MERIT: SCALING KG FOUNDATION MODELS WITH RELATION TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge graph (KG) foundation models (KGFMs) predict links on unseen graphs zero-shot. KGFMs have been pioneered by ULTRA which builds a relation graph from co-occurrences that match motif templates, and passes messages over it to obtain relation representations. More recent KGFMs such as MOTIF and TRIX increase the size of this relation graph, and FLOCK replaces message-passing with probabilistic random-walk ensembles. However, larger relation graphs mean more messages, and more walks mean more sequences to encode, so they improve performance at higher training and inference cost. This cost hinders their use in more complex reasoning tasks, such as complex query answering. To reverse this trend, we design MERIT, the first KGFM with a transformer relation encoder: every relation name attends to every other, and the motifs enter as a soft prior rather than as edges. Over 56 benchmarks, MERIT matches the zero-shot MRR of SoTA KGFMs while being $6\times$ faster than TRIX and over $100\times$ faster than FLOCK at inference. Fine-tuned for complex query answering like UltraQuery, MERITQuery improves over it at matched cost.

1 INTRODUCTION

A knowledge graph (KG) stores facts as triples (h, r, t), each linking a head entity h to a tail entity t through a relation name r. For instance, the fact that the Colosseo is a landmark of Rome is stored as $(\text{Rome}, \textit{hasLandmark}, \text{Colosseo})$ (Figure 2). KGs support applications such as virtual assistants and recommendation (Ilyas et al., 2022; Dong, 2018), and are used to store incomplete data (Hogan et al., 2022). To reason over incomplete KGs, link prediction answers a query $(h, r, ?x)$ by ranking all candidate entities by the predicted plausibility that the resulting triple holds (Section 2). KG foundation models (KGFMs) make link prediction zero-shot: trained once on a collection of source graphs, they predict links on unseen graphs without retraining (Galkin et al., 2024b; Lee et al., 2023). In the *inductive transfer setting*, a KGFM is conditioned at test time on a new inference graph $\mathcal{G}_{inf}$, whose entities and relation names did not appear during pre-training, with edge set $\mathcal{E}_{inf}$. Then, it answers queries $(h, r, ?x)$ over a held-out edge set $\mathcal{E}_t$ that is disjoint from $\mathcal{E}_{inf}$, but involving entities and relation names in $\mathcal{G}_{inf}$, ranking every candidate entity t. To answer queries zero-shot, entity and relation representations have to be computed regardless of specific entities or relation names.

ULTRA (Galkin et al., 2024b) does so by building a relation graph $\mathcal{G}^r_{inf}$ whose nodes correspond to relation names and whose directed edges are co-occurrences that match four hand-crafted motif templates (Figure 2), and by running message-passing on $\mathcal{G}^r_{inf}$ with a relation encoder to obtain relation representations, on which it conditions a message-passing GNN entity encoder (Zhu et al., 2021) (Section 2). The relation encoder passes messages only along the edges of $\mathcal{G}^r_{inf}$, that is, only between relation names whose co-occurrence matches a hand-crafted motif template; we call this *motif-constrained message-passing*. MOTIF (Huang et al., 2025) and TRIX (Zhang et al., 2024) loosen it by increasing the size of the relation graph. FLOCK (Kim et al., 2026), instead, avoids motif-constrained message-passing entirely, replacing the two message-passing steps with probabilistic random-walk ensembles. A larger relation graph means more messages, and in FLOCK more walks mean more sequences to encode, so higher MRR than ULTRA comes at higher training and inference costs. Compared with ULTRA, **TRIX, MOTIF and FLOCK need $6\times$ to $17\times$ the pre-training time per batch and $6\times$ to over $100\times$ the inference time over the 56 benchmarks** (Figure 1). This cost hinders their use in more complex reasoning tasks, such as complex

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.

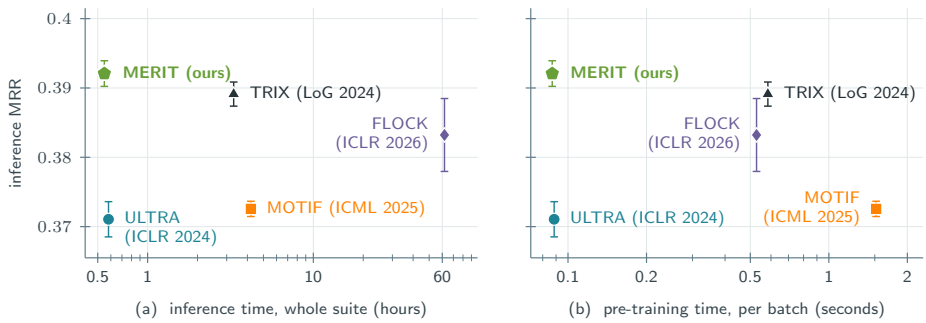

Figure 1: **MERIT matches SoTA KGFMs in MRR, with inference and pre-training times performance comparable to those of ULTRA**, while TRIX and FLOCK need $6\times$ and over $100\times$ more inference time and $7\times$ and $6\times$ more training time, respectively. Each point is the mean MRR (higher is better) over 56 benchmarks, averaged over pre-training runs, five for MERIT, ULTRA and TRIX and three for MOTIF and FLOCK; bars are the sample standard deviation over runs. Panel (a) is inference wall-clock over the whole suite and panel (b) pre-training time per batch, both measured on an H100 GPU and shown on a log scale.

query answering (CQA), where methods built on a neural link predictor run it at every hop of the query (Arakelyan et al., 2021; Bai et al., 2023; Yin et al., 2024) (Sections 2 and B.5). We argue that this cost is why UltraQuery (Galkin et al., 2024a), which uses ULTRA as its link predictor, is still the only KGFM for CQA, and there have not been equivalent versions for TRIX, MOTIF, or FLOCK. Behind these costs, the relation graphs of ULTRA, MOTIF and TRIX are designed on the assumption that a set of hand-crafted motif templates covers the co-occurrences between relation names that matter for link prediction. We argue that this assumption is too restrictive in two ways: two relation names that motif templates connect interact only through the templates that connect them, and two relation names that no template connects exchange no direct message.

Hence, we introduce MERIT (Section 3), which replaces message-passing on the relation graph with a transformer over relation names, so that every relation name attends to every other and the motifs enter as a soft prior rather than as a constraint (Figure 4). How strongly two relation names interact is then learned from their representations and shifted by how often each motif occurs between them, so MERIT keeps the information of the motifs that ULTRA uses without being limited to it (Section 5.1). Attention over all pairs grows quadratically with the number of relation names, but relation vocabularies are small, at most 3,210 tokens with their inverses, on AristoV4 (Table E.1); fused kernels scale exact attention to 64K tokens (Dao et al., 2022), also with an additive bias (Dong et al., 2024; Ying et al., 2021). Across 56 benchmarks, MERIT matches the zero-shot MRR of SoTA KGFMs with inference and pre-training times comparable to those of ULTRA, the fastest existing KGFM, and with a similar number of FLOPs and peak GPU and CPU memory (Figure 1 and Section B). Because MERIT is as efficient as ULTRA, it can be used for complex query answering. MERITQuery improves over UltraQuery at matched cost (Section 5.2).

Contributions. After reviewing how KGFMs reason over KGs (Section 2), (**C1**) we measure the training and inference cost of SoTA KGFMs, reporting FLOPs, peak GPU memory and inference time for the full suite of benchmarks, and pre-training time per batch (Figure 1 and Section B). (**C2**) We introduce MERIT, a framework that allows all relation names to exchange messages while using the motifs as a soft prior (Section 3). (**C3**) We demonstrate that the inference time and memory of MERIT stay comparable with those of ULTRA at every graph size (Section 4), and that it matches SoTA KGFMs in zero-shot MRR at a fraction of their inference time (Section 5.1). (**C4**) We introduce new hardness-balanced benchmarks for CQA following Gregucci et al. (2025) (Section D.3), and MERITQuery, which improves over UltraQuery at matched inference time (Section 5.2).

2 Reasoning over KGs with KGFMs

Knowledge graphs and reasoning over KGs. Formally, a KG is a tuple $\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{E})$, where $\mathcal{V}$ is a finite set of entities, $\mathcal{R}$ is a finite set of relation names, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{R} \times \mathcal{V}$ is a set of triples.

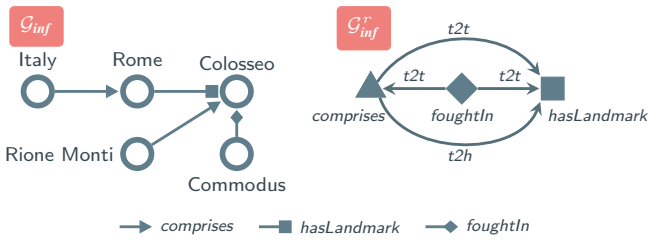

Figure 2: **Example inference graph $\mathcal{G}_{inf}$ (left) and its ULTRA-derived relation graph $\mathcal{G}^r_{inf}$ (right).** For visual simplification, we omit (i) *h2t* edges, since they are the reverse of *t2h*, and (ii) trivial *h2h* and *t2t* self-loops.

The main reasoning tasks over KGs studied in the literature are link prediction and complex query answering (CQA). Given a head entity h and a relation name r, **link prediction** ranks all candidate entities $e \in \mathcal{V}$ by the predicted plausibility that (h, r, e) holds, so that the correct but unobserved target t ranks as high as possible. **CQA** extends link prediction to more complex reasoning tasks where the objective is to retrieve a set of answers to a logical query that poses conditions over entities and relation names in a KG.

Following Arakelyan et al. (2021); Ren & Leskovec (2020), we consider the problem of answering logical queries with a single *target* variable, a set of constants including *anchor* entities, given relation names, and first-order logical operations that include conjunction, disjunction, negation and existential quantification. Complex queries are categorized into different types based on their structure; the types we consider are explained and drawn in Section D.1, and the simplest of them is link prediction, which in this task is referred to as a *1p* query. A more complex query, a *2i* query, involves an *intersection* made of two links. Considering the graph $\mathcal{G}_{inf}$ of Figure 2, and the query *"Where did Commodus fight, among the places that Rione Monti comprises?"*, a link predictor would have to answer the two queries (Commodus, *foughtIn*, $?x$), and (Rione Monti, *comprises*, $?x$), and then combine the two answers.

Reasoning on unseen graphs. Knowledge graph embeddings, such as TransE (Bordes et al., 2013), ComplEx (Trouillon et al., 2016) and RotatE (Sun et al., 2019), learn an embedding for every entity and relation name. GraIL (Teru et al., 2020) and NBFNet (Zhu et al., 2021) generalize to unseen entities but still learn embeddings for relation names. Hence, none of these methods can predict links in the *inductive transfer setting*. In this setting, a model is trained on multiple training graphs and, at test time, conditioned on a new inference graph $\mathcal{G}_{inf} = (\mathcal{V}_{inf}, \mathcal{R}_{inf}, \mathcal{E}_{inf})$ whose entities and relation names do not appear during training (Galkin et al., 2024b). The test graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{R}_t, \mathcal{E}_t)$ holds the triples to predict, which are held out from $\mathcal{G}_{inf}$ ($\mathcal{E}_{inf} \cap \mathcal{E}_t = \emptyset$) but involve only entities and relation names of $\mathcal{G}_{inf}$ ($\mathcal{V}_t \subseteq \mathcal{V}_{inf}, \mathcal{R}_t \subseteq \mathcal{R}_{inf}$).

ULTRA adds to $\mathcal{G}_{inf}$ the inverse (t, r^{-1}, h) of every triple (h, r, t), for instance $(\text{Colosseo}, \textit{hasLandmark}^{-1}, \text{Rome})$, and builds a relation graph $\mathcal{G}^r_{inf}$ in which every relation name in $\mathcal{R}_{inf}$ and its inverse correspond each to a node. The inverses are omitted in Figure 2 for visual simplification. The edges of $\mathcal{G}^r_{inf}$ are the co-occurrences that match the four motif templates h2h, t2t, h2t and t2h, which connect two relation names when they share an entity as heads, as tails, or as the head of one and the tail of the other, in either direction. For example, *foughtIn* and *comprises* share the tail Colosseo, so $\mathcal{G}^r_{inf}$ links them by a t2t edge (Figure 2). Given a query $(h, p, ?x)$ with query relation p, the relation encoder, a variant of NBFNet, runs motif-constrained message-passing on $\mathcal{G}^r_{inf}$ to compute a representation of every relation name in $\mathcal{R}_{inf}$, conditioned on p. The entity encoder, another instance of NBFNet, runs message-passing on $\mathcal{G}_{inf}$ with the relation representations as edge features; a final MLP then scores every candidate entity.

To loosen motif-constrained message-passing, MOTIF (Huang et al., 2025) and TRIX (Zhang et al., 2024) extend the relation graph of ULTRA along orthogonal axes, the former through motif length, by building a relational hypergraph, the latter through per-entity granularity. TRIX additionally couples the relation and entity encoders through iterative updates. FLOCK (Kim et al., 2026), instead, removes message-passing entirely, and with it the motif templates: it iteratively samples random walks, encodes them into sequences, and embeds them with a sequence model, at a cost that grows with the number and the length of the walks. However, these expressiveness gains come at a cost: all three require significantly more time for training and inference than ULTRA (Figure 1).

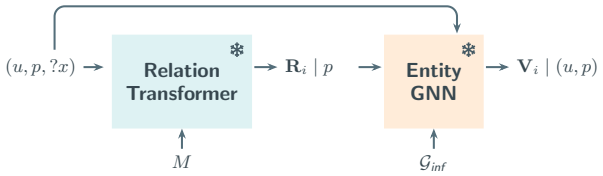

Figure 3: **In MERIT, the query-conditioned relation representations are obtained with the relation transformer.** The rest of the pipeline is as in ULTRA (Section 2).

UltraQuery (Galkin et al., 2024a) adapts ULTRA to CQA: it fine-tunes ULTRA on complex queries and runs it at every hop, mapping its scores to a fuzzy set of entities with values in $[0, 1]$ and combining these sets with fuzzy logic operators. See Section D.2 for more related work on CQA. In Section 3, we allow all relation names to exchange messages while using the motifs as a soft prior, not a constraint. We do so while preserving the computational cost of ULTRA, so that our MERIT can scale to CQA tasks in the same way.

3 MERIT: Motif-Enriched Relational Inductive Transformer

Under motif-constrained message-passing, two relation names exchange messages only if a motif template connects them (Section 2). The number of messages hence grows with the motifs, and with it the training and inference costs (Figure 1). In the following, we propose a framework where every relation name exchanges messages with every other, and where they can exchange messages that go beyond hand-crafted motifs.

MERIT. We introduce MERIT (**M**otif-**E**nriched **R**elational **I**nductive **T**ransformer): a transformer over relation names in which the motifs enter as a soft prior rather than as the edges of a relation graph. The transformer of MERIT reads $\mathcal{G}_{inf}$ only through a ***motif tensor*** $M \in \mathbb{R}^{|\mathcal{R}_{inf}| \times |\mathcal{R}_{inf}| \times T}$: the entry $M_{ij} \in \mathbb{R}^T$, the ***pair descriptor*** of (i, j), holds T numbers that describe how relation names i and j co-occur in $\mathcal{G}_{inf}$, and the row M_i collects the descriptors of i with the other relation names. The tensor enters at two points, the attention and the tokens,

$$M = \Psi(\mathcal{G}_{inf}), \qquad m_{ij}^k = f_k(M_{ij}), \qquad \mathrm{s}_r = g\big(\{\!\{\, M_{rj} : j \in \mathcal{R}_{inf} \setminus \{r\} \,\}\!\}\big) \in \mathbb{R}^d, \tag{1}$$

where Ψ computes M from $\mathcal{G}_{inf}$, $f = (f_1, \ldots, f_K) \colon \mathbb{R}^T \to \mathbb{R}^K$ maps a pair descriptor to one ***motif bias*** m_{ij}^k for each of the K heads, and g maps the multiset of the pair descriptors of a relation name to a vector of d entries, its ***relation signature*** s_r. To ensure zero-shot generalization on unseen graphs, none of Ψ, f and g ties a parameter to a relation name. For every relation name r, the transformer outputs a representation $\mathbf{h}_{r|p}$ conditioned on the query relation p; an entity GNN, here NBFNet (Zhu et al., 2021), then computes the query-conditioned entity representations and scores for all entities as in ULTRA (Galkin et al., 2024b) (Figure 3).

Motif bias. The transformer has L layers, each with multi-head self-attention over all $|\mathcal{R}_{inf}|$ tokens. The motifs enter here as additive bias (Ying et al., 2021) on the attention logits: in each head k, the motif bias m_{ij}^k of Equation (1) shifts the logit with which token i attends to token j. We take $f(\mathbf{0}) = \mathbf{0}$, so that a pair between which no motif occurs, $M_{ij} = \mathbf{0}$, is attended on its query-key logit alone, as any pair in a plain transformer (thin arrows in Figure 4c), while a pair between which motifs occur (thick arrows) is attended with a larger or a smaller weight. A further motif template adds a slice to M and one entry to every pair descriptor, and leaves the token dimension, the heads, the layers and the attended pairs unchanged, so no connection between relation names is added. With $a_{ij}^{(\ell,k)}$ the query-key logit of head k in layer ℓ, the transformer reads

$$\mathbf{x}_{r|p}^{(0)} = \big[\, \mathrm{s}_r \,\big\|\, \mathbf{z}_{r|p} \,\big], \qquad \mathbf{z}_{r|p} = \mathbb{1}_{r=p}\, \mathbf{1} + \mathbb{1}_{r \neq p}\, \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, I), \tag{2}$$

$$\mathbf{x}_{i|p}^{(\ell+1)} = \textsc{Update}\Big(\mathbf{x}_{i|p}^{(\ell)},\, \big\|_{k=1}^{K} \sum\nolimits_{j \in \mathcal{R}_{inf}} \mathrm{softmax}_j\big(a_{ij}^{(\ell,k)} + m_{ij}^k\big)\, \textsc{Value}_k\big(\mathbf{x}_{j|p}^{(\ell)}\big)\Big), \tag{3}$$

where $\|$ denotes concatenation, $\textsc{Value}_k$ is the value map of head k and $\textsc{Update}$ the output projection and residual feed-forward update of a pre-norm transformer encoder layer (Vaswani et al., 2017; Xiong et al., 2020), and $\mathbf{h}_{r|p} = \mathbf{x}_{r|p}^{(L)}$. The query flag $\mathbf{z}_{r|p}$ conditions the representations $\mathbf{h}_{r|p}$ on the query relation p: all ones on p following the labelling trick of ULTRA (Galkin et al., 2024b),

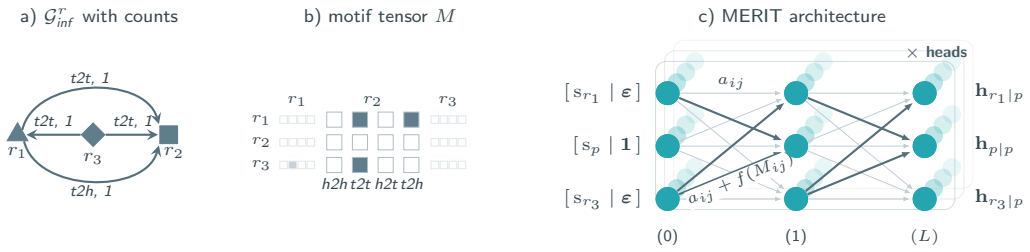

Figure 4: **MERIT runs a transformer over all relation tokens and injects the motifs in the token embeddings and as a bias on the attention.** The motif tensor M b) has the shape of the adjacency tensor of the relation graph $\mathcal{G}^r_{inf}$ a), with entity occurrence counts in place of edges. This tensor is then used as a prior for the MERIT architecture c), where every token contains the relation signature s_r of its relation, followed by random noise ϵ, or by ones for the query relation p (r_2 in the example); every arrow is an attention logit a_{ij}, and the arrows between co-occurring relations add the bias $f(M_{ij})$, over multiple heads.

and standard Gaussian noise on every other relation name, redrawn at every forward pass so that tokens with the same signature stay distinguishable (Abboud et al., 2021).

Relation signature. The relation signature s_r describes relation name r through its row of the motif tensor, independently of the query (Figure 4c). Since g reads the row as a multiset, s_r does not depend on the order of the other relation names and is defined for any number of them. In this way, two relation names that take part in the motifs in the same way receive similar tokens, and two that take part differently can receive different ones.

Instantiation. For Ψ, we use the four motif templates h2h, t2t, h2t and t2h of Section 2, so $T = 4$, and record in M_{ijt} the compressed count of the occurrences of motif t between i and j, one per distinct entity. Our f is linear, $f(M_{ij}) = B^\top M_{ij}$ with $B \in \mathbb{R}^{T \times K}$ shared by all layers, so motif t raises the attention of head k to a pair when $B_{tk} > 0$ and lowers it when $B_{tk} < 0$, by an amount that grows with the count; our g is linear on a fixed summary of the row, $s_r = W_g\, \phi(M_r)$, where $\phi(M_r) \in \mathbb{R}^{2T}$ holds, for each motif t, the largest entry $\max_{j \neq r} M_{rjt}$ of the row and the compressed count of the relation names j with $M_{rjt} > 0$: how strongly and how broadly r takes part in motif t. Both B and W_g start at zero, so at initialization MERIT is a plain transformer over noise and the query flag, and the strength of the prior is learned. See Section A for more details.

MERITQuery. Mirroring UltraQuery (Section 2), in MERITQuery we use MERIT as the link predictor over fuzzy sets of entities, with the logical connectives relaxed via fuzzy logic operators. MERITQuery starts from pre-trained MERIT and is fine-tuned on the complex queries of FB15k237, then answers queries on unseen KGs zero-shot (Section 5.2).

4 How does the cost of MERIT scale?

MERIT replaces message-passing on the relation graph with attention over all relation names, and further motifs add no connection between them (Section 3). We now measure the training and inference costs of MERIT against those of SoTA KGFMs, and answer the following research question: **(RQ1)** *How do the training and inference costs of MERIT scale, compared with those of SoTA KGFMs?* We study it along the size of the inference graph, the number of hops of a query and the number of motif templates.

Experimental setup. We measure training and inference cost on one NVIDIA H100 GPU with 94 GiB of memory, in a node with two AMD EPYC 9454 processors and 768 GiB of CPU memory. For the KGFMs with a relation encoder, all but FLOCK, we also measure the resources of each stage of inference (Section B.3): the precomputation of the relational structure, M for MERIT; the relation encoder, for MERIT the transformer; the entity encoder; and the score computation. Figures 1 and B.2 sum over the 56 benchmarks other than AristoV4, the one benchmark on which we could

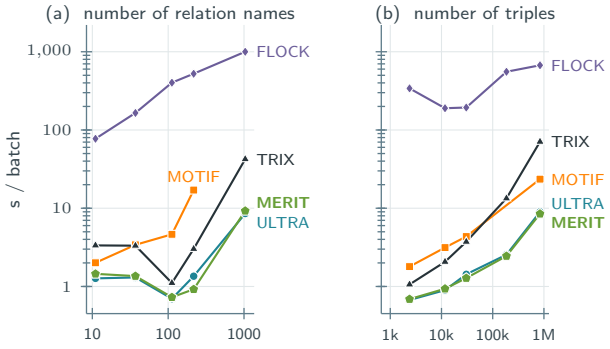

Figure 5: **MERIT keeps the inference cost of ULTRA at every graph size**, while the cost gap to TRIX widens with the number of triples and the gap to MOTIF with the number of relation names; FLOCK stays two to three orders of magnitude above. Markers are medians of the inference wall-clock within log-spaced bins of inference-graph size. Further analysis in Figure B.1, bin details in Table B.1. We omit the incomplete bins of MOTIF from the plot.

not profile MOTIF; Figure 5 includes AristoV4 for every other model and omits the bins of MOTIF that contain it. We measure pre-training time per batch to compare the speed of the models, not their time to convergence, which depends on the training recipe of each model. Section B.1 details the batch sizes and model configurations.

Costs of MERIT stay comparable with those of ULTRA at every graph size. Over the 56 benchmarks, MERIT needs 0.55 hours of inference against 0.58 for ULTRA, 3.31 for TRIX, 4.20 for MOTIF and 61.86 for FLOCK, and 0.087 seconds per pre-training batch against 0.089, 0.584, 1.518 and 0.528 (Figure 1). Across bins of relation names, entities and triples of the inference graph, the inference time of MERIT tracks that of ULTRA (Figure 5 and Section B.2). The largest gap in seconds to ULTRA is on AristoV4, 189 against 153 seconds, while TRIX needs 605 and FLOCK 28,128. FLOCK also needs 507 GB of CPU memory on DBpedia100k, against at most 1.5 GB for all others (Figures B.1 and 5). The relation stage takes 4.8% of the inference time of MERIT, against 84.5% for TRIX and 86.0% for MOTIF (Figure B.2).

The inference time of MERITQuery grows with the number of hops like that of UltraQuery. On the path queries of FB15k237, a *2p* query takes 1.96 times the time of a *1p* query for MERITQuery and 1.94 for UltraQuery, and a *3p* query 2.92 and 2.86 times, consistent with the link predictor running once per hop. If a k-hop query takes k times the time of a *1p* query, MERITQuery would answer more *4p* queries in one hour than TRIX, MOTIF or FLOCK answer *1p* queries (Figure B.4).

Beyond precomputation, more motif templates leave the time of MERIT unchanged. A further motif template adds to the precomputation of M, which takes 8.2% of the inference time of MERIT (Figure B.2). It adds little to the relation stage and leaves the entity encoder unchanged. To measure the relation stage at more templates, we duplicate the four slices of M to $T = 8$ and $T = 16$: the pre-training time per batch and the inference time over all 57 benchmarks, AristoV4 included, change by less than 1% (Section B.4). On AristoV4, where M is largest with $|\mathcal{R}_{inf}|^2 T$ entries, peak GPU memory on the relation stage grows by only 0.49 GB at $T = 8$ and 1.48 GB at $T = 16$, and stays below the largest peak over the suite, 10.73 GB, reached by the entity encoder.

> **Takeaway 1.** With a transformer over relation names, the cost of MERIT stays comparable with that of ULTRA at every graph size and number of hops of complex queries. Beyond precomputation, adding motif templates leaves the overall costs unchanged. We advise reporting training and inference cost alongside MRR when proposing new KGFMs.

5 How does MERIT fare in KG reasoning?

In this section, we evaluate MERIT on link prediction and CQA, the two reasoning tasks of Section 2. In link prediction (Section 5.1), we ask whether MERIT reaches the MRR of SoTA KGFMs and what the motifs, entering as a soft prior, contribute to its MRR. In CQA (Section 5.2), we ask how MERITQuery compares with UltraQuery, so far the only KGFM adapted to CQA.

Table 1: **Zero-shot MERIT matches SoTA KGFMs in every family of benchmarks**. Mean MRR and Hits@10 over the benchmarks of each family and over all 54, averaged over five pre-training runs per model, three for MOTIF and FLOCK, with a single run of MOTIF on AristoV4; ± is the sample standard deviation over runs. **Bold** marks the best mean of each column and every mean within one standard deviation of it, its own or that of the best.

Model	Family 1 (13 graphs)		Family 2 (18 graphs)		Family 3 (23 graphs)		Total (54 graphs)	
	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
ULTRA	0.310±.005	0.453±.007	0.434±.003	0.571±.003	0.342±.006	0.519±.004	0.365±.003	0.521±.003
MOTIF	0.306±.005	0.445±.007	0.431±.008	0.575±.008	0.346±.004	0.522±.006	0.365±.002	0.521±.001
TRIX	**0.329**±.004	**0.484**±.005	**0.451**±.003	0.589±.004	**0.362**±.004	0.536±.004	**0.384**±.002	0.541±.003
FLOCK	**0.331**±.009	**0.493**±.011	**0.448**±.011	**0.602**±.003	**0.364**±.007	**0.548**±.010	**0.384**±.006	**0.552**±.005
MERIT (ours)	**0.331**±.005	**0.489**±.007	**0.454**±.008	**0.600**±.005	**0.365**±.003	**0.544**±.009	**0.386**±.002	**0.549**±.001

5.1 Link prediction

In this subsection, we answer the following research questions: **(RQ2)** *How does the zero-shot and fine-tuning MRR of MERIT compare with that of SoTA KGFMs in the inductive transfer setting?*, and **(RQ3)** *How much of the MRR of MERIT comes from the attention over all relation names, and how much from the motifs?*

Experimental setup. We evaluate on the 54 benchmarks of Galkin et al. (2024b), which exclude the three pre-training graphs, FB15k237, WN18RR and CoDEx Medium. They are grouped into three families by how much of the entity and relation name vocabularies of $\mathcal{G}_{inf}$ already appears in the training graph $\mathcal{G}_{tr}$ of each benchmark:[1] Family 1 shares both entities and relation names, Family 2 shares relation names but introduces new entities, and Family 3 shares neither. For results that also include the pre-training graphs, see Section C.2; the conclusions of Figure 1, over 56 benchmarks, are unchanged on these 54 (Table C.2). We compare MERIT with ULTRA, MOTIF, TRIX and FLOCK (Section 2) and report the mean and the sample standard deviation over pre-training runs, five for MERIT, ULTRA and TRIX and three for MOTIF and FLOCK (Section C.1). Following Gregucci et al. (2026), we evaluate MOTIF on AristoV4 only once, from its released checkpoint, since its relational hypergraph does not fit on the GPU and must be streamed through a sharded precompute pipeline, and use that value for AristoV4 in all three runs of MOTIF. We report MRR and Hits@10 under the filtered protocol (Bordes et al., 2013), which removes the other known true candidates from the ranking. We mark in bold the best mean of each column and every mean within one standard deviation of it, its own or that of the best. We also stratify MRR on the 50 zero-shot benchmarks of Gregucci et al. (2026) by their four half-link scenarios, SQSA, SQUA, UQSA and UQUA, which classify each test triple by whether its query (Q) half $(h, r, ?)$ and its answer (A) half $(?, r, t)$ are seen (S) in $\mathcal{G}_{inf}$, that is, witnessed by a triple of $\mathcal{E}_{inf}$, or unseen (U) (Section C.3). To keep the inductive transfer setting, we fine-tune per benchmark on Family 3 only (Gregucci et al., 2026), and stratify the fine-tuned MRR by the same half-link scenarios. We fine-tune every pre-training run on the training graph $\mathcal{G}_{tr}$ of each Family 3 benchmark, with the fine-tuning protocol of ULTRA, except for FLOCK, which we fine-tune with its own (Kim et al., 2026).

MERIT matches the zero-shot MRR of SoTA KGFMs. Over the 54 benchmarks, MERIT scores 0.386 MRR, against 0.384 for TRIX and FLOCK and 0.365 for ULTRA and MOTIF; in Hits@10 it scores 0.549, within one standard deviation of FLOCK at 0.552 and ahead of TRIX at 0.541 (Table 1). Within each family, the mean MRR and Hits@10 of MERIT are the highest or within one standard deviation of the highest (Table 1; per benchmark in Section C.2). Per half-link scenario, over the 50 zero-shot benchmarks, MERIT has the highest mean on SQSA, 0.398 against 0.393 for ULTRA, and on UQSA, 0.568 against 0.565 for FLOCK, and scores lower than TRIX and FLOCK on SQUA, 0.175 against 0.183 and 0.181 (Table 2). On UQUA, which isolates generalization beyond $\mathcal{G}_{inf}$ (Gregucci et al., 2026), MERIT is competitive with FLOCK, 0.282 against 0.276, and ahead of ULTRA, MOTIF and TRIX at 0.255 to 0.260: MERIT generalizes beyond $\mathcal{G}_{inf}$ as well as FLOCK does.

[1] Family 1, Family 2 and Family 3 are also known as *transductive*, *inductive entity* and *inductive entity and relation* (Galkin et al., 2024b), after the original setting each benchmark was created for; Table E.1 lists the benchmarks in each, with the three pre-training graphs under Family 1.

Model	SQSA	UQSA	SQUA	UQUA
ULTRA	0.393±.004	0.546±.006	0.144±.006	0.258±.013
MOTIF	0.382±.003	0.545±.006	0.160±.006	0.255±.015
TRIX	0.392±.002	**0.564**±.004	**0.183**±.004	0.260±.014
FLOCK	**0.389**±.010	**0.565**±.008	**0.181**±.005	**0.276**±.016
MERIT (ours)	**0.398**±.004	**0.568**±.008	0.175±.005	**0.282**±.010

Table 2: **The gain of zero-shot MERIT reaches UQUA**, and is on par with FLOCK and ahead of the other KGFMs. Mean MRR per half-link scenario over the 50 zero-shot benchmarks; averages and bold as in Table 1.

Fine-tuning raises MERIT on every half-link scenario, while TRIX loses MRR on the two whose answer half is unseen. After per-benchmark fine-tuning, MERIT gains MRR on every scenario over the 22 Family 3 benchmarks, on SQUA from 0.144 to 0.167 and on UQUA from 0.233 to 0.255, and so do ULTRA, MOTIF and FLOCK (Figure 6 and Table C.5). TRIX alone loses MRR on these two scenarios, from 0.163 to 0.126 on SQUA and from 0.221 to 0.196 on UQUA. Over the 22 benchmarks, fine-tuned MERIT scores 0.384 MRR, up from 0.358 zero-shot, ahead of ULTRA at 0.375, MOTIF at 0.373 and TRIX at 0.368, and slightly below FLOCK at 0.391.

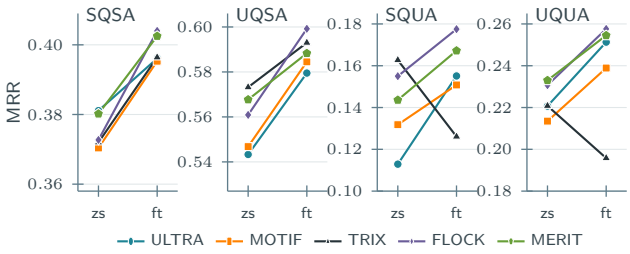

Figure 6: **Fine-tuning raises the MRR of MERIT on every scenario, UQUA included**. Zero-shot (zs) and fine-tuned (ft) mean MRR over the 22 Family 3 benchmarks, one panel per half-link scenario with its own scale.

Is attention all you need? To answer RQ3, we pre-train two variants of MERIT, one that keeps only the attention over all relation names and one that keeps only the motifs. The first is the *plain transformer*, with neither the motif bias nor the relation signature, over noise and the query flag (Section 3): it scores 0.312 MRR over one pre-training run, below ULTRA at 0.365. The second is *masked MERIT*: a relation name i attends only to itself and to the relation names j with $M_{ij} \neq \mathbf{0}$, the relation names with which i exchanges messages under motif-constrained message-passing (Section 2), and the query-key logit $a_{ij}^{(\ell,k)}$ is set to zero, so that its attention depends on the motif bias m_{ij}^{k} alone; it scores 0.368 over three runs, on par with ULTRA. With both, MERIT matches SoTA KGFMs at 0.386. The motifs help at both points where they enter, the attention and the tokens: removing either the motif bias or the relation signature lowers the MRR of MERIT (Section C.5).

5.2 Complex Query Answering

In this subsection, we answer the following research question: **(RQ4)** *How does the MRR of MERITQuery compare with that of UltraQuery, the only KGFM for CQA so far, on each family of benchmarks and on different query types?*

Experimental setup. The CQA benchmarks of UltraQuery cover all three families, and their queries come from the sampler of BetaE (Ren & Leskovec, 2020; Galkin et al., 2022; 2024a). Gregucci et al. (2025) show that this sampler skews the benchmarks towards queries simpler than their type suggests: a query of three hops (3p), for instance, can be answered by predicting a single link (1p) when its other two links are already in the graph given to the model, which for KGFMs is the inference graph $\mathcal{G}_{inf}$. They build the +H benchmarks of Family 1, in which the different levels of hardness of a query type are sampled with the same frequency: for example, for a 3p query, they hold the same number of query answer (QA) pairs that *reduce* to predicting a single link, two links, or all three. We build +H benchmarks for Family 2 and Family 3 following their construction, and evaluate on 18 +H benchmarks: NELL995+H and ICEWS18+H for Family 1, and our 9 Family 2 and 7 Family 3 benchmarks. We fine-tune UltraQuery and MERITQuery on FB15k237 with the same recipe, five times each, starting once from each of the five pre-training runs of ULTRA and of MERIT (Sections C.1 and D.4). To stay in the inductive transfer setting, we therefore leave

Table 3: **MERITQuery is ahead of UltraQuery over the 18 benchmarks and on Family 1 and Family 3, most on the union and negation types**. Mean MRR over the benchmarks of each family and over all 18, per group of query types, averaged over five fine-tuning runs per method; ± and bold, within each pair, as in Table 1.

	All		Conjunctive		Union		Negation	
Family	UltraQuery	MERITQuery	UltraQuery	MERITQuery	UltraQuery	MERITQuery	UltraQuery	MERITQuery
Family 1 (2)	0.062±.012	**0.077**±.005	0.080±.021	**0.103**±.006	0.090±.007	**0.108**±.011	**0.022**±.003	**0.023**±.003
Family 2 (9)	**0.274**±.003	**0.276**±.004	**0.231**±.003	0.227±.003	**0.341**±.012	**0.350**±.011	0.316±.003	**0.326**±.004
Family 3 (7)	0.346±.003	**0.368**±.013	0.357±.004	**0.367**±.009	0.324±.007	**0.367**±.019	0.337±.006	**0.370**±.018
All (18)	0.278±.003	**0.290**±.006	0.263±.004	**0.268**±.004	0.306±.008	**0.330**±.008	0.291±.003	**0.310**±.009

FB15k237+H, the third +H benchmark of Family 1, out of the 18 benchmarks of Table 3 and report it only in Section D.5. We exclude 1p, which is link prediction, and report the other 15 query types all together and in three groups: the conjunctive types, whose formula contains only conjunctions, the union types, with a disjunction and no negation, and the negation types, with a negation (Section D.1). We report the MRR of each query type under the filtered protocol of Ren et al. (2020), and average it over the query types of each group and over the benchmarks.

MERITQuery is ahead of UltraQuery over the 18 benchmarks and on Family 1 and Family 3, most on the union and negation types. Over the 18 benchmarks, MERITQuery scores 0.290 MRR, against 0.278 for UltraQuery, 0.330 against 0.306 on the union types, 0.310 against 0.291 on the negation types and 0.268 against 0.263 on the conjunctive ones (Table 3). Per family, the margin is largest on Family 3, 0.368 against 0.346, where MERITQuery is ahead on each of the 15 query types (Figure D.2). On Family 1, MERITQuery scores 0.077 against 0.062, and on Family 2 the two are within one standard deviation, 0.276 against 0.274, with UltraQuery marginally ahead on the conjunctive types, 0.231 against 0.227, and MERITQuery ahead on the negation types, 0.326 against 0.316. The inference time of MERITQuery matches that of UltraQuery, and is well below the time that TRIX, MOTIF and FLOCK would need if they were adapted to CQA (Section 4 and Figure B.4).

> **Takeaway 2.** With attention over all relation names and the motifs as a soft prior, MERIT matches the MRR of SoTA KGFMs in link prediction, and MERITQuery improves over UltraQuery in CQA. We therefore advise using the motifs as a prior, not as a constraint.

6 Conclusion

In this paper, we measure the training and inference costs of SoTA KGFMs and reveal that TRIX and FLOCK gain MRR over ULTRA at 6× and over 100× its inference time. To reverse this trend, we design MERIT, the first KGFM with a transformer relation encoder, in which every relation name attends to every other and where the motifs enter as a soft prior. We show that ***MERIT matches the zero-shot MRR of SoTA KGFMs at costs comparable with those of the fastest KGFM, ULTRA***. We also show that MERIT needs both the attention over all relation names and the motifs to reach the zero-shot MRR of SoTA KGFMs (Sections C.5 and 5.1). After fine-tuning on the 22 Family 3 benchmarks, MERIT gains MRR on every half-link scenario, as ULTRA, MOTIF and FLOCK do, whereas TRIX loses MRR on the two whose answer half is unseen (Figure 6). Because its cost is comparable with that of ULTRA, we adapt MERIT to CQA as MERITQuery, and build +H benchmarks for Family 2 and Family 3 following Gregucci et al. (2025). Over the 18 +H benchmarks, MERITQuery improves over UltraQuery at matched cost (Table 3 and Section 4).

Beyond precomputation, further motif templates leave the overall costs of MERIT unchanged, so ***richer motif templates become affordable to study***, such as the higher-order motifs of MOTIF, and the motif prior can be combined with the text of relation names, as a prior rather than as the relation graph of SEMMA (Arun et al., 2025). How the MRR of MERIT grows as it is pre-trained on more graphs is left for future work, as is an optimization of the entity encoder.

AI USE STATEMENT

In this work, we used generative AI tools to polish the manuscript text, debug, and support us in coding features that we discussed and subsequently validated. We did not use generative AI tools for the idea, experiment planning, metric definition, literature review, or other required disclosure tasks. We have reviewed all AI-assisted work. In particular, we checked LLM-generated code for potential bugs and misalignment between our ideas and their implementation, and tested it for correctness. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We describe the architecture and the pre-training of MERIT in Sections A and 3, and the setup of the cost measurements in Sections B.1 and 4. For link prediction, Sections C.1, C.4 and C.5 give the pre-training runs, the fine-tuning protocol and the ablations of MERIT. For complex query answering, Sections D.3 and D.4 give the construction of the +H benchmarks and the fine-tuning of MERITQuery, and Section E gives the statistics of every benchmark. Our code is available at https://github.com/an-git-sub/MERIT; it merges the codebases of ULTRA, MOTIF, TRIX and FLOCK, so that MERIT and all baselines run from the same scripts. It also includes the checkpoints of every pre-training run of the five models and of every fine-tuning run of MERITQuery and UltraQuery, together with the +H benchmarks of Family 2 and Family 3.

REFERENCES

Ralph Abboud, İsmail İlkan Ceylan, Martin Grohe, and Thomas Lukasiewicz. The surprising power of graph neural networks with random node initialization. In Zhi-Hua Zhou (ed.), *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pp. 2112–2118. ijcai.org, 2021. doi: 10.24963/IJCAI.2021/291. URL https://doi.org/10.24963/ijcai.2021/291.

Erik Arakelyan, Daniel Daza, Pasquale Minervini, and Michael Cochez. Complex query answering with neural link predictors. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=Mos9F9kDwkz.

Arvindh Arun, Sumit Kumar, Mojtaba Nayyeri, Bo Xiong, Ponnurangam Kumaraguru, Antonio Vergari, and Steffen Staab. SEMMA: A semantic aware knowledge graph foundation model. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 31825–31848. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.1621. URL https://doi.org/10.18653/v1/2025.emnlp-main.1621.

Yushi Bai, Xin Lv, Juanzi Li, and Lei Hou. Answering complex logical queries on knowledge graphs via query computation tree optimization. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 1472–1491. PMLR, 2023. URL https://proceedings.mlr.press/v202/bai23b.html.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pp. 2787–2795, 2013. URL https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural*

Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/67d57c32e20fd0a7a302cb81d36e40d5-Abstract-Conference.html.

Juechu Dong, Boyuan Feng, Driss Guessous, Yanbo Liang, and Horace He. Flex attention: A programming model for generating optimized attention kernels. *CoRR*, abs/2412.05496, 2024. doi: 10.48550/ARXIV.2412.05496. URL https://doi.org/10.48550/arXiv.2412.05496.

Xin Luna Dong. Challenges and innovations in building a product knowledge graph. In Yike Guo and Faisal Farooq (eds.), *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pp. 2869. ACM, 2018. doi: 10.1145/3219819.3219938. URL https://doi.org/10.1145/3219819.3219938.

Michael Galkin, Zhaocheng Zhu, Hongyu Ren, and Jian Tang. Inductive logical query answering in knowledge graphs. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/6246e04dcf42baf7c71e3a65d3d93b55-Abstract-Conference.html.

Michael Galkin, Jincheng Zhou, Bruno Ribeiro, Jian Tang, and Zhaocheng Zhu. A foundation model for zero-shot logical query reasoning. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024a. URL http://papers.nips.cc/paper_files/paper/2024/hash/616521c3cf15f9f7018565c427d40e3b-Abstract-Conference.html.

Mikhail Galkin, Xinyu Yuan, Hesham Mostafa, Jian Tang, and Zhaocheng Zhu. Towards foundation models for knowledge graph reasoning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024b. URL https://openreview.net/forum?id=jVEoydFOl9.

Cosimo Gregucci, Bo Xiong, Daniel Hernández, Lorenzo Loconte, Pasquale Minervini, Steffen Staab, and Antonio Vergari. Is complex query answering really complex? In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/gregucci25a.html.

Cosimo Gregucci, Obaidah Theeb, Daniel Hernández, Antonio Vergari, and Steffen Staab. Half a link can be enough to predict a whole link: Understanding generalization in knowledge graph foundation models. *CoRR*, abs/2606.18001, 2026. doi: 10.48550/ARXIV.2606.18001. URL https://doi.org/10.48550/arXiv.2606.18001.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan F. Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs. *ACM Comput. Surv.*, 54(4):71:1–71:37, 2022. doi: 10.1145/3447772. URL https://doi.org/10.1145/3447772.

Xingyue Huang, Pablo Barceló, Michael M. Bronstein, İsmail İlkan Ceylan, Mikhail Galkin, Juan L. Reutter, and Miguel A. Romero Orth. How expressive are knowledge graph foundation models? In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of*

Machine Learning Research. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/huang25a.html.

Ihab F. Ilyas, Theodoros Rekatsinas, Vishnu Konda, Jeffrey Pound, Xiaoguang Qi, and Mohamed A. Soliman. Saga: A platform for continuous construction and serving of knowledge at scale. In Zachary G. Ives, Angela Bonifati, and Amr El Abbadi (eds.), *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pp. 2259–2272. ACM, 2022. doi: 10.1145/3514221.3526049. URL https://doi.org/10.1145/3514221.3526049.

Jinwoo Kim, Xingyue Huang, Krzysztof Olejniczak, Kyungbin Min, Michael M. Bronstein, Seunghoon Hong, and İsmail İlkan Ceylan. Flock: A knowledge graph foundation model via learning on random walks. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 36664–36705, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/3ddbd473456a57e3cafb1ee51ddf8ff6-Paper-Conference.pdf.

Jaejun Lee, Chanyoung Chung, and Joyce Jiyoung Whang. Ingram: Inductive knowledge graph embedding via relation graphs. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 18796–18809. PMLR, 2023. URL https://proceedings.mlr.press/v202/lee23c.html.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory F. Diamos, Erich Elsen, David García, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. Mixed precision training. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=r1gs9JgRZ.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. In Dawn Song, Michael Carbin, and Tianqi Chen (eds.), *Proceedings of the Sixth Conference on Machine Learning and Systems, MLSys 2023, Miami, FL, USA, June 4-8, 2023*. mlsys.org, 2023. URL https://proceedings.mlsys.org/paper_files/paper/2023/hash/c4be71ab8d24cdfb45e3d06dbfca2780-Abstract-mlsys2023.html.

Hongyu Ren and Jure Leskovec. Beta embeddings for multi-hop logical reasoning in knowledge graphs. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/e43739bba7cdb577e9e3e4e42447f5a5-Abstract.html.

Hongyu Ren, Weihua Hu, and Jure Leskovec. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=BJgr4kSFDS.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=HkgEQnRqYQ.

Komal K. Teru, Etienne G. Denis, and William L. Hamilton. Inductive relation prediction by subgraph reasoning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9448–9457. PMLR, 2020. URL http://proceedings.mlr.press/v119/teru20a.html.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In Maria-Florina Balcan and Kilian Q. Weinberger

(eds.), *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 2071–2080. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/trouillon16.html.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10524–10533. PMLR, 2020. URL http://proceedings.mlr.press/v119/xiong20b.html.

Hang Yin, Zihao Wang, and Yangqiu Song. Rethinking complex queries on knowledge graphs with neural link predictors. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=1BmveEMNbG.

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 Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 28877–28888, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/f1c1592588411002af340cbaedd6fc33-Abstract.html.

Yucheng Zhang, Beatrice Bevilacqua, Mikhail Galkin, and Bruno Ribeiro. TRIX: A more expressive model for zero-shot domain transfer in knowledge graphs. In Guy Wolf and Smita Krishnaswamy (eds.), *Learning on Graphs Conference, 26-29 November 2024, Virtual*, volume 269 of *Proceedings of Machine Learning Research*, pp. 12. PMLR, 2024. URL https://proceedings.mlr.press/v269/zhang25a.html.

Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal A. C. Xhonneux, and Jian Tang. Neural bellman-ford networks: A general graph neural network framework for link prediction. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 29476–29490, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/f6a673f09493afcd8b129a0bcf1cd5bc-Abstract.html.

Zhaocheng Zhu, Mikhail Galkin, Zuobai Zhang, and Jian Tang. Neural-symbolic models for logical queries on knowledge graphs. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 27454–27478. PMLR, 2022. URL https://proceedings.mlr.press/v162/zhu22c.html.

A MERIT IMPLEMENTATION DETAILS

Architecture. The relation transformer has $L = 6$ pre-norm encoder layers with $K = 8$ heads, token dimension 64 and feed-forward width 256, without dropout and without a final layer normalization; the motif bias matrix $B \in \mathbb{R}^{4\times 8}$ and the linear map of the signature, $W_g \in \mathbb{R}^{8\times 8}$ without bias, are initialized to zero. The entity encoder is the six-layer NBFNet of ULTRA, with DistMult messages, sum aggregation, residual connections and layer normalization, followed by a two-layer scoring MLP.

Motif tensor. Let $t \in \{1, \dots, T\}$, $T = 4$, index the motif templates h2h, t2t, h2t and t2h in this order. Our Ψ sets $M_{ijt} = \log(1+C_{ijt})$, where C_{ijt} is the number of occurrences of motif t between i and j, one per distinct entity; we take the logarithm so that counts spanning orders of magnitude do not saturate the attention. For example, in Figure 2 the t2t motif occurs once between *comprises* and *hasLandmark*, at Colosseo, and the t2h motif occurs once, at Rome, so the pair descriptor is $M_{r_1 r_2} = (0, \log 2, 0, \log 2)$ for $r_1 =$ *comprises* and $r_2 =$ *hasLandmark*. We compute M with the four sparse matrix products with which ULTRA builds its relation graph (Galkin et al., 2024b).

Token layout. A token has 64 entries: 8 hold the signature s_r, 56 hold $\mathbf{z}_{r|p}$, all ones for the query relation and standard Gaussian noise for every other relation name. The noise is redrawn at every forward pass, also at inference, so MERIT is stochastic at evaluation.

Relation signature. The fixed summary of the row M_r in $\mathrm{s}_r = W_g\, \phi(M_r)$ is

$$\phi(M_r) = \big(\max_{j\neq r} M_{rj1}, \dots, \max_{j\neq r} M_{rjT},\ \log(1+n_1), \dots, \log(1+n_T) \big),$$

with $n_t = |\{\, j \neq r : M_{rjt} > 0 \,\}|$, and $W_g \colon \mathbb{R}^{2T} \to \mathbb{R}^d$, $d = 8$, is a linear map. A maximum and a count do not depend on the order of the other relation names, so g reads the row as a multiset, as Equation (1) requires.

Pretraining. MERIT is pretrained on the three graphs of ULTRA, FB15k237, WN18RR and CoDEx Medium, with AdamW, batches of 64 queries, 512 strict negatives per query and adversarial temperature 1. The motif tensor is rebuilt on every training batch from the conditioning graph, and built once per graph at inference.

Inference kernel. For inference we follow the standard transformer setting (Micikevicius et al., 2018; Pope et al., 2023) and run the attention of the relation transformer with the FlexAttention kernel (Dong et al., 2024). The FlexAttention kernel applies the motif bias inside the attention kernel instead of materializing it as an $|\mathcal{R}_{inf}| \times |\mathcal{R}_{inf}|$ matrix per head and per query. All inference times and FLOPs of MERIT in the paper are measured on this path, at test batch size 32.

B COST MEASUREMENTS

B.1 BATCH SIZES OF THE COST MEASUREMENTS

We run all five models from one code base, which merges those of ULTRA, MOTIF, TRIX and FLOCK, so that every model runs from the same scripts (https://github.com/an-git-sub/MERIT). We measure the pre-training time per batch of every model at batch size 8 on one GPU. At inference, we run ULTRA, TRIX, MOTIF and MERIT at batch size 32, and FLOCK with the zero-shot configurations released with the code of Kim et al. (2026), which set for each benchmark the number of base walks n, the number of ensembled passes P per query and the batch size. Since the walks of all queries and all P passes in a batch are held in memory at once, the batch size of FLOCK shrinks as n grows: with $n = 512$ and $P = 2$, on CoDEx Large, DBpedia100k and Hetionet, FLOCK runs at batch size 2, and with $n = 16$ and $P = 16$, it runs at batch size 32 only on the inductive benchmarks and at 16 on the transductive ones. At the same batch size of 32, on the 34 benchmarks of Family 2 and Family 3 on which FLOCK uses $n = 16$, MERIT needs 55 seconds of inference in total against 4.67 hours for FLOCK, which takes $305\times$ the time of MERIT. We keep these configurations to preserve the performance of FLOCK, which Kim et al. (2026) show to improve with more ensembled passes and with a number of base walks adapted to the size of the graph.

B.2 INFERENCE COST BY GRAPH SIZE

Figure 5 of Section 4 bins the benchmarks by the number of relation names and of triples of the inference graph, and plots the median inference time of each model in each bin. Table B.1 lists the benchmarks in each bin, and Figure B.1 adds the bins by number of entities and, over all three, the FLOPs, the peak GPU memory and the peak CPU memory at inference.

B.3 INFERENCE RESOURCES PER STAGE

For the GNN-based KGFMs, ULTRA, MOTIF, TRIX and MERIT, we split inference into the four stages of Section 4. Figure B.2 gives the largest peak GPU memory of each stage and its share of the inference time and of the FLOPs over the 56 benchmarks of Figure 1.

B.4 INFERENCE COST WITH MORE MOTIF TEMPLATES

Figure B.3 breaks down by the number of relation names the measurement of Section 4 in which we duplicate the four slices of M to $T = 8$ and $T = 16$: for each relation-name bin of Table B.1, it plots the median resource consumption of MERIT at $T = 4$, 8 and 16.

B.5 INFERENCE COST PER QUERY TYPE

We measure for each model the inference time of link prediction per test query of FB15k237, which is the time of a *1p* query. For UltraQuery and MERITQuery, we also measure the time of *2p* and *3p* queries (Section 4), and find that a query with k hops needs about k times the time of a *1p* query. We assume that the time of TRIX, MOTIF and FLOCK, if adapted as ULTRA is in UltraQuery, would grow with the number of hops in the same way. Figure B.4 shows the resulting projection of the number of path queries each KGFM answers in one hour: ULTRA and MERIT would answer more *4p* queries in one hour than the other KGFMs answer *1p* queries.

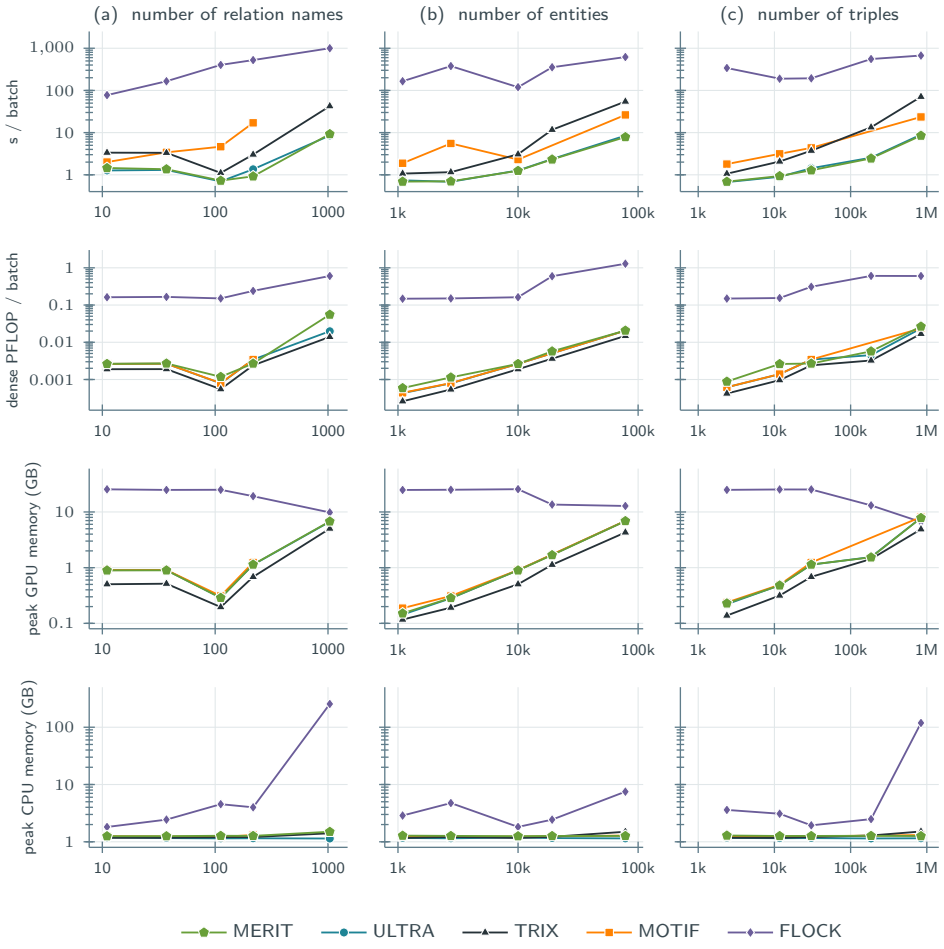

Figure B.1: **MERIT tracks ULTRA in wall-clock, GPU memory and CPU memory on every size axis**, and deviates from it only in dense FLOPs, by the largest factor on the graphs with the most relation names. Rows show inference wall-clock and FLOPs of dense tensor operators as counted by PyTorch, both per batch of 1,000 test triples, and peak GPU and CPU memory; columns bin the benchmarks by relation names, entities and triples of the inference graph (Table B.1) and markers are medians per bin. MOTIF markers are omitted from the bins that contain AristoV4, which has no MOTIF run, so that every marker in a bin is a median over the same benchmarks.

axis	bin	n	median	benchmarks
relation names	7 to 20	13	11	Wiki MT1 health, WN v1, WN v4, Wiki MT1 tax, WN v2, HM 1k, HM 3k, HM 5k, WN18RR, WN v3, Wiki MT2 org, NELL v1, Wiki MT2 sci
	20 to 60	13	37	Wiki MT4 health, Hetionet, Wiki MT3 infra, Metafam, ConceptNet100k, WK-100, YAGO310, CoDEx Small, Wiki MT4 sci, Wiki MT3 art, ILPC Small, CoDEx Medium, NL-100
	60 to 200	17	112	ILPC Large, WK-75, CoDEx Large, WK-25, NELL v4, FB-100, NELL v2, WK-50, NL-0, NL-75, NL-50, NL-25, WDsinger, NELL v3, FB v1, FBNELL, FB-75
	200 to 400	12	217	FB v2, NELL23k, NELL995, FB-50, FB v3, FB-25, FB v4, IndigoBM, FB15k237, FB15k237_10, FB15k237_20, FB15k237_50
	400 to $2k$	2	1.04k	DBpedia100k, AristoV4*
entities	200 to $2k$	7	1.09k	NELL v1, Metafam, WN v1, FB v1, NL-75, FB v2, NL-100
	$2k$ to $5k$	17	2.76k	NL-0, CoDEx Small, NELL v2, NL-25, NL-50, FB v3, FB-100, WK-75, WN v2, FB-75, NELL v4, FB v4, WK-25, NELL v3, FB-25, FB-50, FBNELL
	$5k$ to $12k$	15	10k	WN v3, ILPC Small, WN v4, WK-50, HM 1k, Wiki MT1 health, Wiki MT1 tax, Wiki MT2 org, Wiki MT2 sci, Wiki MT3 art, Wiki MT3 infra, Wiki MT4 health, Wiki MT4 sci, WDsinger, FB15k237_10
	$12k$ to $50k$	13	19.2k	WK-100, FB15k237_20, FB15k237_50, FB15k237, IndigoBM, CoDEx Medium, HM 3k, NELL23k, HM 5k, ILPC Large, WN18RR, AristoV4*, Hetionet
	$50k$ to $200k$	5	78.3k	NELL995, CoDEx Large, ConceptNet100k, DBpedia100k, YAGO310
triples	800 to $5k$	13	2.38k	NELL v1, WN v1, NL-75, FB v1, NL-25, NL-0, NL-100, NL-50, WK-25, WK-75, WN v2, FB v2, NELL v2
	$5k$ to $20k$	21	11.7k	WN v3, FB-100, NELL v4, Metafam, FB v3, NELL v3, FB-75, WK-50, FBNELL, FB-50, FB v4, WN v4, Wiki MT4 sci, WK-100, Wiki MT1 health, Wiki MT2 sci, Wiki MT4 health, WDsinger, Wiki MT1 tax, FB-25, HM 1k
	$20k$ to $100k$	12	30.5k	ILPC Small, Wiki MT3 infra, Wiki MT2 org, NELL23k, FB15k237_10, Wiki MT3 art, CoDEx Small, HM 3k, HM 5k, FB15k237_20, ILPC Large, WN18RR
	$100k$ to $500k$	7	186k	ConceptNet100k, FB15k237_50, NELL995, CoDEx Medium, AristoV4*, IndigoBM, FB15k237
	$500k$ to $3M$	4	838k	CoDEx Large, DBpedia100k, YAGO310, Hetionet

Table B.1: **Details of the bins behind Figures B.1 and 5**. Rows list the benchmarks in each log-spaced bin of relation names, entities and triples of the inference graph, with the bin count n and the median size. * no MOTIF run.

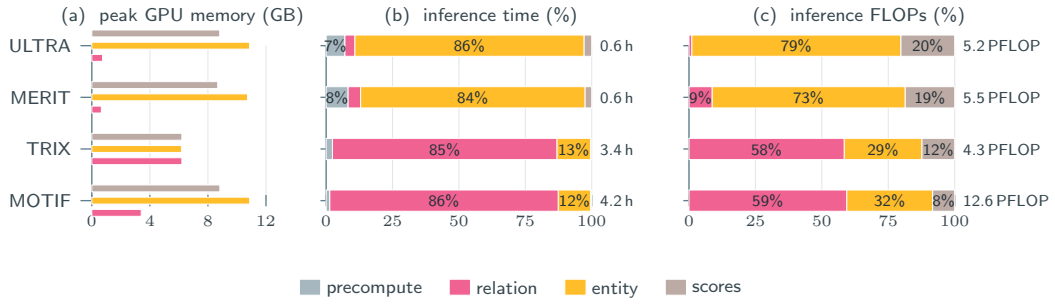

Figure B.2: **The relation stage carries most of the resource consumption of TRIX and MOTIF but a small share for MERIT and ULTRA**, whose cost sits in the shared entity GNN. Bars cover the 56 benchmarks of Figure 1 for the four stages of GNN-based KGFMs: precomputation of the relational structure, relation encoder, entity encoder and score computation. Panels show (a) the largest peak GPU memory of each stage on any benchmark, (b) the share of the wall-clock summed over the suite and (c) the share of the FLOPs summed over the suite, both with the sum printed at the end of each bar; labels give the share of the larger segments. Note that TRIX ties its relation computation to a small entity GNN, hence its relation-stage peak closely matches that of the entity stage.

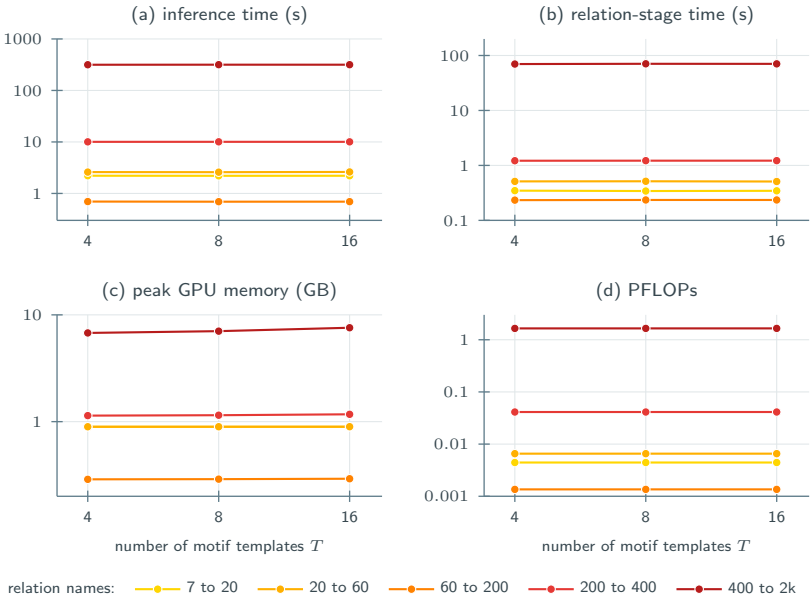

Figure B.3: **The resources of MERIT stay almost constant as the number of motif templates grows**. As indicated in Section 4, peak GPU memory increases slightly with T on the bins with the most relation names. Median resources of MERIT over the benchmarks in each relation-name bin of Table B.1, at $T = 4$ and with the four slices of M duplicated to $T = 8$ and $T = 16$. Each line corresponds to a different bin.

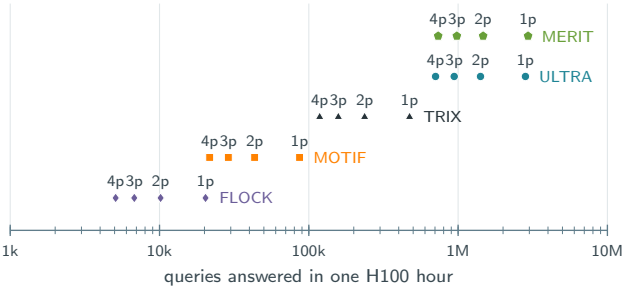

Figure B.4: **MERIT answers more *4p* queries in one hour than TRIX, MOTIF or FLOCK answer *1p* queries, and as many queries as ULTRA at every hop**.

C LINK PREDICTION

C.1 PRE-TRAINED CHECKPOINTS

The mean of every SoTA KGFM is over the checkpoint released by its authors and further pre-training runs from fresh random seeds. For ULTRA, MOTIF and TRIX we use the pre-training runs of Gregucci et al. (2026): the released checkpoint and four seeds for ULTRA and TRIX, the released checkpoint and two seeds for MOTIF. For FLOCK, the official repository releases a single checkpoint, so we pre-train two further seeds ourselves, for three runs of FLOCK. MERIT has five pre-training runs, one per seed.

C.2 ZERO-SHOT LINK PREDICTION PER BENCHMARK

Table C.1 adds to the 54 benchmarks of Section 5.1 the mean over all 57 benchmarks, and Tables C.3 and C.4 give the MRR and Hits@10 on each of the 57. FLOCK samples random walks at inference and is not seeded there, so its sample standard deviation carries both pre-training and inference variability. MERIT redraws its noise at every forward pass, also at inference (Section A), but this variability is negligible against that over pre-training runs: over five evaluation seeds of one pre-training run, its mean MRR over the 57 benchmarks has a sample standard deviation of 0.0003, against 0.0024 over the five pre-training runs. Table C.2 compares inference time and MRR on the 54 benchmarks, and shows that the conclusions of Figure 1, over the 56 benchmarks other than AristoV4 (Section 4), are unchanged on these 54.

Table C.1: **Zero-shot MRR and Hits@10 of MERIT and the SoTA KGFMs on the 54 benchmarks and on all 57.** Mean over the benchmarks of each column, averaged over five pre-training runs per model, three for MOTIF and FLOCK; $\pm$ is the sample standard deviation over runs. The single run of MOTIF on AristoV4 enters every run of MOTIF. Bold as in Table 1.

Model	54 benchmarks		57 benchmarks	
	MRR	Hits@10	MRR	Hits@10
ULTRA	$0.365_{\pm.003}$	$0.521_{\pm.003}$	$0.368_{\pm.002}$	$0.523_{\pm.004}$
MOTIF	$0.365_{\pm.002}$	$0.521_{\pm.001}$	$0.368_{\pm.001}$	$0.524_{\pm.001}$
TRIX	$\mathbf{0.384}_{\pm.002}$	$0.541_{\pm.003}$	$0.386_{\pm.002}$	$0.542_{\pm.003}$
FLOCK	$\mathbf{0.384}_{\pm.006}$	$\mathbf{0.552}_{\pm.005}$	$0.382_{\pm.005}$	$\mathbf{0.548}_{\pm.005}$
MERIT (ours)	$\mathbf{0.386}_{\pm.002}$	$\mathbf{0.549}_{\pm.001}$	$\mathbf{0.388}_{\pm.002}$	$\mathbf{0.551}_{\pm.001}$

Model	inference time (h)	MRR
ULTRA	0.60	$0.365_{\pm.003}$
MOTIF	3.69^{*}	$0.365_{\pm.002}$
TRIX	3.31	$0.384_{\pm.002}$
FLOCK	67.07	$0.384_{\pm.006}$
MERIT (ours)	0.58	$0.386_{\pm.002}$

Table C.2: **The conclusions of Figure 1 are unchanged on the 54 benchmarks in the inductive transfer setting.** Inference time over the 54 benchmarks in hours and MRR averaged over five pre-training runs per model, three for MOTIF and FLOCK; $\pm$ is the sample standard deviation over runs. The single run of MOTIF on AristoV4 enters every run of MOTIF, and * marks its time over the 53 benchmarks other than AristoV4.

Table C.3: **Zero-shot MRR of MERIT and the SoTA KGFMs on each of the 57 benchmarks**. Mean MRR over five pre-training runs per model, three for MOTIF and FLOCK; ± is the sample standard deviation over runs; * marks the single run of MOTIF on AristoV4, which enters every run of MOTIF in the last row, the average over all 57 benchmarks. **Bold** marks the best mean of each row.

Benchmark	ULTRA	MOTIF	TRIX	FLOCK	MERIT
Family 1 (16)					
CoDEx Small	0.467±.011	0.476±.005	0.478±.006	0.446±.006	**0.479**±.007
CoDEx Medium	0.368±.008	0.369±.005	0.364±.003	0.267±.007	**0.374**±.002
CoDEx Large	0.331±.004	0.342±.004	0.338±.002	0.341±.002	**0.346**±.002
WDsinger	0.385±.017	0.395±.003	0.389±.012	**0.403**±.010	**0.403**±.012
FB15k237_10	0.237±.011	0.241±.005	0.239±.008	0.216±.047	**0.248**±.002
FB15k237_20	0.267±.004	0.264±.005	0.265±.004	0.244±.049	**0.270**±.001
FB15k237_50	0.321±.002	0.317±.004	0.322±.001	0.290±.054	**0.326**±.001
FB15k237	0.368±.008	0.358±.010	0.365±.002	0.267±.008	**0.377**±.002
NELL23k	0.238±.006	0.224±.005	0.228±.008	0.219±.014	**0.249**±.006
WN18RR	0.497±.022	0.532±.012	0.512±.004	0.498±.012	**0.535**±.002
AristoV4	0.181±.016	0.096*	0.190±.009	**0.300**±.007	0.186±.053
Hetionet	0.237±.034	**0.273**±.018	0.272±.014	0.245±.003	0.237±.021
NELL995	0.453±.021	0.478±.017	0.474±.012	0.482±.010	**0.510**±.019
ConceptNet100k	0.069±.021	0.037±.024	0.217±.024	0.238±.029	**0.251**±.019
DBpedia100k	0.396±.013	0.398±.014	0.433±.008	**0.451**±.007	0.415±.012
YAGO310	**0.447**±.021	0.433±.021	0.439±.036	0.422±.036	0.384±.050
Family 2 (18)					
FB v1	0.505±.011	0.492±.015	0.510±.011	0.492±.011	**0.531**±.007
FB v2	0.513±.009	0.507±.005	0.519±.010	0.538±.006	**0.542**±.004
FB v3	0.489±.010	0.490±.008	0.494±.005	0.511±.007	**0.512**±.004
FB v4	0.482±.006	0.485±.003	0.486±.004	0.501±.002	**0.502**±.002
WN v1	0.675±.014	0.673±.007	**0.700**±.007	0.655±.053	0.689±.033
WN v2	0.674±.002	0.659±.004	0.685±.004	0.647±.059	**0.696**±.008
WN v3	0.385±.005	0.425±.005	0.433±.010	0.458±.025	**0.460**±.011
WN v4	0.635±.003	0.632±.008	0.649±.001	0.649±.004	**0.655**±.004
NELL v1	0.695±.053	0.677±.106	**0.798**±.009	0.675±.017	0.632±.113
NELL v2	0.528±.016	0.547±.015	0.553±.009	0.565±.024	**0.570**±.015
NELL v3	0.527±.022	0.502±.031	0.546±.011	**0.578**±.019	0.571±.012
NELL v4	0.492±.014	0.479±.022	0.536±.007	**0.543**±.011	0.533±.007
ILPC Small	0.294±.006	0.296±.001	0.301±.002	**0.309**±.001	0.303±.003
ILPC Large	0.296±.003	0.288±.004	0.301±.009	**0.318**±.001	0.306±.004
HM 1k	0.075±.007	0.070±.007	0.066±.007	0.067±.001	**0.081**±.005
HM 3k	0.062±.003	0.057±.003	0.063±.005	0.067±.003	**0.071**±.001
HM 5k	0.056±.003	0.052±.004	0.059±.004	**0.067**±.003	**0.067**±.000
IndigoBM	0.438±.007	0.434±.007	0.426±.008	0.422±.005	**0.447**±.002
Family 3 (23)					
FB-25	0.391±.004	0.386±.002	0.383±.006	**0.402**±.001	0.397±.004
FB-50	0.340±.005	0.339±.001	0.315±.013	**0.352**±.003	0.338±.007
FB-75	0.401±.004	0.394±.005	0.392±.006	**0.416**±.005	0.399±.005
FB-100	0.436±.006	0.423±.006	0.437±.005	**0.452**±.002	0.439±.003
WK-25	0.297±.008	**0.312**±.005	0.304±.005	0.273±.010	0.305±.007
WK-50	0.159±.010	**0.165**±.002	**0.165**±.004	0.128±.013	0.162±.011
WK-75	**0.379**±.006	0.367±.001	0.369±.005	0.377±.008	**0.379**±.011
WK-100	0.172±.005	0.169±.010	0.182±.005	**0.184**±.004	0.182±.010
NL-0	0.355±.009	0.324±.004	0.373±.011	0.376±.013	**0.377**±.011
NL-25	0.356±.026	0.361±.021	**0.389**±.011	0.339±.014	0.375±.022
NL-50	0.379±.014	0.372±.011	0.386±.016	0.341±.024	**0.389**±.029
NL-75	0.330±.021	0.321±.010	**0.346**±.004	0.291±.025	0.313±.034
NL-100	0.453±.016	0.444±.005	**0.461**±.010	0.429±.030	0.456±.024
Metafam	0.349±.089	0.374±.029	0.320±.028	0.475±.023	**0.504**±.075
FBNELL	0.480±.014	0.457±.011	0.474±.017	0.492±.019	**0.495**±.014
Wiki MT1 tax	0.234±.006	0.290±.038	**0.343**±.029	0.283±.012	0.342±.049
Wiki MT1 health	0.271±.039	0.299±.062	0.373±.005	**0.384**±.001	0.376±.006
Wiki MT2 org	0.085±.002	0.084±.008	0.093±.003	**0.100**±.001	0.094±.004
Wiki MT2 sci	0.261±.011	0.270±.039	0.317±.007	0.314±.003	**0.319**±.024
Wiki MT3 art	0.254±.004	0.270±.004	0.292±.009	**0.302**±.003	0.293±.003
Wiki MT3 infra	0.623±.016	0.632±.024	0.646±.007	**0.676**±.006	0.613±.045
Wiki MT4 sci	0.284±.009	0.273±.010	0.287±.003	**0.298**±.010	0.289±.011
Wiki MT4 health	0.589±.053	0.639±.011	**0.685**±.006	0.684±.009	0.549±.064
Average (57)	0.368±.002	0.368±.001	0.386±.002	0.382±.005	**0.388**±.002

Table C.4: **Zero-shot Hits@10 of MERIT and the SoTA KGFMs on each of the 57 benchmarks**. Mean Hits@10 over five pre-training runs per model, three for MOTIF and FLOCK; ± is the sample standard deviation over runs; * marks the single run of MOTIF on AristoV4, which enters every run of MOTIF in the last row, the average over all 57 benchmarks. **Bold** marks the best mean of each row.

Benchmark	ULTRA	MOTIF	TRIX	FLOCK	MERIT
Family 1 (16)					
CoDEx Small	0.664±.007	**0.673**±.004	**0.673**±.003	0.642±.004	0.671±.004
CoDEx Medium	0.520±.016	0.519±.005	0.521±.002	0.399±.009	**0.531**±.003
CoDEx Large	0.461±.008	0.471±.006	0.472±.002	0.465±.003	**0.479**±.003
WDsinger	0.496±.015	0.512±.003	0.499±.014	**0.526**±.008	0.513±.007
FB15k237_10	0.383±.014	0.390±.005	0.385±.012	0.360±.069	**0.400**±.003
FB15k237_20	0.427±.005	0.427±.004	0.425±.006	0.404±.069	**0.433**±.002
FB15k237_50	0.520±.002	0.514±.007	0.520±.001	0.480±.072	**0.526**±.002
FB15k237	0.559±.012	0.555±.010	0.559±.003	0.421±.013	**0.572**±.002
NELL23k	0.402±.009	0.386±.002	0.381±.012	0.377±.018	**0.417**±.005
WN18RR	0.593±.031	0.634±.011	0.608±.002	0.602±.009	**0.642**±.001
AristoV4	0.260±.024	0.142*	0.289±.012	**0.431**±.010	0.284±.093
Hetionet	0.342±.046	0.397±.012	**0.404**±.015	0.372±.001	0.361±.028
NELL995	0.595±.021	0.621±.001	0.625±.010	0.634±.020	**0.641**±.008
ConceptNet100k	0.137±.042	0.078±.063	0.380±.047	0.438±.052	**0.441**±.028
DBpedia100k	0.576±.006	0.569±.007	0.602±.004	**0.625**±.002	0.605±.003
YAGO310	0.620±.024	0.604±.026	0.640±.021	**0.655**±.018	0.588±.065
Family 2 (18)					
FB v1	0.680±.015	0.678±.013	0.676±.010	0.695±.006	**0.698**±.009
FB v2	0.715±.012	0.713±.003	0.723±.007	**0.738**±.002	0.733±.004
FB v3	0.658±.012	0.665±.005	0.665±.006	**0.687**±.002	0.677±.004
FB v4	0.676±.007	0.677±.002	0.682±.003	**0.701**±.002	0.690±.001
WN v1	0.775±.010	0.774±.006	**0.800**±.005	0.793±.008	**0.800**±.015
WN v2	0.770±.005	0.770±.008	0.782±.003	0.780±.008	**0.791**±.006
WN v3	0.499±.011	0.548±.009	0.550±.008	**0.604**±.015	0.600±.012
WN v4	0.708±.004	0.707±.011	0.723±.002	0.727±.004	**0.732**±.004
NELL v1	0.785±.053	0.833±.108	**0.904**±.028	0.871±.022	0.864±.050
NELL v2	0.729±.013	0.756±.015	0.748±.013	**0.777**±.020	0.765±.010
NELL v3	0.703±.027	0.697±.031	0.732±.012	**0.777**±.016	0.766±.008
NELL v4	0.730±.018	0.722±.010	0.758±.012	**0.777**±.008	0.769±.005
ILPC Small	0.444±.006	0.448±.003	0.452±.003	**0.463**±.002	0.455±.003
ILPC Large	0.420±.004	0.421±.006	0.424±.004	**0.439**±.001	0.429±.003
HM 1k	0.128±.009	0.123±.023	0.120±.018	0.123±.008	**0.134**±.011
HM 3k	0.114±.007	0.100±.014	0.109±.011	0.122±.006	**0.128**±.007
HM 5k	0.103±.007	0.088±.013	0.101±.010	**0.124**±.008	0.118±.006
IndigoBM	0.646±.006	0.639±.002	0.643±.008	0.635±.003	**0.653**±.002
Family 3 (23)					
FB-25	0.642±.007	0.636±.003	0.643±.005	**0.663**±.000	0.648±.007
FB-50	0.547±.005	0.544±.003	0.528±.013	**0.564**±.004	0.547±.008
FB-75	0.607±.006	0.603±.009	0.597±.010	**0.624**±.001	0.601±.009
FB-100	0.633±.006	0.618±.009	0.633±.006	**0.660**±.001	0.628±.005
WK-25	0.502±.013	**0.504**±.011	0.493±.008	0.480±.012	0.502±.016
WK-50	0.301±.010	**0.314**±.002	0.307±.006	0.271±.018	0.313±.007
WK-75	**0.542**±.007	0.529±.018	0.511±.004	0.539±.007	0.531±.016
WK-100	0.288±.010	0.288±.007	0.292±.005	**0.301**±.007	0.298±.005
NL-0	0.547±.028	0.511±.013	0.539±.008	**0.594**±.010	0.580±.012
NL-25	0.544±.030	0.509±.010	**0.582**±.009	0.564±.031	0.568±.041
NL-50	**0.552**±.006	0.523±.018	0.548±.007	0.537±.030	0.549±.032
NL-75	**0.521**±.019	0.509±.013	0.520±.005	0.504±.033	0.504±.041
NL-100	0.655±.017	0.650±.003	0.670±.010	**0.673**±.024	0.666±.028
Metafam	0.809±.066	0.792±.060	0.826±.053	0.928±.004	**0.963**±.024
FBNELL	0.650±.016	0.646±.026	0.655±.013	**0.697**±.026	0.674±.010
Wiki MT1 tax	0.304±.010	0.426±.047	**0.453**±.015	0.377±.016	0.447±.061
Wiki MT1 health	0.397±.037	0.421±.023	0.456±.003	**0.475**±.003	0.461±.008
Wiki MT2 org	0.148±.006	0.142±.012	0.154±.004	**0.165**±.002	0.155±.006
Wiki MT2 sci	0.395±.046	0.434±.052	0.464±.016	0.443±.005	**0.474**±.047
Wiki MT3 art	0.418±.007	0.418±.007	0.447±.008	**0.469**±.001	0.444±.007
Wiki MT3 infra	0.763±.021	0.772±.013	0.785±.010	**0.818**±.004	0.775±.016
Wiki MT4 sci	0.448±.007	0.439±.012	0.452±.005	**0.470**±.014	0.454±.016
Wiki MT4 health	0.734±.029	0.769±.006	0.775±.004	**0.780**±.002	0.719±.038
Average (57)	0.523±.004	0.524±.001	0.542±.003	0.548±.005	**0.551**±.001

C.3 LINK PREDICTION PER HALF-LINK SCENARIO

For a test triple (h, r, t), the query half $(h, r, ?)$ is seen iff $\exists\, e \in \mathcal{V}_{inf} \setminus \{t\} : (h, r, e) \in \mathcal{E}_{inf}$, and the answer half $(?, r, t)$ iff $\exists\, e \in \mathcal{V}_{inf} \setminus \{h\} : (e, r, t) \in \mathcal{E}_{inf}$ (Gregucci et al., 2026). Crossing the two outcomes gives the four scenarios SQSA (Seen Query, Seen Answer), SQUA (Seen Query, Unseen Answer), UQSA (Unseen Query, Seen Answer) and UQUA (Unseen Query, Unseen Answer). The 50 zero-shot benchmarks of Table 2 are the 54 of Section 5.1 without NELL995, NELL v1 and Metafam, on which at least one scenario is empty, and AristoV4, on which MOTIF has a single run (Gregucci et al., 2026). In Table C.5, we aggregate the zero-shot and fine-tuned MRR per scenario over 22 of the 23 Family 3 benchmarks, without Metafam, on which at least one scenario is empty.

Table C.5: **Fine-tuning does not lower the MRR of MERIT on SQUA and UQUA, as it does for TRIX**. Mean MRR over the 22 Family 3 benchmarks of Section 5.1, zero-shot and after per-benchmark fine-tuning, averaged over five pre-training runs per model, three for FLOCK and MOTIF; $\pm$ is the sample standard deviation over runs. Bold as in Table 1.

model	Orig	SQSA	UQSA	SQUA	UQUA
zero-shot					
ULTRA	$0.342_{\pm 0.004}$	$\mathbf{0.381}_{\pm 0.006}$	$0.543_{\pm 0.011}$	$0.113_{\pm 0.007}$	$\mathbf{0.221}_{\pm 0.013}$
MOTIF	$0.345_{\pm 0.006}$	$0.370_{\pm 0.003}$	$0.547_{\pm 0.010}$	$0.132_{\pm 0.009}$	$0.214_{\pm 0.017}$
TRIX	$\mathbf{0.364}_{\pm 0.003}$	$0.372_{\pm 0.004}$	$\mathbf{0.573}_{\pm 0.006}$	$\mathbf{0.163}_{\pm 0.007}$	$\mathbf{0.221}_{\pm 0.014}$
FLOCK	$\mathbf{0.359}_{\pm 0.008}$	$\mathbf{0.373}_{\pm 0.011}$	$0.561_{\pm 0.009}$	$0.155_{\pm 0.004}$	$\mathbf{0.231}_{\pm 0.017}$
MERIT	$0.358_{\pm 0.005}$	$\mathbf{0.380}_{\pm 0.004}$	$\mathbf{0.568}_{\pm 0.014}$	$0.144_{\pm 0.008}$	$\mathbf{0.233}_{\pm 0.011}$
fine-tuned					
ULTRA	$0.375_{\pm 0.004}$	$0.396_{\pm 0.002}$	$0.579_{\pm 0.003}$	$0.155_{\pm 0.012}$	$0.251_{\pm 0.002}$
MOTIF	$0.373_{\pm 0.003}$	$0.395_{\pm 0.002}$	$0.585_{\pm 0.002}$	$0.151_{\pm 0.008}$	$0.239_{\pm 0.004}$
TRIX	$0.368_{\pm 0.003}$	$0.396_{\pm 0.004}$	$0.593_{\pm 0.003}$	$0.126_{\pm 0.008}$	$0.196_{\pm 0.010}$
FLOCK	$\mathbf{0.391}_{\pm 0.001}$	$\mathbf{0.404}_{\pm 0.002}$	$\mathbf{0.599}_{\pm 0.002}$	$\mathbf{0.177}_{\pm 0.002}$	$\mathbf{0.258}_{\pm 0.002}$
MERIT	$0.384_{\pm 0.001}$	$\mathbf{0.402}_{\pm 0.002}$	$0.588_{\pm 0.002}$	$0.167_{\pm 0.003}$	$0.255_{\pm 0.002}$

C.4 FINE-TUNED LINK PREDICTION PER BENCHMARK

We follow the fine-tuning protocol of ULTRA (Galkin et al., 2024b): three epochs over the triples of $\mathcal{G}_{tr}$, keeping the checkpoint with the best validation MRR out of the three. Tables C.6 and C.7 give the fine-tuned MRR and Hits@10 on each of the 23 Family 3 benchmarks of Section 5.1.

Table C.6: **Fine-tuned MRR of MERIT and the SoTA KGFMs on each of the 23 Family 3 benchmarks**. Mean MRR after per-benchmark fine-tuning, averaged over five pre-training runs per model, three for MOTIF and FLOCK; ± is the sample standard deviation over runs. **Bold** marks the best mean of each row.

Benchmark	ULTRA	MOTIF	TRIX	FLOCK	MERIT
FB-25	0.387±.003	0.386±.004	0.379±.003	**0.404**±.000	0.394±.004
FB-50	0.337±.002	0.335±.003	0.323±.007	**0.357**±.004	0.339±.002
FB-75	0.406±.004	0.400±.003	0.390±.003	**0.423**±.002	0.409±.004
FB-100	0.439±.004	0.438±.003	0.433±.011	**0.458**±.001	0.453±.003
WK-25	0.308±.005	**0.316**±.004	0.291±.002	0.290±.006	0.307±.004
WK-50	0.146±.004	**0.153**±.011	0.147±.008	0.136±.002	0.149±.010
WK-75	0.378±.004	0.376±.016	0.368±.005	**0.405**±.002	0.375±.004
WK-100	0.163±.004	0.171±.005	0.177±.008	**0.183**±.002	0.166±.007
NL-0	0.339±.004	0.319±.002	0.335±.012	**0.395**±.012	0.359±.014
NL-25	0.396±.008	0.381±.009	0.371±.007	**0.398**±.002	0.395±.008
NL-50	0.412±.006	0.408±.003	0.401±.005	0.403±.006	**0.413**±.003
NL-75	**0.373**±.004	0.355±.007	0.341±.002	0.350±.013	0.366±.004
NL-100	0.468±.003	0.463±.007	0.466±.004	**0.489**±.004	0.472±.006
Metafam	0.998±.002	**1.000**±.000	0.991±.011	**1.000**±.000	**1.000**±.000
FBNELL	0.483±.014	0.474±.016	0.480±.004	**0.524**±.004	0.502±.005
Wiki MT1 tax	0.390±.071	0.398±.083	0.402±.011	0.428±.002	**0.442**±.007
Wiki MT1 health	0.379±.005	0.384±.000	0.380±.001	**0.390**±.002	0.385±.002
Wiki MT2 org	0.103±.001	0.105±.000	0.099±.001	**0.106**±.000	0.104±.001
Wiki MT2 sci	0.352±.027	0.332±.002	0.343±.006	**0.389**±.003	0.365±.014
Wiki MT3 art	0.314±.002	0.317±.001	0.277±.006	**0.329**±.002	0.322±.002
Wiki MT3 infra	0.666±.004	0.690±.002	0.683±.006	**0.709**±.000	0.697±.002
Wiki MT4 sci	0.310±.002	0.310±.004	0.306±.004	**0.330**±.001	0.315±.001
Wiki MT4 health	0.701±.003	0.703±.002	0.701±.003	0.708±.003	**0.710**±.001
Average (23)	0.402±.004	0.401±.003	0.395±.003	**0.418**±.001	0.410±.001

Table C.7: **Fine-tuned Hits@10 of MERIT and the SoTA KGFMs on each of the 23 Family 3 benchmarks**. Mean Hits@10 after per-benchmark fine-tuning, averaged over five pre-training runs per model, three for MOTIF and FLOCK; ± is the sample standard deviation over runs. **Bold** marks the best mean of each row.

Benchmark	ULTRA	MOTIF	TRIX	FLOCK	MERIT
FB-25	0.640±.005	0.633±.006	0.629±.006	**0.664**±.001	0.645±.004
FB-50	0.543±.003	0.538±.007	0.526±.008	**0.570**±.002	0.547±.005
FB-75	0.610±.002	0.599±.007	0.586±.006	**0.631**±.001	0.611±.004
FB-100	0.642±.005	0.635±.006	0.628±.010	**0.666**±.006	0.648±.004
WK-25	0.497±.010	0.505±.004	0.491±.009	0.465±.013	**0.513**±.005
WK-50	**0.298**±.003	0.289±.007	0.288±.010	0.270±.022	0.290±.018
WK-75	0.522±.014	0.535±.011	0.512±.008	**0.558**±.002	0.530±.008
WK-100	0.277±.003	0.286±.004	0.290±.005	**0.300**±.006	0.286±.008
NL-0	0.555±.023	0.542±.004	0.516±.013	**0.626**±.014	0.583±.030
NL-25	0.606±.012	0.567±.011	0.528±.019	**0.615**±.011	0.602±.021
NL-50	**0.593**±.006	0.574±.011	0.549±.006	0.591±.010	0.588±.006
NL-75	**0.574**±.002	0.552±.003	0.512±.008	0.555±.011	0.566±.006
NL-100	0.678±.006	0.677±.007	0.672±.012	**0.718**±.002	0.691±.009
Metafam	**1.000**±.000	**1.000**±.000	**1.000**±.000	**1.000**±.000	**1.000**±.000
FBNELL	0.650±.011	0.637±.010	0.641±.010	**0.709**±.007	0.676±.012
Wiki MT1 tax	0.498±.073	0.516±.027	0.518±.001	0.508±.003	**0.537**±.002
Wiki MT1 health	0.466±.010	0.469±.002	0.474±.004	**0.489**±.001	0.473±.005
Wiki MT2 org	0.169±.000	0.173±.003	0.163±.002	**0.176**±.002	0.175±.002
Wiki MT2 sci	**0.544**±.002	0.528±.009	0.532±.006	0.537±.006	**0.544**±.011
Wiki MT3 art	0.472±.004	0.468±.004	0.410±.012	**0.488**±.003	0.478±.004
Wiki MT3 infra	0.810±.003	0.827±.002	0.820±.005	**0.840**±.001	0.828±.002
Wiki MT4 sci	0.486±.003	0.483±.008	0.475±.002	**0.509**±.003	0.485±.003
Wiki MT4 health	0.784±.003	0.786±.002	0.783±.002	**0.790**±.000	0.788±.002
Average (23)	0.561±.004	0.557±.002	0.545±.004	**0.577**±.001	0.569±.001

C.5 Ablations of MERIT

Table C.8 gives the MRR of every variant of MERIT in Section 5.1. With the *motif bias only*, the tokens carry no relation signature, and with the *relation signature only*, the attention logits carry no motif bias. Every variant keeps the noise and the query flag of Equation (2), and is pre-trained as MERIT is (Section A).

Table C.8: **Both the attention over all relation names and the motifs as a soft prior contribute to the MRR of MERIT**. Mean MRR over the 54 benchmarks of Section 5.1, averaged over the pre-training runs in the second column; $\pm$ is the sample standard deviation over runs.

Model	runs	MRR
MERIT (motif bias and relation signature)	5	$0.386_{\pm.002}$
motif bias only	3	$0.379_{\pm.002}$
relation signature only	1	$0.292_{\pm.000}$
neither (plain transformer)	1	$0.312_{\pm.000}$
ULTRA	5	$0.365_{\pm.003}$
masked MERIT	3	$0.368_{\pm.007}$

D COMPLEX QUERY ANSWERING

D.1 COMPLEX QUERIES AND QUERY TYPES

Figure D.1 draws the 15 complex query types on which we evaluate CQA, and Table E.3 counts the queries of each type in every benchmark. Following Section 2, a_i are anchor entities, r_i relation names, y_i existentially quantified variables, to be grounded into entities, and $?x$ the target variable. The path types chain one to four links,

$$?x : (a_1, r_1, x), \tag{1p}$$
$$?x : \exists\, y_1.\ (a_1, r_1, y_1) \wedge (y_1, r_2, x), \tag{2p}$$

where 3p and 4p extend the chain with y_2 and y_3. The intersection types take the intersection of the target entity sets defined over two to four links,

$$?x : (a_1, r_1, x) \wedge (a_2, r_2, x), \tag{2i}$$

where 3i and 4i add a third and a fourth link. Combining a path and an intersection gives

$$?x : \exists\, y_1.\ (a_1, r_1, y_1) \wedge (a_2, r_2, y_1) \wedge (y_1, r_3, x), \tag{2i1p}$$

and 1p2i inverts the order of operations. The union types replace a conjunction with a disjunction,

$$?x : (a_1, r_1, x) \vee (a_2, r_2, x), \tag{2u}$$

and 2u1p follows the disjunction with a link. The negation types negate a triple pattern of a query type above,

$$?x : (a_1, r_1, x) \wedge \neg(a_2, r_2, x), \tag{2in}$$

where 3in, 2in1p and 2pi1pn negate one triple pattern of 3i, 2i1p and 1p2i, and 2nu1p negates the whole 2p path of 1p2i, $\neg((a_1, r_1, y_1) \wedge (y_1, r_2, x))$, which De Morgan laws rewrite as $\neg(a_1, r_1, y_1) \vee \neg(y_1, r_2, x)$. In Figure D.1 and Tables 3 and E.3, the conjunctive types of Section 5.2 are the path types, the intersection types, 1p2i and 2i1p; the union types are 2u and 2u1p; and the negation types are 2in, 3in, 2in1p, 2pi1pn and 2nu1p.

We name the query types as Gregucci et al. (2025) do, writing the number of steps of each path and of each conjunction.

D.2 RELATED WORK ON CQA

As in link prediction (Section 2), CQA methods such as CQD (Arakelyan et al., 2021), QTO (Bai et al., 2023) and FIT (Yin et al., 2024) rely on link predictors that learn an embedding for every entity and relation name. GNN-QE (Zhu et al., 2022) generalizes to unseen entities, but still learns embeddings for relation names. Hence, none of these methods can answer queries in the inductive transfer setting.

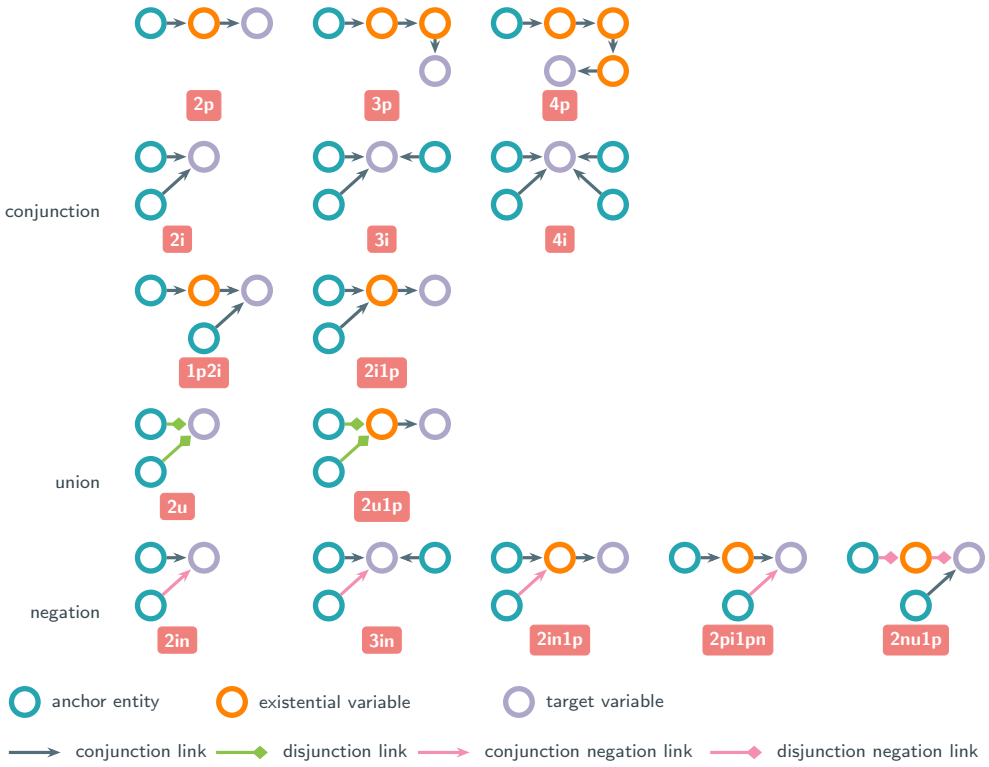

Figure D.1: **The 15 complex query types on which we evaluate CQA**, grouped as in Tables 3 and E.3. Structures adapted from Ren & Leskovec (2020), with the 4p and 4i types of Gregucci et al. (2025).

D.3 THE +H BENCHMARKS OF FAMILY 2 AND FAMILY 3

Gregucci et al. (2025) call a QA pair *full-inference* if its reasoning tree contains only links of the test graph, and *partial-inference* if it contains at least one link of the training graph and at least one of the test graph, in which case the pair reduces to a simpler query type. Their +H benchmarks balance QA pairs: for each query type, every reduction, that is each simpler query type it reduces to and full-inference, holds $10{,}000$ QA pairs, and no anchor entity and no relation name appears in more than 20% of the QA pairs of a query type. Following this construction, we rebuild the Family 2 and Family 3 benchmarks of UltraQuery (Galkin et al., 2022; 2024a) as +H benchmarks, where the inference graph $\mathcal{G}_{inf}$ takes the place of the training graph and the test graph $\mathcal{G}_t$ holds the links to predict, and keep the +H benchmarks of Family 1 as released.

Similarly to Gregucci et al. (2025), we sample queries for one reduction at a time and assign every QA pair to the reduction it falls in, drawing the links of the reasoning tree from $\mathcal{G}_{inf}$ or from $\mathcal{G}_t$ as the reduction prescribes. Then, as the inference graphs of Family 3 carry far fewer relation names than the graphs of Family 1, their most frequent relation name covers 11.7% (Science) to 45.9% (Location) of the links, against 5.6% (FB15k237) to 15.9% (ICEWS18). A cap of 20% would leave few QA pairs to the query types with such a relation name, so we raise it to 25% of the QA pairs of a reduction, on both families. We fill every reduction of a query type with the same number of QA pairs, the largest number at most $10{,}000$ that every reduction of the type reaches under this cap; it goes from $10{,}000$ down to 4 for the 4p and 4i queries of Location. We keep a graph only if every query type has QA pairs under this cap, which excludes the Award, Education, Organization and Sport graphs of UltraQuery from Family 3. The 1p queries are the 1p test queries of UltraQuery, unchanged, as in the +H benchmarks of Family 1 (Gregucci et al., 2025).

D.4 EXPERIMENTAL DETAILS

We fine-tune UltraQuery and MERITQuery with the configuration that Galkin et al. (2024a) release for UltraQuery, and change only the starting checkpoints: the pre-training runs of ULTRA and of MERIT on the three graphs of Section 5.1, in place of the checkpoint pre-trained on four graphs (Galkin et al., 2024b) from which Galkin et al. (2024a) start, so that CQA starts from the same five pre-training runs per model as link prediction. Each fine-tuning run trains on the training queries of FB15k237 (Table E.3) for ten epochs of $4{,}000$ steps, $40{,}000$ steps in total, with Adam at a learning rate of 5×10^{-4} and batches of 32 queries on each of four GPUs. We answer the queries of all 19 benchmarks with the fuzzy logic operators of UltraQuery.

D.5 RESULTS PER QUERY TYPE

Figure D.2 breaks down Table 3 by query type in each family, and includes the 1p queries and FB15k237+H.



Figure D.2: **MERITQuery is ahead of UltraQuery on every query type of Family 3, and on** 12 **and** 11 **of the** 16 **of Family 1 and Family 2**. Mean MRR over the benchmarks of each family per query type, averaged over the five fine-tuning runs per method of Section 5.2, with FB15k237+H in Family 1; error bars are the sample standard deviation over runs. 1p appears only here.

E BENCHMARK STATISTICS

Table E.1 lists by family the 57 link-prediction benchmarks, the 54 of Section 5.1 and the three pre-training graphs; Table E.2 lists the graphs behind the 19 CQA benchmarks, the 18 of Section 5.2 and FB15k237+H; and Table E.3 counts the queries of each query type in each of them. All counts of relation names and triples are those of the original graphs: every model adds to $\mathcal{G}_{inf}$ the inverse of each triple (Section 2), and so works with twice as many relation names and triples.

Table E.1: **The 57 link-prediction benchmarks**. Entities, relation names and triples of the inference graph, and the number of test triples, as reported by Galkin et al. (2024b).

Benchmark	Entities	Relation names	Triples	Test triples
Family 1 (16)				
CoDEx Small	2,034	42	32,888	1,828
CoDEx Medium	17,050	51	185,584	10,311
CoDEx Large	77,951	69	551,193	30,622
WDsinger	10,282	135	16,142	2,203
FB15k237_10	11,512	237	27,211	18,150
FB15k237_20	13,166	237	54,423	19,776
FB15k237_50	14,149	237	136,057	20,324
FB15k237	14,541	237	272,115	20,466
NELL23k	22,925	200	25,445	4,952
WN18RR	40,943	11	86,835	3,134
AristoV4	44,949	1,605	242,567	20,000
Hetionet	45,158	24	2,025,177	112,510
NELL995	74,536	200	149,678	2,818
ConceptNet100k	78,334	34	100,000	1,200
DBpedia100k	99,604	470	597,572	50,000
YAGO310	123,182	37	1,079,040	5,000
Family 2 (18)				
FB v1	1,093	180	1,993	411
FB v2	1,660	200	4,145	947
FB v3	2,501	215	7,406	1,731
FB v4	3,051	219	11,714	2,840
WN v1	922	9	1,618	373
WN v2	2,757	10	4,011	852
WN v3	5,084	11	6,327	1,143
WN v4	7,084	9	12,334	2,823
NELL v1	225	14	833	201
NELL v2	2,086	88	4,586	935
NELL v3	3,566	142	8,048	1,620
NELL v4	2,795	76	7,073	1,447
ILPC Small	6,653	48	20,960	2,902
ILPC Large	29,246	65	77,044	10,184
HM 1k	9,899	11	18,638	476
HM 3k	19,218	11	38,285	1,349
HM 5k	23,792	11	48,425	2,124
IndigoBM	14,775	229	250,195	14,904
Family 3 (23)				
FB-25	4,097	216	17,147	5,716
FB-50	4,445	205	11,636	3,879
FB-75	2,792	186	9,316	3,106
FB-100	2,624	77	6,987	2,329
WK-25	3,228	74	3,391	1,131
WK-50	9,328	93	9,672	3,225
WK-75	2,722	65	3,430	1,144
WK-100	12,136	37	13,487	4,496
NL-0	2,026	112	2,287	763
NL-25	2,146	120	2,230	744
NL-50	2,335	119	2,576	859
NL-75	1,578	116	1,818	607
NL-100	1,709	53	2,378	793
Metafam	656	28	7,257	184
FBNELL	4,752	183	10,685	597
Wiki MT1 tax	10,000	9	16,526	1,834
Wiki MT1 health	10,000	7	14,110	1,566
Wiki MT2 org	10,000	11	21,976	2,441
Wiki MT2 sci	10,000	16	14,852	1,650
Wiki MT3 art	10,000	45	28,023	3,113
Wiki MT3 infra	10,000	27	21,646	2,405
Wiki MT4 sci	10,000	42	12,516	1,388
Wiki MT4 health	10,000	20	15,337	1,703

Table E.2: **The graphs behind the** 19 **CQA benchmarks**, those of Gregucci et al. (2025) for Family 1 and of UltraQuery (Galkin et al., 2024a) for Family 2 and Family 3. Entities and relation names are those of the inference graph.

Dataset	Entities	Relation names	Training triples	Inference triples	Test triples
Family 1					
FB15k237+H	14,505	237	272,115	–	20,438
NELL995+H	63,361	200	114,213	–	14,267
ICEWS18+H	20,840	250	213,304	–	24,689
Family 2					
FB 106%+H	13,802	233	246,713	269,449	4,012
FB 113%+H	13,021	234	200,839	243,035	7,447
FB 122%+H	12,314	233	149,440	215,447	11,645
FB 134%+H	11,471	233	114,365	183,906	12,265
FB 150%+H	10,782	231	81,342	165,677	14,878
FB 175%+H	9,781	218	51,261	133,248	14,446
FB 217%+H	9,058	223	26,228	106,194	14,089
FB 300%+H	8,266	206	12,220	93,579	14,329
FB 550%+H	7,275	156	2,633	66,763	11,252
Family 3					
WikiTopics Art+H	10,000	45	27,262	28,023	3,113
WikiTopics Health+H	10,000	20	15,539	15,337	1,703
WikiTopics Infrastructure+H	10,000	27	21,990	21,646	2,405
WikiTopics Location+H	10,000	34	85,063	80,269	8,917
WikiTopics People+H	10,000	25	55,698	58,530	6,503
WikiTopics Science+H	10,000	42	12,576	12,516	1,388
WikiTopics Taxonomy+H	10,000	31	18,921	19,416	2,157

Table E.3: **The queries of the** 19 **CQA benchmarks**. Number of test queries per query type, and of training and validation queries of FB15k237, the fine-tuning dataset of UltraQuery and MERIT-Query; – marks a query type without queries in the split.

	conjunction									union		negation				
Dataset	1p	2p	3p	4p	2i	3i	4i	1p2i	2i1p	2u	2u1p	2in	3in	2in1p	2pi1pn	2nu1p
FB15k237, the fine-tuning dataset																
Training	149,689	149,689	149,689	–	149,689	149,689	–	–	–	–	–	14,968	14,968	14,968	14,968	14,968
Validation	20,094	5,000	5,000	–	5,000	5,000	–	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000	5,000
Family 1																
FB15k237+H	22,804	936	1,461	2,670	6,692	9,615	20,189	1,586	604	6,695	473	5,000	3,057	2,663	2,093	5,000
NELL995+H	17,021	1,054	1,583	2,322	8,278	13,118	22,326	1,957	1,260	8,168	1,073	4,000	5,236	2,708	1,995	4,000
ICEWS18+H	30,605	2,125	3,582	4,338	11,918	27,252	38,767	14,746	5,140	7,516	3,855	1,548	7,811	2,875	2,919	1,508
Family 2																
FB 106%+H	5,446	5,960	2,340	196	8,227	10,102	9,086	23,013	3,481	5,942	1,273	5,446	1,119	2,833	1,037	5,446
FB 113%+H	9,782	3,607	2,354	142	13,156	23,277	18,671	23,753	5,646	9,361	2,310	6,379	2,272	1,232	1,301	6,379
FB 122%+H	14,458	2,378	5,810	854	9,498	16,408	24,835	18,185	7,424	7,942	4,794	4,978	3,931	884	970	4,976
FB 134%+H	15,270	3,332	6,327	444	10,021	16,790	28,876	22,155	6,620	7,874	5,859	5,343	4,124	1,397	1,202	5,343
FB 150%+H	17,683	2,392	6,170	1,271	9,066	17,718	28,279	17,012	7,570	7,240	4,878	4,547	5,029	1,284	1,025	4,547
FB 175%+H	17,476	3,350	7,325	2,180	9,351	18,914	31,746	22,411	7,507	7,289	5,474	5,124	4,749	1,188	1,184	5,125
FB 217%+H	16,902	3,838	7,556	1,903	9,932	18,229	28,265	16,581	8,002	7,317	6,643	4,844	4,760	1,539	1,616	4,844
FB 300%+H	17,105	2,190	6,483	2,263	7,944	15,667	26,630	15,841	8,248	6,218	5,116	4,541	4,839	1,057	1,258	4,545
FB 550%+H	13,670	3,187	7,274	3,083	11,569	21,941	33,753	22,496	8,691	8,319	6,540	5,242	4,165	1,441	1,656	5,243
Family 3																
WikiTopics Art+H	4,638	4,656	4,879	1,369	5,487	3,647	1,614	8,267	2,529	3,095	1,085	4,632	906	3,311	3,754	4,596
WikiTopics Health+H	2,039	3,747	1,112	260	4,879	5,198	3,127	5,817	6,518	3,080	1,570	2,039	714	4,457	2,258	2,025
WikiTopics Infrastructure+H	3,756	4,204	458	58	2,401	2,085	916	2,289	3,791	1,366	1,077	2,641	296	4,259	4,141	2,062
WikiTopics Location+H	11,832	4,248	94	15	216	89	15	325	401	106	453	6,322	62	340	53	5,615
WikiTopics People+H	10,399	6,750	11,616	231	13,960	9,352	4,390	13,591	15,196	7,356	5,883	8,127	1,641	3,015	2,823	7,995
WikiTopics Science+H	1,759	1,609	720	299	2,421	2,394	1,867	2,356	1,341	1,604	764	1,757	529	2,777	2,778	1,721
WikiTopics Taxonomy+H	2,783	1,113	115	26	954	1,132	825	512	3,479	630	1,329	1,077	263	283	1,287	944