intermediate vertex's mark: $v = p_t$. Its two components give the claimed colors. For $i = j = p_s$, recover each endpoint diagonal from $\bar{c}^\star_{p_s p_t}$ by Lemma H.7. Repeated occurrences select the same entry; the external occurrence tag still distinguishes their output positions. Finally, define f_0 by dropping the marks and f_{r+1} by applying f_r to the receiver and both colors of every multiset entry. Induction gives $c^r_{ij} = f_r(\bar{c}^r_{ij})$. The same construction permits projection onto any subset of mark coordinates. □

I.1 Proof of the expressivity hierarchy

Recall that $\mathcal{F}_A$ is the family of all length-preserving predictions that factor through the profile A. The readers in this definition are unrestricted; the proposition concerns the information supplied by an interface, rather than containment between prescribed finite neural architectures.

Proof of Proposition 4. **Reconstruction.** The unique tuple in Γ with a diagonal color in coordinate t is the one for $u = p_t$; it recovers the t-th column of Ψ. This also handles repeated occurrences. Conversely, a marked diagonal at p_1 determines the multiset of colors $b = \bar{c}^\star_{p_1 u}$. Applying Lemma I.1 to each b, then forgetting marks, recovers $(c^\star_{p_1 u}, \dots, c^\star_{p_k u})$, preserving the same u across coordinates and its multiplicity. Thus Ψ^{mark} determines Γ. For $k = 1$, the terminal multiset at $c^\star_{p_1 p_1}$ recovers Γ by its first projection. For $k = 2$, use the multiset at $c^\star_{p_1 p_2}$ and transpose its second component by Lemma H.7. Hence Ψ also determines Γ in these cases.

Function families. The reconstructions give maps r_1, r_2 such that $\Psi = r_1 \circ \Gamma$ and $\Gamma = r_2 \circ \Psi^{\mathrm{mark}}$. Composing an arbitrary reader with these maps proves $\mathcal{F}_\Psi \subseteq \mathcal{F}_\Gamma \subseteq \mathcal{F}_{\Psi^{\mathrm{mark}}}$. Equivalently, since

these profiles record the output length, $\mathrm{Ker}(\mathcal{F}_A) = \ker(A)$: unequal same-length profiles can be separated by assigning distinct constant output arrays, and unequal lengths already give distinct output components. The reverse reconstruction for $k \leq 2$ gives equality of the selected and joint families on that subdomain. The same kernel implications also nest their continuous compatible targets on every compact subset; no continuity of the reconstruction maps is needed for this formulation.

Shared witnesses. On $\mathbb{Z}_4^2$, the rook graph R joins distinct vertices sharing a coordinate; the Shrikhande graph S has adjacency differences $\mathcal{A} = \{\pm(1,0), \pm(0,1), \pm(1,1)\}$, modulo four. Use constant vertex features and adjacency pair features. Both graphs have strongly regular parameters $(16, 6, 2, 2)$, which determine every paired relation count. Inductively, their unmarked colors have the same three classes at every round: diagonal D, adjacent A, and nonadjacent N.

One marked position. Take $P = (r)$, $r = (0,0)$, so the unmarked profiles agree. Write $b_u = A_{ru}$ and $B_{uv} = b_u A_{uv} b_v$. The neighbors of r induce two disjoint triangles in R and a six-cycle in S, giving $\mathrm{tr}(B_R^3) = 12$ and $\mathrm{tr}(B_S^3) = 0$. The unique marked intermediate vertex makes B_{uv} a function of $\bar{c}^1_{uv}$. Then $\bar{c}^2_{uv}$ determines $q_{uv} = B_{uv} \sum_w B_{uw} B_{wv}$, $\bar{c}^3_{ru}$ determines $\sum_v q_{vu}$, and $\bar{c}^4_{rr}$ determines $\sum_{u,v} q_{vu} = \mathrm{tr}(B^3)$. Thus the marked diagonals differ. This single-mark gain on rook–Shrikhande is already reported by Zhou et al. (2023, Appendix G, Table 7).

Three-position paths. First choose the triangles

$$P_R = ((0,0),(0,1),(0,2)), \qquad P_S = ((0,0),(0,1),(1,1)). \tag{94}$$

Their Ψ profiles agree. Their common-neighbor intersections are $\{(0,3)\}$ and $\varnothing$, respectively, by the adjacency rules. Hence the multiplicity of AAA in Γ differs. Marked refinement also detects this difference in two rounds: each $\bar{c}^1_{p_s u}$ identifies the marked intermediates and recovers all three adjacency indicators to u; the next diagonal refinement counts those with all three indicators equal to one.

For the other strict inclusion, take the open paths

$$Q_R = ((0,1),(0,0),(1,0)), \qquad Q_S = ((0,1),(0,0),(0,3)). \tag{95}$$

Their endpoints are nonadjacent and neither triple has a common neighbor. The shared degrees, pairwise intersection counts and zero triple intersection give the same complete Γ histogram:

Pattern	DAN, ADA, NAD	AAN, NAA	ANA	ANN, NNA	NNN	AAA, NAN
Multiplicity of each	1	2	1	2	4	0

Projecting marks onto the central occurrence gives the one-position witness above, so the marked central diagonals differ by Lemma I.1. Both separating pairs use simple three-position paths. Each unequal-profile witness yields a lookup table available to the finer interface and not the coarser one, proving strictness of the function-family inclusions as well as of the kernel inclusions. On the finite witness sets these lookup tables are continuous, so strictness also holds for the corresponding continuous target classes on suitable compact subsets. □

Corollary I.2 (Strict path-marked comparison)**.** *For $N, K \geq 1$, under the stated arbitrary-capacity and tagged-readout conventions,*

$$\mathrm{Ker}(\mathcal{H}^{\mathrm{mark,ET}}_{N,K}) = \mathrm{Ker}(\mathcal{H}^{\mathrm{mark,EVT}}_{N,K}) = \ker(\Psi^{\mathrm{mark}}_{N,K}). \tag{96}$$

For $N \geq 16$, this kernel is strictly contained in $\ker \Psi_{N,K} = \mathrm{Ker}(\mathcal{H}_{N,K})$. On finite sets where the restricted kernels differ, the marked families realize strictly more compatible lookup tables.

Proof. Soundness follows from rooted-color factorization. For separation on any finite set, the reverse-copy setup of Equations (58) and (59), augmented with endpoint marks, makes the bidirectional initial colors local. A biased MLP maps them to basis codes, and the finite-code ET construction realizes marked refinement. EVT realizes it after its aligned setup layer. By Lemma I.1, each selected diagonal recovers the indexed marked profile; its occurrence tag identifies the output position. The terminal multiset cardinality records n, and the output component records k. A shared tagged MLP therefore separates unequal profiles and interpolates compatible finite lookup tables across bounded sizes. Strictness follows from Proposition 4, including its one-position witness. □

Aggregation control and scope. For any $k \geq 2$, the indexed two-row histograms $\left(\{\{(c^\star_{p_s u}, c^\star_{p_t u}) : u \in [n]\}\}\right)_{s<t}$ have the same kernel as Ψ: each follows from $c^\star_{p_s p_t}$ by stable reconstruction and transposition, while its diagonal entries recover both endpoint diagonals and both pair orientations. The experiment in Section 3.3 and Appendix C tests the gain of joint three-row reading over these separate summaries. The comparison with marked refinement is established by the proof above.

The profiles are compared after stabilization. A binary membership mark cannot replace occurrence marks. Joint reading accesses kn pair states instead of k^2, while preserving the unmarked cache. At fixed depth, the dense marked implementation uses $O(n^3 d)$ work per query; the reusable cascade encodes the graph once and uses $O(k^3 d)$ path-stage work per query. These are implementation costs, not computational lower bounds.

J Directional Attention: Strictness, Position, and Memory

We first give the contraction identities and strict comparison behind the directional operator used in the main architecture. The remaining optional extensions address positional bias and activation memory: EVT–RoPE changes the path-stage positional parameterization while preserving its information boundary, and streaming over the intermediate index evaluates every oriented triangular head with less activation memory.

J.1 Directional triangular attention and strictness

The aligned pair map remains exactly Equation (14): its receiver (i, j) reads y_{ij}, not y_{ji}. Direction is introduced only through the two indexed views in Equation (1). These views remain jointly equivariant because transposition commutes with graph relabeling:

$$(\sigma \odot Y)^{\leftarrow}_{ij} = y_{\sigma^{-1}(j),\sigma^{-1}(i)} = (\sigma \odot Y^{\leftarrow})_{ij}. \tag{97}$$

Thus head-wise direction choices preserve relabeling equivariance while keeping the two ordered pair states distinct.

Chain and common-endpoint contractions. The benefit can be seen without attention scores. Under Definition H.4, the following identities apply to raw channels; a practical pre-normalized block instead contracts its normalized, projected pair channels. Let $A \in \mathbb{R}^{m\times m}$ be one scalar pair channel, set $A^{\rightarrow} = A$ and $A^{\leftarrow} = A^\top$, and set the triangular logits to zero. With identity scalar value projections and all m intermediate indices valid, a head returns $m^{-1}\mathcal{T}_{\alpha\beta}(A)$, where

$$\mathcal{T}_{\alpha\beta}(A)_{ij} := \sum_{r=1}^{m} A^{\alpha}_{ir} A^{\beta}_{rj}, \qquad \alpha, \beta \in \{\rightarrow, \leftarrow\}. \tag{98}$$

The four choices are

views (α, β)	indexed contraction	matrix form and motif
$\rightarrow\rightarrow$	$\sum_r A_{ir}A_{rj}$	A^2: directed chain $i \rightarrow r \rightarrow j$
$\rightarrow\leftarrow$	$\sum_r A_{ir}A_{jr}$	$AA^\top$: common target $i \rightarrow r \leftarrow j$
$\leftarrow\rightarrow$	$\sum_r A_{ri}A_{rj}$	$A^\top A$: common source $i \leftarrow r \rightarrow j$
$\leftarrow\leftarrow$	$\sum_r A_{ri}A_{jr}$	$(A^\top)^2$: reverse directed chain

The code names the right-arrow view `stored` and the left-arrow view `transpose`. With four heads, `balanced` assigns one head to each row of this table, while `chain_only` assigns all four heads to its first row. With padding, uniform attention divides by the number of valid intermediate indices, rather than the padded size. The middle two are the bilinear cores of AlphaFold's outgoing and incoming triangle multiplication, up to its projections, gates, and post-sum processing (Jumper et al., 2021; Abramson et al., 2024). AlphaFold's triangle-attention blocks and the triplet architectures reviewed in Appendix K share some of these index geometries but use different scores or values. EVT assigns the four choices head-wise inside the same attention layer, whose value depends jointly on both oriented pair states. With nonzero logits it produces content-dependent, normalized analogues rather than fixed matrix products.

Softmax produces the average in Equation (98), whereas the displayed matrix products are sums. For fixed m, the factor m is absorbed exactly by the output projection. Uniformly over $m \le N$, a companion diagonal head supplies the anchor $s = 1/m$ from Equation (62); on every compact input set, the following pointwise MLP uniformly approximates $(u, s) \mapsto u/s$, since $s \ge 1/N$. With masks, this requires either a valid diagonal anchor or an explicit count of valid intermediate indices exposed to the post-map.

Strictness relative to the forward-tokenized compositional restriction. Let $\mathcal{E}_N^{Y,\mathrm{pure}}$ and $\mathcal{V}_N^{Y,\mathrm{pure}}$ be the rooted ET/EVT families obtained by restricting every triangular head to $(\to, \to)$ and using the pointwise initialization Equation (43). Write $\gamma_N^{Y,\mathrm{pure}}(X, Y; i, j) = c_{ij}^{R_N,\mathrm{pure}}$ for the corresponding stable rooted color.

Proposition J.1 (Strict directional gain)**.** *For unrestricted directed pair inputs and $N \ge 4$,*

$$\mathrm{Ker}(\mathcal{E}_N^Y) = \mathrm{Ker}(\mathcal{V}_N^Y) = \ker(\gamma_N^Y) \subsetneq \ker(\gamma_N^{Y,\mathrm{pure}}) = \mathrm{Ker}(\mathcal{E}_N^{Y,\mathrm{pure}}) = \mathrm{Ker}(\mathcal{V}_N^{Y,\mathrm{pure}}). \tag{99}$$

Moreover, the pure families cannot uniformly approximate the rooted coordinate map $(A; i, j) \mapsto (AA^\top)_{ij}$ on the finite witness below with error smaller than $1/2$, whereas one $(\to, \leftarrow)$ head realizes it up to the recoverable factor $1/m$.

Proof. The mixed families contain their pure restrictions, giving the non-strict kernel inclusion. For the pure kernel equality, omit the transpose part of Lemma H.7: the terminal canonical multiset over intermediate vertices still recovers both endpoint colors and the required global multisets. Consequently, the factorization argument of Equation (57) shows that every pure ET or EVT state is a function of its stable pure color. Conversely, the forward setup and zero-logit $(\to, \to)$ basis-histogram construction Equations (60) and (61) realize every pure refinement. Thus $\mathrm{Ker}(\mathcal{E}_N^{Y,\mathrm{pure}}) = \mathrm{Ker}(\mathcal{V}_N^{Y,\mathrm{pure}}) = \ker(\gamma_N^{Y,\mathrm{pure}})$.

For strictness, let vertex features be constant and consider

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{100}$$

Starting from $(A_{ij}, \mathbf{1}_{\{i=j\}})$, compositional 2-FWL reaches the stable class matrix

$$\begin{pmatrix} 0 & 1 & 2 & 3 \\ 1 & 0 & 3 & 2 \\ 4 & 4 & 5 & 6 \\ 4 & 4 & 6 & 5 \end{pmatrix}. \tag{101}$$

Indeed, direct enumeration of the ordered pairs of initial colors over all intermediate vertices gives these seven classes after one refinement. If H_a denotes the multiset over intermediate vertices of any entry labeled a, then

$$\begin{aligned} H_0 &= \{\{(0,0),(1,1),(2,4),(3,4)\}\}, \\ H_1 &= \{\{(0,1),(1,0),(2,4),(3,4)\}\}, \\ H_2 &= \{\{(0,2),(1,3),(2,5),(3,6)\}\}, \\ H_3 &= \{\{(0,3),(1,2),(2,6),(3,5)\}\}, \\ H_4 &= \{\{(4,0),(4,1),(5,4),(6,4)\}\}, \\ H_5 &= \{\{(4,2),(4,3),(5,5),(6,6)\}\}, \\ H_6 &= \{\{(4,2),(4,3),(5,6),(6,5)\}\}. \end{aligned} \tag{102}$$

These multisets are constant within each displayed class and pairwise distinct, so one further refinement preserves exactly the same partition; it is stable. Hence rooted pairs $(3, 1)$ and $(3, 2)$ are indistinguishable to every pure graph-stage ET or EVT, while

$$(AA^\top)_{31} = 0, \qquad (AA^\top)_{32} = 1. \tag{103}$$

Every such pure pair readout therefore has entrywise uniform error at least $1/2$ on the rooted two-point set $\{(A; 3, 1), (A; 3, 2)\}$ for $(A; i, j) \mapsto (AA^\top)_{ij}$. The numerical class labels in Equation (101)

are arbitrary; only their equality pattern matters. By contrast, the $(\rightarrow, \leftarrow)$ head computes this target in one uniform triangular step: on this four-vertex witness it returns $\frac{1}{4}AA^\top$, and a fixed output projection multiplies by four. The mixed family contains the pure family by assigning every head $(\rightarrow, \rightarrow)$, so this witness makes the inclusion strict. The approximation lower bound and mixed realization follow from Equations (62), (98) and (103). □

The obstruction disappears on symmetric inputs, where $A^\top = A$, and more generally whenever the reverse atomic pair type is determined by the forward one. This rooted witness does not by itself prove strictness after a restricted $P \times P$ selection; that additionally depends on which ordered occurrences the path selects.

Implementation check on the directed witness. Assigning one head to each orientation gives the implemented EVT access to all four directed pair contractions. On the witness of Proposition J.1, chain-only EVT leaves the two rooted-pair states equal up to numerical precision (maximum gap 2.2×10^{-16} over four random initializations), whereas balanced EVT separates them in every run (minimum gap 5.2×10^{-2}). This one-layer probe checks the predicted separation on the directed witness in the implemented model.

J.2 Endpoint-aware EVT–RoPE

Only the path EVT has semantic coordinates: vertex tokens are occurrences and pair tokens have two ordered occurrence endpoints. We therefore apply RoPE to the second EVT, never to arbitrary graph labels in the cached stage.

For even rotary width d_r, let

$$R_a = \bigoplus_{h=1}^{d_r/2} \begin{bmatrix} \cos(\omega_h a) & -\sin(\omega_h a) \\ \sin(\omega_h a) & \cos(\omega_h a) \end{bmatrix}, \qquad R_a^\top R_b = R_{b-a}. \tag{104}$$

Give x_s one port V at s, and y_{st} a source port S at s and target port T at t. For an oriented view, these are the endpoints of the pair state actually read:

$$(\mathrm{pos}_{\mathsf{S}}(y_{st}^{\alpha}), \mathrm{pos}_{\mathsf{T}}(y_{st}^{\alpha})) = \begin{cases} (s,t), & \alpha = \rightarrow, \\ (t,s), & \alpha = \leftarrow. \end{cases} \tag{105}$$

For a set $\mathcal{A}$ of query–key port pairings, add

$$\ell_{\mathcal{A}}^{\mathrm{rot}}(z,z') = \frac{1}{\sqrt{|\mathcal{A}| d_r}} \sum_{(a,b)\in\mathcal{A}} \langle R_{\mathrm{pos}_a(z)} q_z^{ab}, R_{\mathrm{pos}_b(z')} k_{z'}^{ab} \rangle \tag{106}$$

to the plain content score, with a trainable gate. The attention types use:

attention	port pairings $\mathcal{A}$	offsets
$X_s \leftarrow X_u$	(V,V)	$u-s$
$X_s \leftarrow Y_{uv}$	$(\mathsf{V},\mathsf{S}), (\mathsf{V},\mathsf{T})$	$u-s, v-s$
$Y_{st} \leftarrow X_u$	$(\mathsf{S},\mathsf{V}), (\mathsf{T},\mathsf{V})$	$u-s, u-t$
$Y_{sr}^{\alpha}, Y_{rt}^{\beta}$ in $\mathrm{TriAtt}^{\alpha,\beta}$	$(\mathsf{S},\mathsf{S}), (\mathsf{T},\mathsf{T}), (\mathsf{S},\mathsf{T}), (\mathsf{T},\mathsf{S})$	differences between the actual oriented endpoints in Equation (105)

For $(\alpha,\beta) = (\rightarrow,\rightarrow)$, the four triangular offsets reduce to $r-s, t-r, t-s, 0$; the other variants follow by swapping the corresponding source and target positions. Only queries and keys are rotated; values and residual states stay in a common frame. Source and target projections remain untied to preserve orientation. This is the endpoint analogue of ordinary RoPE (Su et al., 2024).

The exact theorem keeps the injective absolute codes ϕ, ψ. RoPE is an additional relative bias, not a replacement: constant values do not reveal absolute positions through rotated logits alone. Let $\mathcal{H}_{N,K}^{\mathrm{RoPE}}$ extend the family of Definition H.4 by adding Equation (106) to every attention in the second

EVT. We require that the absolute codes remain in the token states and that every rotary gate can be set exactly to zero. The latter choice gives $\mathcal{H}_{N,K} \subseteq \mathcal{H}^{\mathrm{RoPE}}_{N,K}$, hence

$$\mathrm{Ker}(\mathcal{H}^{\mathrm{RoPE}}_{N,K}) \subseteq \mathrm{Ker}(\mathcal{H}_{N,K}) = \ker(\Psi_{N,K}).$$

Conversely, the rotary matrices are deterministic functions of the retained occurrence positions. Induction through the second stage therefore shows that every RoPE state remains a function of the positioned selected states, so every member is profile compatible and

$$\ker(\Psi_{N,K}) \subseteq \mathrm{Ker}(\mathcal{H}^{\mathrm{RoPE}}_{N,K}).$$

Thus the kernels are equal. On every compact $S \subset \mathcal{D}_{\le N,\le K}$, family containment and Theorem 2 give

$$\begin{aligned}\mathcal{C}_\Psi(S) &= \overline{\{H|_S : H \in \mathcal{H}_{N,K}\}}^{\|\cdot\|_{\infty,S}} \\ &\subseteq \overline{\{H|_S : H \in \mathcal{H}^{\mathrm{RoPE}}_{N,K}\}}^{\|\cdot\|_{\infty,S}}.\end{aligned}$$

Every RoPE map is continuous and profile compatible, and these properties are preserved by uniform limits; hence the reverse inclusion holds as well. EVT–RoPE consequently has the same proved common kernel and relative uniform closure, but a different parameterization.

J.3 Flash-style triangular attention

Triangular attention is cubic in arithmetic but need not materialize cubic activations. Streaming over blocks of the intermediate index evaluates the exact same map in the spirit of FlashAttention (Dao et al., 2022).

Suppress batch indices, fix one head with direction pair (α, β), and let $m = n$ or $m = k$. Its projected oriented views are $Q^\alpha_{ir}, V^{\mathrm{L},\alpha}_{ir}$ and $K^\beta_{rj}, V^{\mathrm{R},\beta}_{rj}$, with query/key dimension d_a and value dimension d_v per head. Write

$$\begin{aligned} S_{irj} &= \langle Q^\alpha_{ir}, K^\beta_{rj}\rangle/\sqrt{d_a}, & P_{irj} &= \frac{e^{S_{irj}}}{\sum_u e^{S_{iuj}}}, \\ V^\triangle_{irj} &= V^{\mathrm{L},\alpha}_{ir} \odot V^{\mathrm{R},\beta}_{rj}, & O_{ij} &= \sum_r P_{irj} V^\triangle_{irj}. \end{aligned} \tag{107}$$

Proposition J.2 (Streaming triangular attention)**.** *Partition the reduction axis indexed by r into blocks of size at most b. In exact arithmetic, the forward map Equation (107) can be evaluated without a globally materialized tensor indexed by (i, r, j). For batch size B and H heads, its attention-core working activation memory, measured in stored scalar entries, is*

$$O\big(BHm^2(d_a + d_v) + BHbm^2(d_a + d_v)\big)\,, \tag{108}$$

where the first term is quadratic state resident in high-bandwidth memory (HBM) and the second is a conservative bound on temporary storage for the current block of intermediate indices. At fixed numerical precision, the same asymptotic bound holds in bytes. Thus hardware-bounded b gives $O(BHm^2(d_a + d_v))$ memory. The arithmetic remains $\Theta(BHm^3(d_a + d_v))$. Reverse-mode training with tile recomputation admits the same asymptotic attention-core working-memory bound.

Proof. For each output (i, j), retain a running maximum μ, normalizer z, and vector accumulator a, initialized as $\mu = -\infty$, $z = 0$, and $a = 0$. For a new block U, set

$$\begin{aligned} \mu^+ &= \max\{\mu, \max_{r\in U} S_{irj}\}, & \rho &= e^{\mu-\mu^+}, \\ z^+ &= \rho z + \sum_{r\in U} e^{S_{irj}-\mu^+}, & a^+ &= \rho a + \sum_{r\in U} e^{S_{irj}-\mu^+} V^\triangle_{irj}. \end{aligned} \tag{109}$$

Induction over blocks gives $O_{ij} = a/z$. Only the current block of scores and joint values, together with quadratic output accumulators, is live; every intermediate index is still visited for every output pair. Globally resident oriented projections $Q^\alpha, K^\beta, V^{\mathrm{L},\alpha}, V^{\mathrm{R},\beta}$, outputs, and softmax statistics give the first term in Equation (108). The temporary block, with at most b intermediate indices for all

output pairs, gives the second; a physical kernel further tiles the (i, j) axes, so only its output tile occupies SRAM or registers and no cubic tensor is written to HBM.

For training, save the four projections, O, and $\mathrm{LSE}_{ij} = \mu_{ij} + \log z_{ij}$, all quadratic in m, and recompute each score and probability tile during the backward pass. With upstream G_{ij}, set

$$D_{irj} = P_{irj}\big(\langle G_{ij}, V^{\triangle}_{irj}\rangle - \langle G_{ij}, O_{ij}\rangle\big). \tag{110}$$

Then

$$\partial Q^{\alpha}_{ir} = d_a^{-1/2} \sum_j D_{irj} K^{\beta}_{rj}, \qquad \partial K^{\beta}_{rj} = d_a^{-1/2} \sum_i D_{irj} Q^{\alpha}_{ir},$$
$$\partial V^{\mathrm{L},\alpha}_{ir} = \sum_j P_{irj} G_{ij} \odot V^{\mathrm{R},\beta}_{rj}, \qquad \partial V^{\mathrm{R},\beta}_{rj} = \sum_i P_{irj} G_{ij} \odot V^{\mathrm{L},\alpha}_{ir}. \tag{111}$$

Separate passes accumulate the left derivatives by (i, r) and the right derivatives by (r, j), assigning each derivative entry to one accumulation pass. For a transposed view, the final scatter simply swaps the two physical tensor indices; it does not alter the live-memory order. These gradient accumulators are quadratic; the recomputed P, D tiles are released before advancing, so no cubic tensor is saved and global atomic updates can be avoided by accumulating each entry in its assigned pass. The bounds in Proposition J.2 cover only the attention core. They exclude parameter and optimizer state, as well as residual and non-attention MLP activations; projection tensors, outputs, LSE values, and their attention-core gradients are included. □

This is not a direct call to standard scaled dot-product attention: the value depends jointly on both oriented pair states in Equation (107). Triton (Tillet et al., 2019) is a natural production target; Helion, a higher-level tiled DSL that lowers to Triton, is a convenient prototyping tool (PyTorch Team at Meta, 2025). These are software directions, not reported kernels or speedups. The proved claim is the exact quadratic attention-core working-memory schedule under hardware-bounded tiling; cubic arithmetic remains.

K Extended Related Work

The central distinction in this paper is between the information available to a model and the computation used to expose it. We position both aspects relative to graph Transformers, path models, and conditional architectures.

Graph Transformers and higher-order expressivity. Graphormer, GraphGPS, TokenGT, and edge-augmented Transformers inject graph structure through attention biases, hybrid message passing, or vertex/edge tokens (Ying et al., 2021; Rampášek et al., 2022; Kim et al., 2022; Hussain et al., 2022). Higher-order variants instead move part or all of the computation from vertices to ordered pairs or triplets.

Triangular and triplet-based pair interactions. Non-softmax pair layers already aggregate two pair states through a third index. PPGN interleaves pointwise MLPs with featurewise matrix multiplication, while PPGN++ also supplies the transposed pair tensor; coordinate transposition is part of the general basis of second-order equivariant linear maps (Maron et al., 2019a; Puny et al., 2023; Maron et al., 2019b). AlphaFold's outgoing and incoming triangle-multiplication blocks use the complementary common-target and common-source contractions, and Uni-Mol+ combines both in one pair update (Jumper et al., 2021; Abramson et al., 2024; Lu et al., 2024).

Softmax-normalized variants make different choices within the same family of three-index geometries. Edge Transformers (ETs) use the compositional pair states (i, r) and (r, j) (Bergen et al., 2021); their graph-distinguishing boundary is analyzed through 2-FWL (Müller et al., 2024). Separately, Kozachinskiy et al. (2025) prove lower bounds for one-layer standard-softmax Transformers on several compositional tasks and, for quotient binary-relation composition, for one-layer mixed standard/triangular-attention Transformers with subpolynomial head counts, embedding dimension, and output-MLP size; these task-specific bounds are not ET-specific. AlphaFold's starting- and ending-vertex triangle-attention blocks use complementary common-endpoint incidence patterns. In

each block, the receiver supplies the query, one of the other two pair states supplies key and value, and the remaining pair state supplies a scalar bias (Jumper et al., 2021; Abramson et al., 2024). Triplet Edge Attention scores the receiver (i, j) jointly with (i, r), (j, r), and vertex/global features, then aggregates a value from (i, r), using the common-target geometry (Jung & Ahn, 2023). The Triplet Graph Transformer couples complementary common-target and common-source patterns, using $(i, r), (j, r)$ and $(r, i), (r, j)$, respectively. In its attention form, one of these pair states supplies the value while the other modulates its score and gate; its cheaper aggregation uses softmax-normalized, sigmoid-gated analogues of the corresponding bilinear contractions (Hussain et al., 2024). In these attention variants, each value comes from one pair state; EVT computes values jointly from both.

Our four oriented variants provide common notation for the chain, common-target, common-source, and reverse-chain index patterns, without identifying the architectures' complete score and value maps. EVT distributes these variants across heads and couples the resulting pair states to vertices. Relative to the forward-tokenized compositional ET family formalized in this paper, access to reversed pair states yields a strictly finer rooted-pair common kernel on directed inputs. The path theorem then characterizes the profile obtained by selecting the resulting stable colors at path occurrences (Sections 2 and 3 and Appendices H.2 and J.1).

Coupled vertex–relation representations. Architectures with vertex and edge states are common in relational reasoning, molecular modeling, and routing. RouteNet, for example, couples paths and links, while edge-aware graph Transformers update several entity types (Rusek et al., 2020; Hussain et al., 2022). EVT retains all ordered pairs and supports native vertex–pair communication. TT applies this typed block on both the ambient graph and the selected occurrence graph; other instances vary the path processor while retaining the same reusable interface.

Graph-conditioned sequences and paths. Cai and Lam's graph-to-sequence Transformer, WikiGraphs, and CERES condition sequence generation or language modeling on graph representations (Cai & Lam, 2020; Wang et al., 2021; Feng et al., 2022). PathNN, pathGCN, and Gretel use paths as aggregation objects or prediction targets (Michel et al., 2023; Eliasof et al., 2022; Cordonnier & Loukas, 2019). Our path is an ordered index sequence that may repeat vertices and indexes the output itself; in graph applications it is typically adjacency compliant. The complete pair selection preserves occurrence-level relations while keeping the upstream graph memory independent of the path.

Target labeling and subgraph expressivity. The labeling trick conditions graph representations on target sets and ordered structures (Wang et al., 2025). The tuple labels of Zhou et al. (2023) record every occurrence position, including repeated vertices; their framework also aggregates over tuples to obtain graph representations. Xu & Zou (2024) compare tuple representations obtained by varying the number of marks and the order of the network. Subgraph WL analyses characterize the effects of marking and pooling order (Zhang et al., 2023), while Feng et al. (2023) extend FWL through joint tuple aggregation during refinement. The gain of marking and its per-query recomputation cost are established (Wang et al., 2025); the single-mark rook–Shrikhande separation is also known (Zhou et al., 2023, Appendix G).

Building on these constructions, we characterize an ordered graph-to-path interface and compare three precise information profiles: complete $P \times P$ selection, invariant joint $P \times V$ reading of the same stable unmarked pair cache, and occurrence-marked refinement. To our knowledge, this exact comparison has not previously been established for the ordered pair-cache interface considered here. It identifies which distinctions can be recovered by changing the readout while preserving graph reuse, and which require richer information. The assumption of maximally expressive vertex representations in Wang et al. (2025, Theorem 21) is stronger than a 2-FWL bound; that completeness result alone does not give our ET/EVT characterization.

Genomic context and cache-then-query models. DNA language models learn from linear sequences (Nguyen et al., 2023; Benegas et al., 2023; Dalla-Torre et al., 2025), whereas pangenomes and pantranscriptomes represent multiple biological sequences in one graph (Garrison et al., 2018; Liao et al., 2023; Sibbesen et al., 2023). The many-paths-per-graph regime makes a cache-then-query decomposition natural. It is analogous to an encoder–decoder system in which the source memory is reusable, but our decoder is relational and aligned with both selected vertices and pairs.

Symmetry and relative universality. Graph-network universality and relative approximation are established (Keriven & Peyré, 2019; Maron et al., 2019c; Azizian & Lelarge, 2021), including shared parameters across bounded graph sizes. Transformer approximation with positional information is also studied by Yun et al. (2020); its L^p guarantees do not supply our uniform-norm conclusion. We identify the relevant information explicitly as a selected 2-FWL profile. The architecture-specific work establishes exact internal feature-bank compilation and a shared readout over bounded path lengths, allowing the relative approximation argument to apply.

L Standalone Software and Reproduction Guide

The supplement separates reusable model implementations, dataset management, and experiment reproduction. It can be extracted and used independently of the manuscript and development repository; no private account or cloud deployment is needed for the checks and numerical reconstruction below. All software paths and commands in these appendices are relative to its extracted root. Its `README.md` is the entry point, and `requirements-tested.txt` records tested dependency versions.

L.1 Scope of reproduction

Reconstructing a summary from saved measurements, evaluating a saved model, and training a new model are different operations. The release makes these distinctions explicit in its commands and output receipts.

Component	Supplied capability and scope
Numerical evidence	Compact measurements, grouped scores and selected raw logits reconstruct 27 tables and eight plots. Original conditional bootstrap intervals are not recomputed.
Dataset toolbox	Fixed synthetic generation recipes, public Atlas retrieval/reconstruction, and Figshare retrieval with ThorAxe cohort/split checks. Full datasets are not packaged in the software archive.
Model toolbox	Graph encoders, ordered selection, path processors and task heads; preserved task-specific implementations are identified separately.
Experiment adapters	Fresh main synthetic fits, Atlas training/evaluation, and nested ThorAxe graph/sequence campaigns. Auxiliary studies have varying coverage, specified in their experimental appendices.
Historical checkpoints	Run identities and availability inventories, not trained weights. Historical replay requires separately supplied, verified model states.
Optimization histories	Original Task 1 histories are included; the original Task 2 per-update histories are unavailable. New histories are recorded and labelled as new.

Table L.1: Reproduction coverage of the standalone supplement. A successful software check or numerical reconstruction is not a new training campaign or a replay of historical checkpoints.

Checkpoint and figure provenance. No trained predictor weights are distributed in this software archive. The inventories in `experiments/resources/` and `experiments/atlas/resources/` document historical run identities and recovery requirements. Prediction or metadata hashes are not substitutes for missing weight hashes. Fresh fitting produces new checkpoints; it does not recover an unavailable historical state. In particular, ThorAxe `node_esmc_embeddings.pt` files contain model inputs, not predictor checkpoints.

The figure command recomputes numerical summaries and produces independent layouts from retained measurements. It does not rerun inference, create new timing measurements, or recreate missing trajectories from final scores. The historical Task 2 optimization plot remains a rendered archival figure in the manuscript; its missing numerical histories are explicitly reported by the software (Appendix D.3.2).

L.2 Installation and numerical reconstruction

Python 3.12 is the tested environment. From the extracted supplement:

```
python -m venv .venv
source .venv/bin/activate
python -m pip install -e '.[test]'
python reproduce.py verify
python reproduce.py figures
python -m pytest
```

After dependency installation, these commands need no dataset downloads or full benchmark training; tests may perform small fixture updates. The verifier and figure command do not fit models. The file verifier checks the distribution manifest; the figure command uses compact records in `figures/data/` and writes `outputs/figures/`. It saves CSV tables, PDF/PNG plots, numerical summaries, and a receipt with input/code/output digests. Add `--tables-only` to omit rendering, or `--output PATH` to choose a destination. Matplotlib renders the numerical figures without TeX. The test suite checks software behavior on fixtures and small examples, not the accuracy or convergence of all trained models.

L.3 Dataset management and information boundaries

The `datasets/` toolbox keeps graph observations, ordered queries, targets, masks, and split identities explicit. It provides generation and retrieval recipes rather than silently downloading data during import. `GRAPHPATH_DATA_ROOT` can relocate the data cache.

For the main synthetic comparisons, the following command generates the fixed inputs and checks their content fingerprints:

```
python reproduce.py data generate synthetic \
  --name d2-path-product-3 t1-replica1 t1-replica2 s4-budget
```

Task 2 retains normalization fitted on the full 2,048-graph training pool and a fixed 512-graph optimization subset; a new subset must not silently change that normalization. Queries sharing a graph remain in the same partition.

Atlas preparation retrieves checksum-pinned public archives and rebuilds the temporal graph/path benchmark. ThorAxe preparation retrieves eight Figshare archives and verifies the fixed candidate cohort and gene splits; it does not rerun upstream sequence-embedding or ThorAxe construction. Their commands and provenance qualifications are in Appendices E.3 and F.7. The loaders require explicit opt-in to load their test partitions; hash verification of those files is not test evaluation. Neither data preparation nor numerical reconstruction authorizes a new selection decision using held-out labels. Source citations and third-party data terms are retained in `datasets/README.md` and the dataset resource files.

L.4 Model interfaces and cache validity

The `models/` toolbox exposes `build_synthetic_model`, `build_atlas_model`, and `build_thorax_model`. The general encoder library is under `src/path_augmented_transformer/`; its practical selection, normalization and prediction-head conventions are described in Appendices G.2 and G.3.

For a padded batch of size B, the general `GraphPathBatch` uses vertex tensors of shape (B, n, d_X), ordered-pair tensors of shape (B, n, n, d_Y), and path-index tensors of shape (B, k), together with Boolean validity masks. Valid path positions form a nonempty prefix; padding is excluded from attention and aggregation. Its optional `graph_index` associates multiple query paths with shared graph states without breaking differentiation. Atlas additionally has a one-graph/many-query interface with zero-weight padded queries.

Reuse is valid for fixed graph inputs and fixed encoder weights. At inference, encode once and evaluate many paths; during learning, shared graph computation may be reused within an update, but caches must be recomputed after encoder parameters change. Freezing an encoder for a reader experiment is a different training protocol, not an implicit cache optimization.

Model names do not make task adapters interchangeable. Atlas uses its preserved original implementation under a private namespace; the canonical ThorAxe graph and sequence campaigns use isolated source snapshots. The public ThorAxe factory is a convenient interface for new experiments,

not a replacement for those historical trainers. Use recorded experiment recipes, rather than factory defaults, to recover the stated architectures, readouts and optimization settings. Source verification establishes code identity, not identity of unavailable historical weights.

L.5 Training, evaluation, and validation scope

The experiment guide `experiments/README.md` gives complete invocations. Task-specific commands appear in Appendices D.2, D.3, E.3 and F.7. Synthetic `--inspect` and Atlas recipe inspection do not fit a model; synthetic `--smoke` performs a one-update integration check. A full experiment requires the declared seeds, data exposure, update budgets and validation-selection rule. A shorter diagnostic is not a replacement for a completed fit. New runs must use separate output directories and retain their source, input and selection identities.

Atlas historical replay requires external weights and archived validation predictions with matching hashes. ThorAxe campaign generation separates inner fitting, inner-only selection and final fitting; its explicit `final-archived` mode reuses historical choices rather than conducting a new nested search. These choices must remain visible in any comparison. Timing figures describe the original measurement hardware, not the computer used to regenerate the figure.

`VALIDATION.md` records the standalone release checks and their exact scope: isolated installation, model forward/backward and fresh-state round trips, numerical reconstruction, synthetic input regeneration, Atlas reconstruction, and ThorAxe example/cohort/split checks. These do not establish a new full training campaign, historical checkpoint replay, or new TPU measurements. In particular, the recovered ThorAxe input hashes do not establish identity with the unretained historical embedding snapshot. All reported experimental values in this paper remain the archived measurements, with the limitations stated in their task sections.