HARNESSING STRUCTURAL CONTEXT FOR ENTITY ALIGNMENT FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Entity alignment (EA) across unseen knowledge graph (KG) pairs requires a model to reuse alignment knowledge without learning pair-specific entity embeddings. In a structure-only setting, this raises two connected challenges: evidence from the two KGs must interact through limited seed anchors, and structurally similar candidates must be distinguished beyond coarse similarity. We propose ContextEA, a transferable EA framework that learns cross-KG context construction and candidate discrimination jointly. For each query, its encoder propagates query-relative, relation-conditioned structural signals over an anchor-bridged unified graph. Its decoder then uses entity, neighborhood, relation, and anchor evidence to calibrate ambiguous candidates; the calibration loss also updates the encoder during training. This design enables a pretrained model to apply learned structural alignment patterns to new KG pairs. Across 29 benchmark settings from OpenEA, SRPRS, and DBP, ContextEA improves over structure-only transferable EA baselines. Without target-specific finetuning, it also exceeds the finetuned EAFM baseline across all three benchmark groups.

1 INTRODUCTION

Knowledge graphs (KGs) built from different sources often describe overlapping entities with different identifiers, schemas, and local structures (Ji et al., 2022). Entity alignment (EA) aims to identify entities across such KGs that refer to the same real-world object (Chen et al., 2017; Sun et al., 2018; Chen et al., 2024; Yin et al., 2024; Chen et al., 2025a), and therefore serves as a basic component of knowledge fusion, cross-KG retrieval, KG completion, and downstream KG applications (Zeng et al., 2021; Chen et al., 2025b). Classical EA methods are usually trained for a fixed KG pair with some seed alignments as anchors (Zhang et al., 2022). In contrast, practical applications increasingly require *transferable* EA: a model should learn alignment knowledge from source KG pairs and then generalize to unseen target KGs with different scales, densities, and heterogeneity patterns.

This demand has recently motivated the development of EA foundation models. EAFM (Cui et al., 2026) establishes this pretrain-and-transfer setting with anchor-conditioned parallel encoding and shared relation context. Unlike fixed-pair methods, these query-relative structural signals can be computed on unseen KGs without learning new entity-ID embeddings. Our question is how to make such context pass directly between the two entity graphs and how to use alignment errors to improve the context being encoded.

The first limitation lies in *delayed and weak cross-KG interaction during encoding*. EAFM shares anchor-conditioned initialization and relation features across its two entity streams, but entity-level messages remain within each KG until matching. Thus, a shared relation space need not translate into direct cross-KG entity-context exchange during encoding. For transfer to unseen KG pairs, this exchange must operate on query-relative structure rather than on memorized KG-specific entity embeddings. Figure 1 (a) illustrates the issue, showing a clear drop in performance as query entities move further from training anchors.

The second limitation concerns *structural disambiguation during decoding*. Even when the gold entity has already been retrieved into a small top-k candidate set, similarity-based ranking may still remain structurally ambiguous. In practice, several candidates may receive very similar coarse scores, while only one is truly consistent with the query under local neighborhood patterns, relation semantics, and anchor-supported correspondences. Figure 1 (b) shows this issue: the coarse gap

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.

between the gold target and the hardest negative remains very small, and sometimes even negative. This suggests that a transferable EA model should not only construct better structural context during encoding, but also exploit it more explicitly during decoding.

Motivated by these observations, we propose ContextEA to couple cross-KG context construction with hard-candidate calibration in transferable EA. Prior bridge-based interaction mainly passes embedding states tied to a particular KG pair, whereas transferable EA requires the bridge to carry query-conditioned structural context. Anchor bridges provide the channel, but query-conditioned initialization determines which local relations are active. The anchor-aware relation graph links structural roles across KGs even when relation names differ. Relation-conditioned propagation can therefore transfer query-relevant neighborhood and relation patterns, rather than merely diffusing unconditioned entity features. Because these signals depend on the current query, relations, and available anchors rather than fixed entity IDs, the same learned parameters can process a new KG pair. The decoder then calibrates top candidates using entity, neighborhood, relation, and anchor evidence from this encoder. Its calibration score enters the training objective and back-propagates to the encoder, forming a closed loop in which better cross-KG context supports calibration and calibration supervision shapes the learned context. Thus, the decoder is not a drop-in post-hoc reranker. It depends on reliable encoder context and is optimized with the encoder.

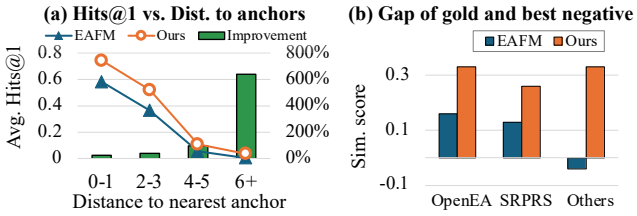


Figure 1: (a) EA performance drops substantially as the query entity becomes farther from the training anchors, indicating that transferable EA is highly sensitive to the availability of cross-KG structural context. (b) In EAFM, the score gap between the gold target and the hardest negative remains very small or even negative. Our structural calibration can enlarge the final score gap and improves discrimination.

The contributions of this work are summarized as follows:

- We revisit transferable EA from the perspective of structural context and identify two coupled bottlenecks in existing work: weak cross-KG interaction during entity encoding and insufficient structural disambiguation during alignment decoding.
- We propose a jointly trained encoder-decoder framework in which query-relative, relation-conditioned context crosses anchor bridges during entity encoding, and hard-candidate calibration supplies supervision back to the encoder.
- We achieve strong results on 29 datasets from OpenEA (Sun et al., 2020b), SRPRS (Guo et al., 2019), and DBP (Sun et al., 2017; 2018). The pretrained ContextEA already outperforms the finetuned EAFM baseline. We further provide analyses on graph characteristics, efficiency, ablations, and parameter sensitivity to show why the method works.

2 Related Work

Knowledge graph foundation models. Recent KG reasoning research has shifted from transductive embedding models toward transferable graph models based on relative structural patterns rather than dataset-specific entity embeddings. Earlier inductive methods reduce reliance on fixed entity identities by reasoning over relational paths and local subgraphs (Zhu et al., 2021; Zhang & Yao, 2022; Huang et al., 2023). Related studies also explore transferable representation learning through multi-source pretraining and local adaptation (Sun et al., 2023b). Building on this trend, recent KG foundation models target broader cross-graph generalization. ULTRA learns transferable reasoning through a relation-centric design and supports zero-shot transfer to unseen KGs (Galkin et al., 2024). KG-ICL further studies prompt-based in-context reasoning on KGs (Cui et al., 2024). Recent studies also enhance cross-graph transferability through more expressive transferable representations (Huang et al., 2025; Zhang et al., 2025). EAFM is the closest prior to our setting, as it adapts the KGFM paradigm to EA through anchor-conditioned parallel encoding and a learnable interaction matcher (Cui et al., 2026). Our work follows the transferable spirit of EAFM, but focuses on how to better inject and exploit structural context throughout both encoding and decoding.

Entity alignment. EA has evolved from translation-based embedding methods to structure-aware graph matching models. Early studies such as MTransE learn cross-KG correspondences in a shared embedding space (Chen et al., 2017). Later methods strengthen structural modeling with graph neural networks and relation-aware interaction (Wang et al., 2018; Sun et al., 2020a; Mao et al., 2020; 2021; Liu et al., 2022; Surisetty et al., 2022). IMEA further improve local context modeling or transferability (Xin et al., 2022). Recent studies also revisit embedding-based EA from the perspective of robustness and adaptivity (Zeng et al., 2022; Gao et al., 2022; Sun et al., 2023a), and analyze entity similarity under multi-source KG settings (Sun et al., 2023c). In recent years, some work exploits large language models for EA (Chen et al., 2024; 2026; Cheng et al., 2025). Despite their strong performance, most existing EA methods are still optimized for a specific KG pair, and their final decisions remain largely dominated by coarse similarity. EAFM takes an important step toward transferable EA by enabling zero-shot generalization across unseen KGs (Cui et al., 2026). This line of work motivates our focus on how to inject and exploit structural context more effectively.

3 PRELIMINARIES

Entity alignment. Let $\mathcal{G}_1 = (\mathcal{E}_1, \mathcal{R}_1, \mathcal{T}_1)$ and $\mathcal{G}_2 = (\mathcal{E}_2, \mathcal{R}_2, \mathcal{T}_2)$ denote two KGs, where $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ denote the sets of entities, relations and facts, respectively. Let $\mathcal{S} = \{(e_i^{(1)} \in \mathcal{E}_1, e_i^{(2)} \in \mathcal{E}_2)\}$ be a set of one-to-one seed alignments between $\mathcal{G}_1$ and $\mathcal{G}_2$. EA aims to identify, for each query entity $q \in \mathcal{E}_1$, its counterpart in $\mathcal{E}_2$. In practice, the EA model learns a scoring function $s(q, c)$ over candidates $c \in \mathcal{E}_2$, and the final prediction is obtained by ranking candidates according to this score.

Transferable entity alignment. Transferable EA considers a set of source tasks $\mathcal{D}_{\text{src}} = \{(\mathcal{G}_1^{\text{s}}, \mathcal{G}_2^{\text{s}}, \mathcal{S}^{\text{s}})\}$ and requires the model to generalize to unseen target KG pairs. The target pair still provides a small seed set to facilitate direct transfer. The key requirement is that the model should rely on transferable structural cues rather than text-based entity identities, so that it remains effective when the entity vocabulary, relation vocabulary, graph scale, and graph heterogeneity all change.

Entity alignment foundation model. EAFM formulates transferable EA as a query-conditioned, relation-guided cross-KG reasoning problem. Its key idea is to avoid memorizing KG-specific entity embeddings and instead represent an unseen KG through relative structural patterns. Given a query entity q and two KGs $\mathcal{G}_1, \mathcal{G}_2$, EAFM first encodes relations in a query-conditioned manner. Specifically, the relation encoder is initialized with a query-aware signal: relations connected to the query entity q are initialized as all-one vectors, while all other relations are initialized as all-zero vectors, thereby explicitly injecting query-specific conditions into the subsequent relation propagation process. It builds a merged relation graph over $\mathcal{R}_1 \cup \mathcal{R}_2$ and computes relation representations

$$\mathbf{R}(q) = f_{\text{rel}}(\mathcal{G}_r, q), \tag{1}$$

where $\mathcal{G}_r$ is the relation graph that captures relation-to-relation interactions induced by aligned anchors, $\mathbf{R}(q) \in \mathbb{R}^{|\mathcal{R}_1 \cup \mathcal{R}_2| \times d}$ denotes relation representation matrix conditioning on entity q. In this way, the learned relation representations capture patterns relevant to the current alignment task and remain applicable to unseen entities and local structures in the target KG.

EAFM also initializes entity states from anchors near the query and activates their aligned counterparts. Its shared entity encoder propagates within the two KGs in parallel; the anchor pairs synchronize query-relative initialization, but are not cross-KG entity-message edges in that encoder. A lightweight matcher then scores candidate pairs. We retain this ID-independent starting point while changing where entity messages can travel and how the final matching signal trains the encoder.

4 METHOD

4.1 FRAMEWORK OVERVIEW

Transferable EA requires cross-KG messages to carry structural evidence that is meaningful for the current query, rather than embeddings tied to a particular target KG pair. We therefore construct query-relative relation and entity states before propagating them across known anchors, and use the resulting context to distinguish ambiguous candidates. The decoder is trained with the encoder so that

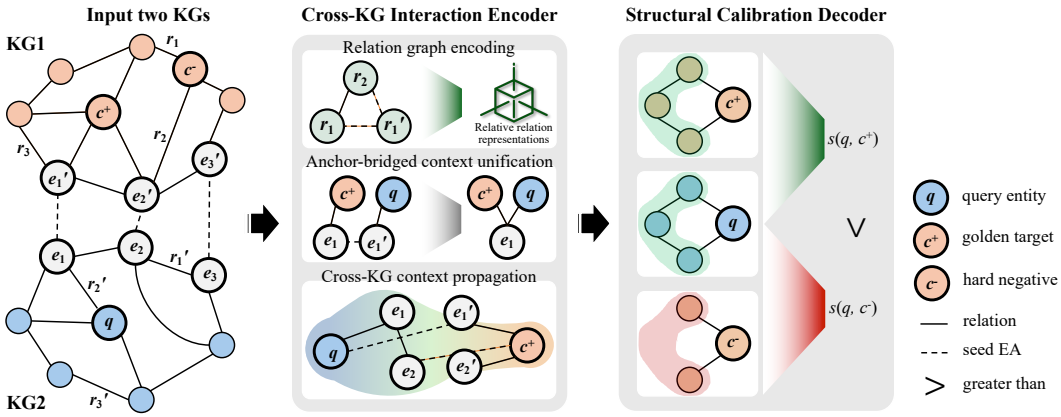


Figure 2: Framework overview of ContextEA: query-relative relation encoding, anchor-bridged entity propagation, full-set coarse scoring, top-k structural calibration, and final ranking over the full target set. Decoder supervision also updates the encoder during training.

calibration also supervises context construction. Given two KGs, their available seed anchors, and a query q, the pipeline in Figure 2 proceeds as follows. (1) Build an anchor-aware relation graph and initialize relation and entity states relative to q. (2) Connect aligned seed entities in a unified entity graph and propagate relation-conditioned context within and across the KGs. (3) Score q against the eligible target entities with the coarse matcher and select its top-k candidates. (4) Compute structural corrections only for these candidates, then rank over the original full eligible target set; candidates outside the shortlist retain their coarse scores. During training, the gold target is combined with coarse hard negatives for decoder supervision, whose gradients also update the encoder.

4.2 Cross-KG Interaction Encoder

The encoder combines query-relative relation context with an entity-level path across the two KGs. EAFM already shares relation features and anchor-conditioned initialization across parallel entity encoders, but its entity messages remain within each KG until matching. We retain the query-relative signals and add explicit cross-KG propagation through known anchors, without learning target-specific entity-ID embeddings.

Shared relation context. The nodes of the merged relation graph are $\mathcal{R}_1 \cup \mathcal{R}_2$; seed-aligned entities are identified in a virtual shared entity space when constructing its edges. Relations are connected by head-head, head-tail, tail-head, tail-tail, and inverse interactions, so cross-KG relation dependencies can arise through shared anchors rather than matching relation names. For a query q, incident KG relations start from all-ones vectors and the remaining KG relations from zero vectors. We reserve two additional relation types: r_{br} for bidirectional anchor bridges and r_{self} for self-loops. Their default initial vectors are shared learnable parameters, replaced by the all-ones vector when the corresponding type is incident to q. The relation encoder thus produces $\mathbf{R}(q) \in \mathbb{R}^{(|\mathcal{R}_1 \cup \mathcal{R}_2|+2)\times d}$; the two special types have no edges in the merged relation graph and are updated through the relation layers' self-state transformations. Query activation determines which local relations contribute to an entity message, while the relation graph connects structural roles across KGs through known anchors. These states are formed before entity propagation reaches a bridge: the bridge supplies a path, whereas the query-relative states determine the structural evidence carried along it.

Anchor-bridged graph unification. We then unify the two KGs using seed anchors from the training split. Specifically, each support anchor pair $(e_i^{(1)}, e_i^{(2)}) \in \mathcal{S}_{\text{sup}}$ is instantiated as an explicit cross-KG bridge edge. Original KG edges preserve intra-KG structure, while anchor bridges allow entity messages to cross between KGs during encoding; this entity-level propagation differs from merely merging anchors to construct the relation graph. Following the EAFM protocol, these bridges use only known support alignments. During training, when a supervised pair is used as the current prediction target, its own entity-level bridge edge is masked from the propagation

graph for that batch; the gold counterpart is therefore used as a label, not as a direct cross-KG neighbor. The remaining support anchors still provide the cross-KG propagation channel and aggregate relation/anchor statistics. Held-out test alignments are not used to construct bridge edges, and final evaluation is still performed over the full eligible target entity set; thus this protocol uses only the support information allowed by the task setting and does not introduce test-label leakage.

Cross-KG interaction propagation. On the unified graph, query-relative entity states propagate along original KG edges and anchor bridges. For an edge (u, r, v), we first compute an independent gate from the source entity and encoded relation states, then generate and aggregate messages:

$$\begin{aligned} \alpha_{urv}^{(l)} &= \mathrm{sigmoid}\Big(\mathbf{w}_\alpha^\top \,\mathrm{LeakyReLU}([\mathbf{W}_r\mathbf{r}_r(q); \mathbf{W}_s\mathbf{h}_u^{(l)}]) + b_\alpha\Big), \\ \mathbf{m}_{urv}^{(l)} &= \alpha_{urv}^{(l)}\big(\mathbf{h}_u^{(l)} + \mathbf{r}_r(q)\big), \\ \mathbf{h}_v^{(l+1)} &= \mathbf{h}_v^{(l)} + \mathrm{Dropout}\left(\mathrm{LN}\left(\mathbf{W}_h \sum_{(u,r,v)} \mathbf{m}_{urv}^{(l)}\right)\right). \end{aligned} \tag{2}$$

The sum runs over incoming edges to v; both directions of an anchor bridge use the encoded state $\mathbf{r}_{r_{\mathrm{br}}}(q)$, while self-loops use $\mathbf{r}_{r_{\mathrm{self}}}(q)$. Edges have independent sigmoid gates, without degree normalization or neighbor-wise softmax. An empty incoming-edge set has a zero aggregate. Although the bridge type starts from a shared parameter, $\mathbf{h}_u^{(l)}$ is initialized and updated relative to q. Thus, bridge messages carry query-dependent structural states across KGs, which the downstream decoder can compare with the candidate's local context.

4.3 Structural Calibration Decoder

The encoder produces structural states for all entities, but coarse matching can assign similar scores to the gold target and hard negatives. For each query q, we first score every eligible target entity c with $s_{\mathrm{emb}}(q, c) = [|\mathbf{h}_q - \mathbf{h}_c|; \mathbf{h}_c]\mathbf{W}_{\mathrm{emb}}$, where $\mathbf{h}_q, \mathbf{h}_c \in \mathbb{R}^d$ and $\mathbf{W}_{\mathrm{emb}} \in \mathbb{R}^{2d\times 1}$. After the evaluation protocol's known-anchor filtering, $\mathcal{C}_k(q) = \mathrm{TopK}_c\ s_{\mathrm{emb}}(q, c)$ selects only the candidates requiring structural calibration. This is not an external retrieval stage and uses no test-time gold labels; k controls calibration cost, not the final ranking universe.

Complementary structural evidence. The decoder reads four views from the same encoder context. Direct entity compatibility ϕ_{ent} compares the pair's states, including element-wise interactions beyond the coarse linear score. Neighborhood evidence ϕ_{nei} compares one-hop entity context, and relation evidence ϕ_{rel} compares adjacent relation patterns. Anchor-supported evidence ϕ_{anc} checks whether known alignments link the two local neighborhoods. These views distinguish candidates that have similar coarse scores but differ in their structural support; their exact feature definitions and empty-neighborhood handling are in Appendix K.

Anchor-supported local structure. Let $\mathcal{N}_e(x)$ contain up to 24 one-hop $(\text{relation}, \text{neighbor})$ entries, including inverse KG relations but excluding bridges. For a candidate pair (q, c), the anchor-supported entry matches are

$$P(q, c) = \{(i, j) \mid (r_i, u_i) \in \mathcal{N}_e(q),\ (s_j, v_j) \in \mathcal{N}_e(c),\ (u_i, v_j) \in \mathcal{S}_{\mathrm{sup}}\}. \tag{3}$$

The set $P(q, c)$ identifies which local neighbor pairs are supported by known alignments, rather than by coarse similarity alone. We concatenate the four views as $\phi(q, c) = [\phi_{\mathrm{ent}}(q, c); \phi_{\mathrm{nei}}(q, c); \phi_{\mathrm{rel}}(q, c); \phi_{\mathrm{anc}}(q, c)]$ and use a lightweight MLP to produce a bounded correction $\Delta(q, c) = \tanh(f_{\mathrm{dec}}(\phi(q, c)))$. Appendix K specifies the pooled vectors, relation-consistency statistics, and zero values when no support is available.

Final score calibration. Only selected candidates receive a structural correction; the others retain their coarse scores:

$$s_{\mathrm{final}}(q, c) = s_{\mathrm{emb}}(q, c) + \beta\, \mathbb{I}[c \in \mathcal{C}_{\mathrm{dec}}(q)]\, \Delta(q, c), \tag{4}$$

where $\beta \in \mathbb{R}$ controls the correction strength. At inference $\mathcal{C}_{\mathrm{dec}}(q) = \mathcal{C}_k(q)$, and the final rank is computed over the full eligible target set; candidates outside the top-k retain their coarse scores.

4.4 TRAINING

To optimize transferable EA, training samples either KG direction and combines a full-candidate coarse loss with a decoder loss. For a query q with gold target c^+, let $\mathcal{E}_{\text{opp}}$ be the eligible opposite-KG entity set and $\mathcal{H}_m(q)$ contain up to $m = 5$ highest-scoring incorrect coarse candidates. During training, $\mathcal{C}_{\text{dec}}(q) = \{c^+\} \cup \mathcal{H}_m(q)$: the gold target is included explicitly for decoder supervision, even if it is absent from the coarse top-k. The losses are

$$\begin{aligned} \mathcal{L}_{\text{coarse}}(q) &= -\log \frac{e^{s_{\text{emb}}(q,c^+)}}{\sum_{c\in\mathcal{E}_{\text{opp}}} e^{s_{\text{emb}}(q,c)}}, \\ \mathcal{L}_{\text{dec}}(q) &= -\log \frac{e^{s_{\text{final}}(q,c^+)}}{\sum_{c\in\{c^+\}\cup\mathcal{H}_m(q)} e^{s_{\text{final}}(q,c)}}, \\ \mathcal{L} &= \mathbb{E}_q \left[\mathcal{L}_{\text{coarse}}(q) + \lambda_{\text{dec}}\mathcal{L}_{\text{dec}}(q)\right]. \end{aligned} \tag{5}$$

The coarse loss compares the gold target with the full eligible set, while the decoder loss focuses on hard negatives. During pretraining, the decoder term starts in epoch 20 with $\lambda_{\text{dec}} = 0.2$ (and is zero before then); finetuning enables it from the start. The support-query bridge masking described above applies to each training batch. Because s_{final} depends on encoder states, decoder gradients update both the relation and entity encoders through the selected pairs' scores, not through the discrete top-k selection. Thus, calibration is part of joint training rather than inference-only reranking.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETTINGS

Datasets. Following EAFM (Cui et al., 2026), we evaluate on 29 EA datasets from three benchmark groups: OpenEA (Sun et al., 2020b), SRPRS (Guo et al., 2019), and DBpedia-based benchmarks (DBP) (Chen et al., 2017; Sun et al., 2018). Appendix A summarizes the statistics of these datasets. They cover cross-KG and cross-lingual alignment, 15K to 100K graphs, and sparse (V1) and dense (V2) regimes. The two pretraining source pairs named below also appear in the 29-dataset table; their held-out results measure performance within the source domains. The other 27 pairs are not used for pretraining and measure transfer to target KGs. For direct transfer, target training anchors define structural bridges while model parameters remain frozen; "finetune" additionally updates the model on the target training split. Each dataset uses a 20%/10%/70% train/validation/test split.

Implementations. Following EAFM, we use D-W-15K-V1 and EN-DE-15K-V1 as source datasets so that pretraining covers both cross-KG and cross-lingual structural patterns. We adopt the same training configuration as EAFM for fair comparison, including a 6-layer relation encoder, a 6-layer entity encoder, hidden size 32, balance coefficient $\beta = 0.3$, AdamW optimizer with learning rate 5×10^{-4}, batch size 64, and early stopping based on validation MRR. We set the number of candidates $k = 16$ in our decoder for a efficiency-effectiveness trade-off. The code is in supplementary material.

Metrics and implementations. We report Mean Reciprocal Rank (MRR), Hits@10 and Hits@1 results to assess EA performance. We treat Hits@1 as the most direct measure of exact EA results. The main baselines are ULTRA-EA (a EA variant of ULTRA (Galkin et al., 2024)) and EAFM (Cui et al., 2026), both evaluated under the same "pretrain" and "finetune" settings.

5.2 MAIN RESULTS

Table 1 summarizes the average results of each EA groups. Per-dataset results are reported in Appendix B. Table 1 reports group-level averages over repeated runs, while Table 6 reports per-dataset results from a representative run under the same protocol; the recomputed group averages differ by less than 0.003 due to normal run-to-run variation. Three general patterns can be observed.

(i) ContextEA consistently outperforms the baselines across all dataset groups under both pretraining and finetuning settings. In the pretraining scenario, it shows clear improvements over EAFM. Notably, the pretrained ContextEA already exceeds the performance of the finetuned EAFM baseline on

Table 1: Comparison of average EA results across different dataset groups, where the **bold** and underline indicate the best and second-best results, respectively.

Method	OpenEA			SRPRS			DBP			Average		
	MRR	H@10	H@1	MRR	H@10	H@1	MRR	H@10	H@1	MRR	H@10	H@1
ULTRA-EA pretrain	0.477	0.635	0.385	0.350	0.557	0.241	0.203	0.379	0.107	0.395	0.569	0.297
ULTRA-EA finetune	0.509	0.691	0.416	0.380	0.587	0.271	0.223	0.409	0.137	0.424	0.614	0.328
EAFM pretrain	0.602	0.764	0.517	0.530	0.735	0.426	0.294	0.452	0.214	0.529	0.702	0.440
EAFM finetune	0.656	0.814	0.573	0.605	0.792	0.508	0.464	0.662	0.394	0.609	0.782	0.524
ContextEA pretrain	0.713	0.857	0.636	0.675	0.837	0.587	0.522	0.667	0.444	0.669	0.819	0.589
ContextEA finetune	**0.722**	**0.859**	**0.648**	**0.680**	**0.839**	**0.594**	**0.588**	**0.743**	**0.503**	**0.688**	**0.834**	**0.608**

all benchmark groups. This suggests that the proposed method possesses strong transferability even before adaptation to target KGs. **(ii)** Finetuning further enhances performance. The gains are relatively moderate on OpenEA and SRPRS, whereas a more pronounced improvement is observed on DBP. This indicates that target-specific adaptation plays a greater role in more challenging settings. **(iii)** A clear performance gap remains between ContextEA and ULTRA-EA under both settings. This result suggests that EA-oriented pretraining alone is insufficient when the model architecture is not explicitly tailored to entity alignment, highlighting the need to model cross-KG correspondences.

5.3 Detailed Analysis on Graph Characteristics

Table 2: MRR results across different graph characteristics. Following the EAFM grouping protocol, the sparse group includes OpenEA/SRPRS V1 splits and DBP benchmarks, while the dense group includes OpenEA/SRPRS V2 splits. The small-scale group includes OpenEA 15K, SRPRS, and DBP15K datasets; the 100K group includes OpenEA 100K and DBP 100K datasets.

Method	Graph Density		Graph Scale		Heterogeneity	
	Sparse/V1	Dense/V2	15K/Small	100K	Cross-KG	Cross-Lingual
ULTRA-EA pretrain	0.315	0.509	0.393	0.398	0.496	0.301
ULTRA-EA finetune	0.342	0.540	0.420	0.432	0.530	0.325
EAFM pretrain	0.451	0.639	0.532	0.523	0.597	0.465
EAFM finetune	0.525	0.727	0.602	0.622	0.677	0.545
ContextEA pretrain	0.575	0.806	0.674	0.666	0.731	0.615
ContextEA finetune	0.596	0.813	0.681	0.695	0.746	0.630

Table 2 analyzes performance under different graph characteristics. ContextEA consistently performs best across all categories. **(i)** On sparse KGs, the pretrained model already outperforms the finetuned EAFM baseline, showing that stronger cross-KG interaction helps when structural cues are weak. On dense KGs, the advantage becomes even larger, suggesting that the encoder-decoder design makes better use of richer structure contexts. **(ii)** A similar trend is observed across graph scales. ContextEA performs well on both small and large datasets, with larger gains on bigger graphs. This indicates that cross-KG propagation and structural calibration become more important as the candidate space grows. **(iii)** Our method also remains stable under different heterogeneity levels. These results show that ContextEA generalizes well across variations in density, scale, and heterogeneity.

5.4 Ablation Study

Effectiveness of our encoder and decoder. Table 3 shows that replacing our encoder causes a much larger performance drop than removing the decoder. The encoder propagates structural context across KGs and produces the coarse ranking over the full target set; the decoder uses this context to calibrate ambiguous top-k candidates, so its direct benefit is concentrated on hard cases. Decoder scores also enter the training loss and back-propagate to the encoder. The full-model gain therefore involves both candidate calibration and joint training, although this ablation cannot isolate the contribution of decoder gradients.

Table 3: Ablation results across dataset groups. In the variant "w/o Our encoder", we use the EAFM encoder instead. In the variant "w/o Our decoder", we use coarse similarities for decoding.

Method	OpenEA			SRPRS			DBP			Average		
	MRR	H@10	H@1	MRR	H@10	H@1	MRR	H@10	H@1	MRR	H@10	H@1
ContextEA	0.713	0.857	0.636	0.675	0.837	0.587	0.522	0.667	0.444	0.669	0.819	0.589
w/o Our decoder	0.696	0.843	0.618	0.666	0.834	0.577	0.493	0.643	0.412	0.653	0.806	0.571
w/o Our encoder	0.565	0.731	0.479	0.535	0.741	0.429	0.240	0.361	0.177	0.501	0.670	0.413

With the EAFM encoder and our decoder, average MRR falls to 0.501, below the EAFM pretrain reference of 0.529 in Table 1. This is because the decoder depends on reliable cross-KG structural context from the encoder. Without explicit entity-level cross-KG propagation, the neighborhood, relation, and anchor evidence available for hard candidates can be less reliable. Calibration based on such evidence may promote incorrect candidates and disrupt the coarse ranking; its loss may also send misleading training signals to the encoder. The decoder thus depends on the quality of the encoder context rather than acting as a separate post-hoc reranker.

Table 4: Decoder-view ablation across dataset groups over repeated runs.

Variant	OpenEA MRR	SRPRS MRR	DBP MRR	Avg. MRR
Full ContextEA	0.713 ± 0.003	0.675 ± 0.002	0.522 ± 0.004	0.669 ± 0.003
w/o entity view	0.708 ± 0.003	0.672 ± 0.002	0.515 ± 0.004	0.665 ± 0.003
w/o neighborhood view	0.704 ± 0.004	0.670 ± 0.003	0.509 ± 0.005	0.661 ± 0.004
w/o relation view	0.706 ± 0.003	0.671 ± 0.002	0.511 ± 0.004	0.663 ± 0.003
w/o anchor-aware view	0.702 ± 0.004	0.668 ± 0.003	0.505 ± 0.005	0.659 ± 0.004
w/o entire decoder	0.696 ± 0.004	0.666 ± 0.003	0.493 ± 0.006	0.653 ± 0.004

Effectiveness of different structure contexts in our decoder. Table 4 reports repeated-run results for the four decoder views. Removing any single view reduces average MRR, while removing the entire decoder leads to a larger decrease in every dataset group. The effect is most pronounced on DBP, where the complete decoder provides the largest gain over coarse matching.

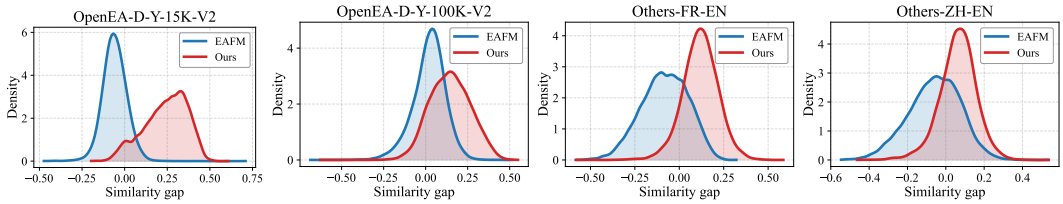


Figure 3: Density plots of the similarity gap between the gold target and the hardest negative candidate on four datasets. Results on other datasets are in Appendix C.

5.5 Other Analyses

Gold vs. hard-negative score gaps. Figure 3 shows how ContextEA changes pairwise discrimination. For each query entity, we compute the similarity gap between the gold target and the hardest negative candidate. Larger positive values indicate clearer separation, while negative values mean that the hardest negative is ranked above the true target. We can see that the density curve of ContextEA is shifted to the right, meaning that more queries obtain a larger positive margin. These results complement the ranking metrics in Tables 1–3. The improvement is not only in average MRR or Hits@1, but also in the score distribution: the model creates a clearer gap between the gold entity and the hardest negative. This matches the purpose of the structural calibration decoder, which enlarges the score gap when coarse similarity is not enough.

Efficiency analysis. Figure 4 compares the efficiency–effectiveness trade-off of ContextEA. Detailed results are presented in Appendix D. The experiments are conducted on an NVIDIA RTX 4090 GPU. Efficiency is measured as the average inference time per query. We vary the size k to examine its impact. The fastest variant is "Ours (w/o Decoder)", which also achieves a large gain over EAFM in Hits@1 while reducing inference time. This shows that our encoder is not only more accurate, but also more efficient than the original EAFM inference pipeline. Adding the decoder increases inference time, but it further improves accuracy over the decoder-free variant. The best trade-off is achieved at small to medium candidate sizes, while larger k mainly adds cost with limited benefit. Overall, the results support our claim that structural calibration is lightweight and effective when applied to a small candidate set.

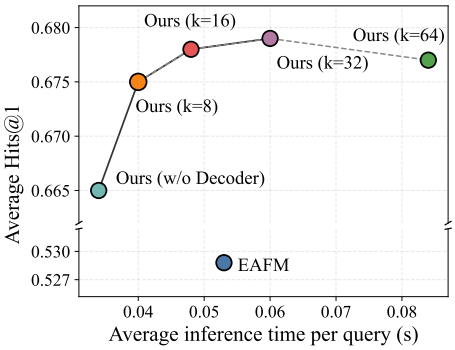


Figure 4: Efficiency-effectiveness trade-off.

Sensitivity analysis. Figure 5 studies the sensitivity of the structural calibration weight β. Detailed results are in Appendix E. The overall trend is stable, which indicates that the decoder does not rely on brittle hyperparameter tuning. A moderate value around $\beta = 0.3$ gives the best average performance, suggesting that structural calibration should act as a correction to the coarse score rather than dominate it. When β is too small, structural evidence is underused; when it becomes too large, some datasets show mild over-correction. This is consistent with our goal of keeping the decoder lightweight but effective.

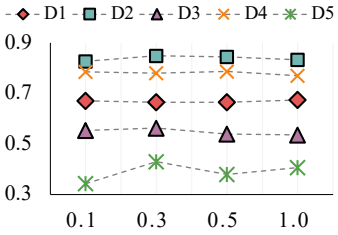


Figure 5: MRR w.r.t. weight β.

Case study. Figure 6 shows a typical successful alignment made by ContextEA. The query entity is correctly aligned to the green target entity, while the red entity is a hard negative. The hard negative is heavily influenced by the anchor "Gary Sinise", which provides three relations around it. The anchor also has a direct link to the query. This gives the wrong candidate strong local support. EAFM cannot clearly separate the two candidates using coarse similarity. In contrast, our encoder passes local semantic and relational signals to build a clearer context. The decoder then benefits from this improved structural context. Thus, ContextEA can produce a clear score gap.

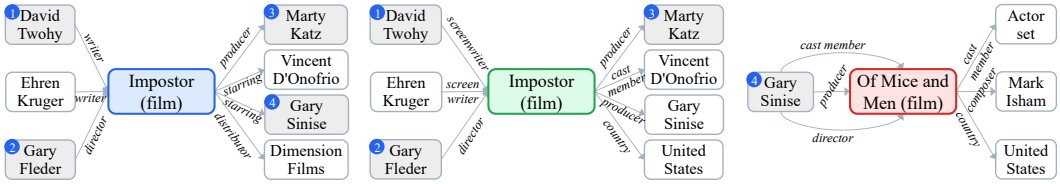


Figure 6: A case study of EA. The left subgraph shows the query entity. The middle green node is the correct matched entity in the target KG, and the right red node is a hard negative candidate. Blue numbered nodes indicate anchor entities.

Robustness and failure cases. We further examine the stability across random seeds, sensitivity to anchor perturbations, and structural patterns of incorrect predictions (Appendices F, G, and J).

6 Conclusion and Future Work

We study transferable EA from the perspective of structural context and identify two bottlenecks: weak cross-KG interaction during encoding and limited structural discrimination during final ranking. ContextEA addresses them with a lightweight cross-KG interaction encoder and structural calibration decoder, yielding consistent gains on OpenEA, SRPRS, and DBP. Future work includes stronger propagation modules for larger heterogeneous KGs and richer anchor-aware calibration.

AI USE STATEMENT

Generative AI tools assisted with language editing, code review, and experimental table formatting. The authors take responsibility for the final manuscript, including all reported data and claims.

REFERENCES

Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *IJCAI*, pp. 1511–1517, Melbourne, Australia, 2017. ijcai.org.

Shengyuan Chen, Qinggang Zhang, Junnan Dong, Wen Hua, Qing Li, and Xiao Huang. Entity alignment with noisy annotations from large language models. In *NeurIPS*, 2024.

Shengyuan Chen, Zheng Yuan, Qinggang Zhang, Wen Hua, Jiannong Cao, and Xiao Huang. Neusymea: Neuro-symbolic entity alignment via variational inference. In *NeurIPS*, 2025a.

Xuan Chen, Tong Lu, and Zhichun Wang. Llm-align: Utilizing large language models for entity alignment in knowledge graphs. *Data Intell.*, 8(1):220–243, 2026.

Zerui Chen, Huiming Fan, Qianyu Wang, Tao He, Ming Liu, Heng Chang, Weijiang Yu, Ze Li, and Bing Qin. How do language models reshape entity alignment? a survey of lm-driven ea methods: Advances, benchmarks, and future. In *EMNLP*, pp. 23230–23245, 2025b.

Jingwei Cheng, Chenglong Lu, Linyan Yang, Guoqing Chen, and Fu Zhang. Easyea: Large language model is all you need in entity alignment between knowledge graphs. In *Findings of ACL*, pp. 20981–20995, 2025.

Yuanning Cui, Zequn Sun, and Wei Hu. A prompt-based knowledge graph foundation model for universal in-context reasoning. In *NeurIPS*, 2024.

Yuanning Cui, Zequn Sun, Wei Hu, Kexuan Xin, and Zhangjie Fu. Breaking the reasoning horizon in entity alignment foundation models. *CoRR*, abs/2601.21174, 2026.

Mikhail Galkin, Xinyu Yuan, Hesham Mostafa, Jian Tang, and Zhaocheng Zhu. Towards foundation models for knowledge graph reasoning. In *ICLR*, Vienna, Austria, 2024.

Yunjun Gao, Xiaoze Liu, Junyang Wu, Tianyi Li, Pengfei Wang, and Lu Chen. Clusterea: Scalable entity alignment with stochastic training and normalized mini-batch similarities. In *KDD*, pp. 421–431, 2022.

Lingbing Guo, Zequn Sun, and Wei Hu. Learning to exploit long-term relational dependencies in knowledge graphs. In *ICML*, pp. 2505–2514, 2019.

Xingyue Huang, Miguel Romero, İsmail İlkan Ceylan, and Pablo Barceló. A theory of link prediction via relational weisfeiler-leman on knowledge graphs. In *NeurIPS*, 2023.

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 *ICML*. OpenReview.net, 2025.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. A survey on knowledge graphs: Representation, acquisition, and applications. *TNNLS*, 33(2):494–514, 2022.

Xuhui Jiang, Yinghan Shen, Zhichao Shi, Chengjin Xu, Wei Li, Zixuan Li, Jian Guo, Huawei Shen, and Yuanzhuo Wang. Unlocking the power of large language models for entity alignment. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 7566–7583, 2024. doi: 10.18653/v1/2024.acl-long.408.

Xiao Liu, Haoyun Hong, Xinghao Wang, Zeyi Chen, Evgeny Kharlamov, Yuxiao Dong, and Jie Tang. Selfkg: Self-supervised entity alignment in knowledge graphs. In *WWW*, pp. 860–870, 2022.

Xin Mao, Wenting Wang, Huimin Xu, Yuanbin Wu, and Man Lan. Relational reflection entity alignment. In *CIKM*, pp. 1095–1104, 2020.

Xin Mao, Wenting Wang, Yuanbin Wu, and Man Lan. Boosting the speed of entity alignment 10×: Dual attention matching network with normalized hard sample mining. In *WWW*, pp. 821–832, Ljubljana, Slovenia, 2021. ACM.

Xin Mao, Wenting Wang, Yuanbin Wu, and Man Lan. Lightea: A scalable, robust, and interpretable entity alignment framework via three-view label propagation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 825–838, 2022. doi: 10.18653/v1/2022.emnlp-main.52.

Zequn Sun, Wei Hu, and Chengkai Li. Cross-lingual entity alignment via joint attribute-preserving embedding. In *ISWC*, pp. 628–644, 2017.

Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. Bootstrapping entity alignment with knowledge graph embedding. In *IJCAI*, pp. 4396–4402, Stockholm, Sweden, 2018. ijcai.org.

Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In *AAAI*, pp. 222–229, New York City, NY, USA, 2020a. AAAI Press.

Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. A benchmarking study of embedding-based entity alignment for knowledge graphs. *PVLDB*, 13(11):2326–2340, 2020b.

Zequn Sun, Wei Hu, Chengming Wang, Yuxin Wang, and Yuzhong Qu. Revisiting embedding-based entity alignment: A robust and adaptive method. *TKDE*, 35(8):8461–8475, 2023a.

Zequn Sun, Jiacheng Huang, Jinghao Lin, Xiaozhou Xu, Qijin Chen, and Wei Hu. Joint pre-training and local re-training: Transferable representation learning on multi-source knowledge graphs. In *KDD*, pp. 2132–2144, 2023b.

Zequn Sun, Jiacheng Huang, Xiaozhou Xu, Qijin Chen, Weijun Ren, and Wei Hu. What makes entities similar? A similarity flooding perspective for multi-sourced knowledge graph embeddings. In *ICML*, pp. 32875–32885, 2023c.

Anil Surisetty, Deepak Chaurasiya, Nitish Kumar, Alok Singh, Gaurav Dhama, Aakarsh Malhotra, Ankur Arora, and Vikrant Dey. Reps: Relation, position and structure aware entity alignment. In *Companion of The Web Conference*, pp. 1083–1091, 2022.

Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. Cross-lingual knowledge graph alignment via graph convolutional networks. In *EMNLP*, pp. 349–357, 2018.

Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, Rui Yan, and Dongyan Zhao. Relation-aware entity alignment for heterogeneous knowledge graphs. In *IJCAI*, pp. 5278–5284, Macao, China, 2019. ijcai.org.

Kexuan Xin, Zequn Sun, Wen Hua, Wei Hu, and Xiaofang Zhou. Informed multi-context entity alignment. In *WSDM*, pp. 1197–1205, 2022.

Hang Yin, Liyao Xiang, Dong Ding, Yuheng He, Yihan Wu, Pengzhi Chu, Xinbing Wang, and Chenghu Zhou. Lambda: Learning matchable prior for entity alignment with unlabeled dangling cases. In *NeurIPS*, 2024.

Kaisheng Zeng, Chengjiang Li, Lei Hou, Juanzi Li, and Ling Feng. A comprehensive survey of entity alignment for knowledge graphs. *AI Open*, 2:1–13, 2021.

Weixin Zeng, Xiang Zhao, Xinyi Li, Jiuyang Tang, and Wei Wang. On entity alignment at scale. *VLDB J.*, 31(5):1009–1033, 2022.

Rui Zhang, Bayu Distiawan Trisedya, Miao Li, Yong Jiang, and Jianzhong Qi. A benchmark and comprehensive survey on knowledge graph entity alignment via representation learning. *The VLDB Journal*, 31(5):1143–1168, 2022.

Yongqi Zhang and Quanming Yao. Knowledge graph reasoning with relational digraph. In *WWW*, pp. 912–924, 2022.

Yucheng Zhang, Beatrice Bevilacqua, Mikhail Galkin, and Bruno Ribeiro. TRIX: A more expressive model for zero-shot domain transfer in knowledge graphs. *CoRR*, abs/2502.19512, 2025.

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 *NeurIPS*, pp. 29476–29490, 2021.

Table 5: Statistics of EA datasets.

Group	Dataset	$\mathcal{G}_1$				$\mathcal{G}_2$			
		KG	# Ent	# Rel	# Facts	KG	# Ent	# Rel	# Facts
OpenEA	EN-DE-15K-V1	DBpedia (EN)	15,000	215	47,676	DBpedia (DE)	15,000	131	50,419
	EN-DE-15K-V2	DBpedia (EN)	15,000	169	84,867	DBpedia (DE)	15,000	96	92,632
	EN-FR-15K-V1	DBpedia (EN)	15,000	267	47,334	DBpedia (FR)	15,000	210	40,864
	EN-FR-15K-V2	DBpedia (EN)	15,000	193	96,318	DBpedia (FR)	15,000	166	80,112
	D-W-15K-V1	DBpedia	15,000	248	38,265	Wikidata	15,000	169	42,746
	D-W-15K-V2	DBpedia	15,000	167	73,983	Wikidata	15,000	121	83,365
	D-Y-15K-V1	DBpedia	15,000	165	30,291	YAGO3	15,000	28	26,638
	D-Y-15K-V2	DBpedia	15,000	72	68,063	YAGO3	15,000	21	60,970
	EN-DE-100K-V1	DBpedia (EN)	100,000	381	335,359	DBpedia (DE)	100,000	196	336,240
	EN-DE-100K-V2	DBpedia (EN)	100,000	323	622,588	DBpedia (DE)	100,000	170	629,395
	EN-FR-100K-V1	DBpedia (EN)	100,000	400	309,607	DBpedia (FR)	100,000	300	258,285
	EN-FR-100K-V2	DBpedia (EN)	100,000	379	649,902	DBpedia (FR)	100,000	287	561,391
	D-W-100K-V1	DBpedia	100,000	413	293,990	Wikidata	100,000	261	251,708
	D-W-100K-V2	DBpedia	100,000	318	616,457	Wikidata	100,000	239	588,203
	D-Y-100K-V1	DBpedia	100,000	287	294,188	YAGO3	100,000	32	400,518
	D-Y-100K-V2	DBpedia	100,000	230	576,547	YAGO3	100,000	31	865,265
SRPRS	D-W-V1	DBpedia (EN)	15,000	253	38,421	Wikidata (EN)	15,000	144	40,159
	D-W-V2	DBpedia (EN)	15,000	220	68,598	Wikidata (EN)	15,000	135	75,465
	D-Y-V1	DBpedia (EN)	15,000	219	33,571	YAGO3 (EN)	15,000	30	34,660
	D-Y-V2	DBpedia (EN)	15,000	206	71,257	YAGO3 (EN)	15,000	30	97,131
	EN-DE-V1	DBpedia (EN)	15,000	225	38,281	DBpedia (DE)	15,000	118	37,069
	EN-DE-V2	DBpedia (EN)	15,000	207	56,983	DBpedia (DE)	15,000	117	59,848
	EN-FR-V1	DBpedia (EN)	15,000	221	36,508	DBpedia (FR)	15,000	177	33,532
	EN-FR-V2	DBpedia (EN)	15,000	217	71,929	DBpedia (FR)	15,000	174	66,760
DBP	ZH-EN	DBpedia (ZH)	19,388	1,701	70,414	DBpedia (EN)	19,572	1,323	95,142
	JA-EN	DBpedia (JA)	19,814	1,299	77,214	DBpedia (EN)	19,780	1,153	93,484
	FR-EN	DBpedia (FR)	19,661	903	105,998	DBpedia (EN)	19,993	1,208	115,722
	D-W-100K	DBpedia	100,000	330	463,294	Wikipedia	100,000	220	448,774
	D-Y-100K	DBpedia	100,000	302	428,952	YAGO3	100,000	31	502,563

A Statistics of EA datasets

We list the statistics of all EA datasets in Table 5. It includes three benchmark collections: OpenEA, SRPRS, and DBP. For each dataset, we show the two paired KGs and report the numbers of entities, relations, and triples. These benchmarks cover different KG pairs, languages, and graph sizes, from 15K to 100K entities, with varying structural densities. This diversity supports a systematic evaluation under different alignment settings.

B Detailed EA Results

We present the per-dataset pretraining and finetuning results in Tables 6 and 7, respectively. The tables report MRR, Hits@10 (H@10), and Hits@1 (H@1) where available for OpenEA, SRPRS, and DBP. The reported metrics are defined in the main text.

B.1 Input Information and Comparison Settings

Table 8 compares input information and target-pair training. Its transferable row refers to evaluation without target-specific finetuning; Table 7 reports finetuned results separately.

ContextEA uses KG structure and the given seed links, without entity names, text descriptions, or knowledge from an LLM. Text-based methods such as RDGCN use names and pretrained word vectors (Wu et al., 2019); LLM-based methods can draw on model knowledge or generated alignment clues (Chen et al., 2024; Jiang et al., 2024; Cheng et al., 2025). These extra signals may help match entities that are hard to identify from structure alone. Their raw scores are useful references, but do not provide a fair direct comparison of structure-only transferable models. We limit our main performance claim to the structure-only transferable EA setting.

Some EA benchmarks provide clear name clues: aligned entities in DWY100K often have nearly identical names (Mao et al., 2022). In real KGs, names or descriptions may be missing, unclear, shortened, or inconsistent across languages. Methods that rely on them may be affected when text is

Table 6: Per-dataset entity alignment results before target-pair finetuning (MRR, H@10, and H@1 where available).

Group	Dataset	ULTRA-EA		EAFM		ContextEA		
		MRR	H@10	MRR	H@10	MRR	H@10	H@1
OpenEA	D-W-15K-V1	0.431	0.613	0.604	0.795	0.674	0.847	0.578
	D-W-15K-V2	0.671	0.852	0.738	0.890	0.842	0.961	0.774
	D-Y-15K-V1	0.560	0.699	0.636	0.770	0.704	0.809	0.636
	D-Y-15K-V2	0.901	0.936	0.950	0.987	0.971	0.994	0.958
	D-W-100K-V1	0.342	0.539	0.459	0.650	0.548	0.733	0.455
	D-W-100K-V2	0.568	0.749	0.437	0.590	0.694	0.853	0.608
	D-Y-100K-V1	0.558	0.731	0.694	0.853	0.740	0.879	0.665
	D-Y-100K-V2	0.772	0.849	0.762	0.888	0.908	0.971	0.871
	EN-DE-15K-V1	0.587	0.780	0.737	0.900	0.786	0.934	0.710
	EN-DE-15K-V2	0.754	0.862	0.765	0.889	0.886	0.969	0.835
	EN-FR-15K-V1	0.254	0.487	0.533	0.782	0.620	0.847	0.504
	EN-FR-15K-V2	0.267	0.477	0.460	0.686	0.723	0.924	0.613
	EN-DE-100K-V1	0.386	0.532	0.520	0.682	0.573	0.744	0.489
	EN-DE-100K-V2	0.248	0.434	0.594	0.736	0.746	0.863	0.683
	EN-FR-100K-V1	0.178	0.330	0.364	0.572	0.439	0.630	0.347
	EN-FR-100K-V2	0.163	0.298	0.378	0.552	0.588	0.760	0.495
SRPRS	D-W-V1	0.258	0.432	0.340	0.559	0.514	0.724	0.410
	D-W-V2	0.489	0.755	0.633	0.827	0.819	0.951	0.742
	D-Y-V1	0.248	0.422	0.421	0.665	0.515	0.731	0.407
	D-Y-V2	0.377	0.591	0.662	0.861	0.876	0.975	0.819
	EN-DE-V1	0.302	0.492	0.514	0.710	0.580	0.759	0.484
	EN-DE-V2	0.409	0.616	0.704	0.876	0.790	0.917	0.715
	EN-FR-V1	0.229	0.401	0.378	0.596	0.480	0.692	0.374
	EN-FR-V2	0.484	0.746	0.589	0.784	0.834	0.961	0.758
DBP	FR-EN 15K	0.075	0.245	0.139	0.261	0.414	0.583	0.325
	JA-EN 15K	0.095	0.294	0.147	0.283	0.402	0.552	0.322
	ZH-EN 15K	0.084	0.245	0.157	0.291	0.369	0.524	0.287
	D-W-100K	0.452	0.621	0.453	0.657	0.647	0.791	0.569
	D-Y-100K	0.310	0.488	0.573	0.771	0.778	0.885	0.717

less reliable. ContextEA does not depend on clean text, making structure-only EA useful in such settings. It can still struggle when neighborhoods are sparse or too few seed links are available.

C Additional Results on Gold vs. Hard-Negative Score Gaps

Figure 7 reports the similarity-gap distributions for the remaining datasets not shown in the main text. The same interpretation applies as in Figure 3: a larger positive gap indicates clearer separation between the gold target and the hardest negative candidate.

D Detailed Results on Efficiency

Table 9 reports group-wise and overall efficiency-effectiveness averages. The "w/o Decoder" row in the efficiency table measures the effectiveness-efficiency trade-off under the efficiency-profiling setting, whereas the "w/o Our decoder" row in Table 3 comes from an independent module ablation. Both are averaged over multiple random seeds under their respective settings; since they are not the same set of runs, the numbers need not match exactly. Their average MRR differs by less than 0.01, which is within the normal variation across repeated runs.

Average MRR rises from 0.668 at $k = 8$ to 0.671 at $k = 16$, while time per query rises from 0.040 to 0.048 s. Raising k to 32 adds only 0.001 MRR and more time; $k = 64$ takes longer without

Table 7: Per-dataset entity alignment results after target-pair finetuning (MRR, H@10, and H@1 where available).

Group	Dataset	ULTRA-EA		EAFM		ContextEA		
		MRR	H@10	MRR	H@10	MRR	H@10	H@1
OpenEA	D-W-15K-V1	0.461	0.713	0.582	0.777	0.665	0.839	0.570
	D-W-15K-V2	0.701	0.902	0.753	0.908	0.840	0.959	0.773
	D-Y-15K-V1	0.590	0.749	0.664	0.787	0.699	0.802	0.631
	D-Y-15K-V2	0.951	0.986	0.959	0.990	0.972	0.994	0.958
	D-W-100K-V1	0.372	0.619	0.514	0.711	0.564	0.747	0.471
	D-W-100K-V2	0.598	0.809	0.630	0.791	0.732	0.880	0.651
	D-Y-100K-V1	0.588	0.791	0.730	0.873	0.745	0.878	0.673
	D-Y-100K-V2	0.802	0.919	0.876	0.946	0.916	0.973	0.883
	EN-DE-15K-V1	0.617	0.830	0.729	0.890	0.782	0.931	0.705
	EN-DE-15K-V2	0.784	0.912	0.805	0.914	0.887	0.967	0.838
	EN-FR-15K-V1	0.284	0.547	0.528	0.778	0.617	0.840	0.502
	EN-FR-15K-V2	0.297	0.537	0.572	0.814	0.702	0.907	0.592
	EN-DE-100K-V1	0.416	0.592	0.557	0.728	0.583	0.745	0.502
	EN-DE-100K-V2	0.278	0.464	0.674	0.796	0.757	0.867	0.697
	EN-FR-100K-V1	0.208	0.360	0.431	0.647	0.445	0.637	0.351
	EN-FR-100K-V2	0.193	0.328	0.491	0.684	0.628	0.802	0.536
SRPRS	D-W-V1	0.288	0.462	0.431	0.647	0.522	0.731	0.418
	D-W-V2	0.519	0.785	0.719	0.885	0.821	0.948	0.745
	D-Y-V1	0.278	0.452	0.483	0.719	0.512	0.731	0.404
	D-Y-V2	0.407	0.621	0.821	0.949	0.872	0.973	0.813
	EN-DE-V1	0.332	0.522	0.555	0.744	0.585	0.769	0.486
	EN-DE-V2	0.439	0.646	0.762	0.893	0.786	0.914	0.711
	EN-FR-V1	0.259	0.431	0.405	0.638	0.479	0.691	0.373
	EN-FR-V2	0.514	0.776	0.666	0.866	0.840	0.962	0.768
DBP	FR-EN 15K	0.075	0.245	0.343	0.572	0.474	0.659	0.375
	JA-EN 15K	0.095	0.294	0.332	0.543	0.460	0.625	0.370
	ZH-EN 15K	0.084	0.245	0.327	0.540	0.424	0.592	0.335
	D-W-100K	0.452	0.671	0.602	0.788	0.747	0.887	0.667
	D-Y-100K	0.410	0.588	0.718	0.867	0.835	0.952	0.768

Table 8: Input information and training settings of representative EA methods.

Methods	Extra input	Target-pair training	Comparison
ContextEA, EAFM, ULTRA-EA	None in the structure-only setting	No in the pretraining evaluation	Main baseline
RDGCN	Names and pretrained word vectors	Yes	Text reference
LLM4EA	LLM-generated alignment labels	Yes, using target-pair labels	LLM reference
ChatEA, EasyEA	LLM knowledge and language processing	Varies; EasyEA is training-free	LLM reference
BootEA, Dual-AMN, LightEA-I, NeuSymEA	None in these variants	Yes	Target-specific reference

improving MRR. This suggests that most useful corrections involve candidates near the top of the coarse ranking. We use $k = 16$ as a practical balance between accuracy and speed.

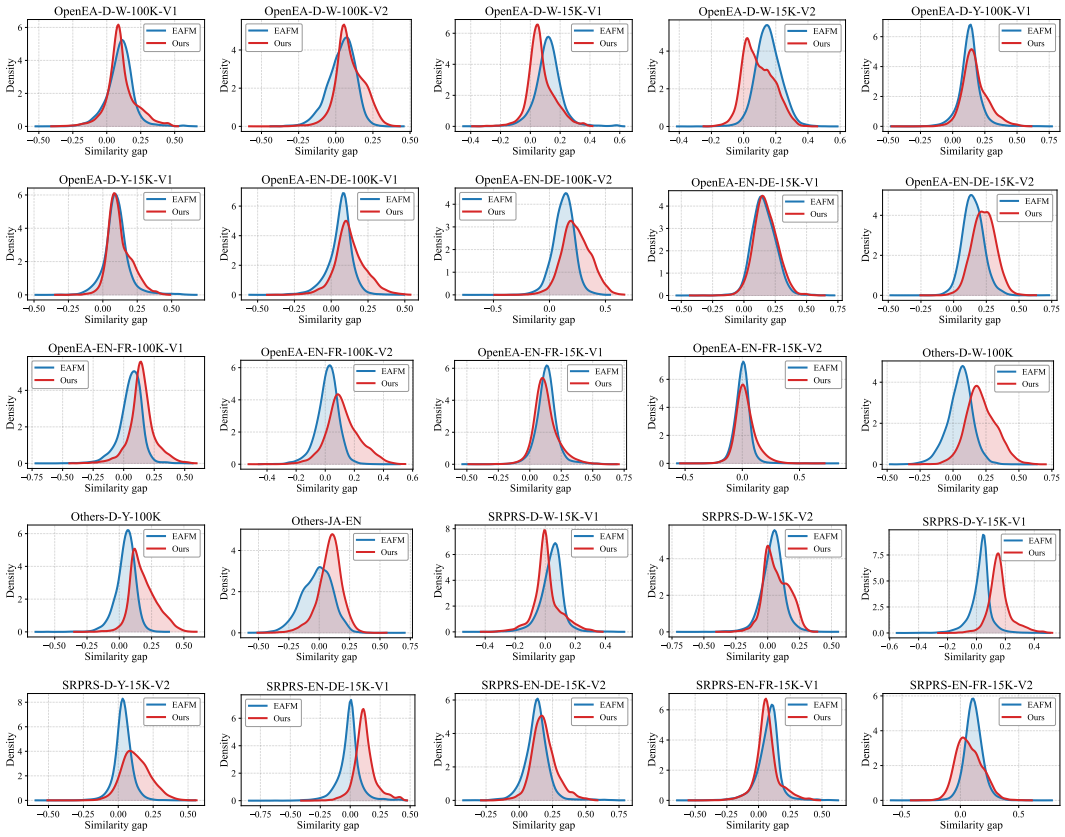


Figure 7: Additional results on gold vs. hard-negative score gaps.

Table 9: Average efficiency-effectiveness comparison under different inference settings. We report average MRR and average inference time (s) per query across dataset groups.

Method	**OpenEA**		**SRPRS**		**DBP**		**Average**	
	Time (s)	MRR	Time (s)	MRR	Time (s)	MRR	Time (s)	MRR
EAFM	0.059	0.602	0.014	0.530	0.094	0.294	0.053	0.529
ContextEA w/o Decoder	0.037	0.702	0.009	0.664	0.068	0.509	0.034	0.658
ContextEA ($k = 8$)	0.042	0.712	0.016	0.674	0.074	0.519	0.040	0.668
ContextEA ($k = 16$)	0.050	0.715	0.023	0.676	0.082	0.522	0.048	0.671
ContextEA ($k = 32$)	0.062	0.716	0.036	0.678	0.092	0.522	0.060	0.672
ContextEA ($k = 64$)	0.087	0.715	0.058	0.675	0.116	0.521	0.084	0.670

E More Results of MRR about Weight β

Table 10 reports more detailed MRR results under different values of β.

Average MRR is highest at $\beta = 0.3$, but the best value differs by dataset. A small weight gives the decoder less influence, while larger weights do not bring steady gains. We therefore use $\beta = 0.3$ as a shared setting rather than claiming it is best for every KG pair.

F Variation Across Random Seeds

Table 11 reports MRR across six runs under the same split and evaluation protocol.

Table 10: Sensitivity analysis of the structural calibration weight β using representative test datasets.

β	MRR					Average
	D-W-15K-V1	D-W-15K-V2	D-W-100K-V1	EN-DE-15K-V1	FR-EN	
0.1	0.671	0.827	0.554	0.785	0.343	0.636
0.3	0.665	0.849	0.563	0.780	0.430	0.657
0.5	0.665	0.845	0.539	0.787	0.380	0.643
1.0	0.674	0.833	0.536	0.770	0.407	0.644

Table 11: Six-seed evaluation of ContextEA. Values are MRR; the last column is mean $\pm$ standard deviation.

Group	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	Seed 6	Mean $\pm$ Std
OpenEA	0.710	0.716	0.713	0.711	0.715	0.712	0.713 ± 0.002
SRPRS	0.672	0.678	0.675	0.673	0.677	0.674	0.675 ± 0.002
DBP	0.517	0.525	0.522	0.519	0.526	0.521	0.522 ± 0.003
Overall	0.666	0.672	0.669	0.667	0.672	0.668	0.669 ± 0.002

Overall MRR ranges from 0.666 to 0.672 across the six seeds, with a standard deviation of 0.002. The small spread shows that the overall result does not depend on one favorable run. DBP varies slightly more than the other groups, so we report both its mean and standard deviation.

G Anchor Robustness

Table 12 reports four-seed MRR summaries for three perturbations of the target training anchors. The low-anchor setting reduces the available anchor ratio; the noisy-anchor setting replaces a fraction of anchors with incorrect pairs; and the missing-anchor setting removes high-degree anchors.

With the 6% anchor setting, overall MRR is 0.633, compared with 0.670 using all training anchors. It falls to 0.355 at the 1.5% anchor level, and to 0.540 when 20% of training anchors are made incorrect. Removing high-degree anchors also lowers MRR, suggesting that the location of seed links may matter as well as their number. Together, these tests show that the model needs enough correct and useful anchors; anchor quality remains a limit of structure-only transfer.

H 100K-Scale Encoder Analysis

Table 13 isolates encoder-only inference on OpenEA D-W-100K-V1. Both methods use the same GPU, 10,000 evaluation queries, and test batch size 4; the decoder is disabled. This protocol is separate from the per-dataset k sweep in Appendix D.

On this 100K dataset, the unified encoder uses 3.87 GB of process GPU memory, compared with 3.59 GB for EAFM. Its measured encoder time is 603.32 s versus 965.04 s, and its total time is also lower. Thus, cross-KG propagation has a modest memory cost in this test, without a measured time penalty. These numbers describe this dataset and hardware, not a speed claim for every graph.

I Decoder Contribution by Query Type

We further analyze decoder behavior on six 15K datasets: OpenEA D-W, D-Y, and EN-DE, and SRPRS D-W, D-Y, and EN-DE (V1 in each collection). Each dataset contributes at most 10,000 test pairs, yielding 58,350 queries in total; the decoder calibrates the top-16 coarse candidates. The coarse and calibrated rankings use the same trained checkpoint. Buckets are defined according to the coarse ranking, entity sparsity, and query-to-anchor distance before decoder calibration. Since some buckets overlap, their shares do not sum to 100%.

Table 12: Anchor robustness over four random seeds. Entries are MRR mean ± standard deviation. Low-anchor levels downsample the available training anchors; noise and missing levels are percentages of training anchors perturbed.

Setting	Level	OpenEA	SRPRS	DBP	Overall
Low-anchor	1.5%	0.403 ± 0.005	0.351 ± 0.001	0.209 ± 0.004	0.355 ± 0.003
Low-anchor	3.0%	0.575 ± 0.002	0.494 ± 0.005	0.337 ± 0.006	0.512 ± 0.002
Low-anchor	6.0%	0.694 ± 0.003	0.613 ± 0.008	0.471 ± 0.005	0.633 ± 0.003
Low-anchor	Full train anchors	0.713 ± 0.001	0.675 ± 0.000	0.522 ± 0.004	0.670 ± 0.001
Noisy-anchor	5%	0.690 ± 0.001	0.639 ± 0.001	0.498 ± 0.003	0.643 ± 0.001
Noisy-anchor	10%	0.659 ± 0.002	0.608 ± 0.001	0.465 ± 0.004	0.611 ± 0.001
Noisy-anchor	20%	0.583 ± 0.002	0.546 ± 0.002	0.392 ± 0.006	0.540 ± 0.002
Missing-anchor	10%	0.681 ± 0.001	0.631 ± 0.000	0.500 ± 0.003	0.635 ± 0.001
Missing-anchor	20%	0.652 ± 0.002	0.604 ± 0.003	0.468 ± 0.004	0.607 ± 0.002
Missing-anchor	30%	0.614 ± 0.001	0.569 ± 0.001	0.432 ± 0.005	0.570 ± 0.001

Table 13: Encoder-only efficiency on OpenEA D-W-100K-V1 with 10,000 queries and test batch size 4.

Encoder	Process GPU memory	Encoder time	Total time	Per-query time
EAFM dual-tower encoder	3.59 GB	965.04 s	1271.80 s	0.127 s
ContextEA unified encoder	3.87 GB	603.32 s	959.34 s	0.096 s

Table 14: Decoder performance by query type on six 15K datasets. Bucket shares are calculated over all evaluated queries; buckets may overlap.

Bucket	Share	Coarse MRR	Calibrated MRR	ΔMRR	ΔH@1	ΔH@10
All queries	100.0%	0.6304	0.6484	+0.0180	+0.0292	+0.0183
Gold in top-k	84.0%	0.7468	0.7608	+0.0140	+0.0325	+0.0065
Gold not in top-k	16.0%	0.0202	0.0592	+0.0390	+0.0120	+0.0800
Gold in top-k, not top-1	30.0%	0.2912	0.3322	+0.0410	+0.1000	+0.0200
Hard, small coarse score gap	15.6%	0.3321	0.3831	+0.0510	+0.1300	+0.0100
Sparse entity	60.9%	0.4756	0.4966	+0.0210	+0.0350	+0.0200
Non-sparse entity	39.1%	0.8714	0.8847	+0.0133	+0.0202	+0.0156
Near anchor	52.7%	0.7499	0.7679	+0.0180	+0.0310	+0.0150
Far or unanchored	3.5%	0.1544	0.1634	+0.0090	+0.0050	+0.0020

Calibration raises overall MRR from 0.6304 to 0.6484. The MRR gain is larger when the gold entity is in the top-k set but not ranked first (+0.0410), or when top candidates have close coarse scores (+0.0510). This fits the decoder's role of checking uncertain candidates. Sparse-entity cases also gain more than non-sparse cases, although the buckets do not show which decoder signal causes that difference.

Cases with gold outside the top-k set can improve as well. The decoder does not score the gold entity in those cases, but it can lower the scores of over-ranked top-k candidates; the final ranking still covers all target entities. The buckets overlap, so their gains cannot be added across rows. This fixed-checkpoint inference analysis is also different from the independently trained w/o-decoder ablation in Table 3; it does not isolate the effect of decoder loss during training.

J Failure-Case Study

We apply automated heuristic diagnostics to incorrect predictions on six 15K datasets: OpenEA D-W, D-Y, and EN-DE, and SRPRS D-W, D-Y, and EN-DE. At most 10,000 test pairs are evaluated per dataset, yielding 26,588 failures in total. In the primary-label analysis, each failure receives exactly one dominant diagnostic label; "general failure" means that none of the specific rules fires. The

categories are diagnostics rather than human-annotated causal explanations. Table 15 shows that low local structural density is the most frequent primary diagnostic among failed pairs.

Table 15: Primary diagnostic labels for 26,588 incorrect alignments on six 15K datasets. Percentages use the number of failed pairs as the denominator.

Failure type	Count	Share of failures
Sparse region	20,724	77.9%
General failure	3,511	13.2%
Weakly anchored	1,882	7.1%
Misleading neighborhood	385	1.4%
Heterogeneous relation schema	57	0.2%
Asymmetric structure	29	0.1%

We also compute a multi-label variant, where one failed case can receive multiple diagnostic labels. This view captures co-occurring failure signatures, such as a misleading neighborhood appearing together with sparse local evidence.

Table 16: Multi-label diagnostic coverage for the same 26,588 incorrect alignments. A failed case can contribute to multiple rows.

Failure type	Count	Coverage
Sparse region	22,400	84.2%
General failure	3,511	13.2%
Weakly anchored	2,050	7.7%
Misleading neighborhood	1,450	5.5%
Heterogeneous relation schema	310	1.2%
Asymmetric structure	250	0.9%

Sparse regions account for 77.9% of the primary labels, while weak anchor support accounts for another 7.1%. Misleading neighborhoods appear in 5.5% of cases in the multi-label view, compared with 1.4% as the main label. This shows that several warning signs can occur in one failed case. The labels describe common error patterns, but they do not prove the cause of each prediction.

Interpreting individual failure. For a sparse-region error, the relevant evidence is the number of incident triples and informative neighbors around the query and gold target; for a weak-anchor error, it is the distance to and availability of seed anchors. An asymmetric-structure error requires comparing the query and gold neighborhoods, while a heterogeneous-relation error requires checking whether their relation patterns are compatible across KGs. For a misleading-neighborhood error, the predicted wrong target and the gold target should be compared against the same query: a similar local neighborhood around the wrong target can make the coarse scores hard to distinguish. Accordingly, the aggregate diagnostic labels should not be interpreted as verified causes for individual errors.

K Structural Decoder Feature Definitions

This section specifies the four views used in Section 4.3; $\odot$ denotes element-wise multiplication. Neighborhood and relation pools use the same cached one-hop entries, capped at 24 per entity. All mean pools over empty neighbor or relation sets return zero vectors.

Direct entity compatibility. The entity view compares the query and candidate encoder states:

$$\phi_{\text{ent}}(q, c) = [|\mathbf{h}_q - \mathbf{h}_c|; \mathbf{h}_q \odot \mathbf{h}_c]. \tag{6}$$

Neighborhood structure. For $x \in \{q, c\}$, let $\mathcal{N}(x)$ be its one-hop neighbor entities and $\mathbf{g}_x = \text{MeanPool}(\{\mathbf{h}_n \mid n \in \mathcal{N}(x)\})$. The neighborhood view is

$$\phi_{\text{nei}}(q, c) = [|\mathbf{g}_q - \mathbf{g}_c|; \mathbf{g}_q \odot \mathbf{g}_c]. \tag{7}$$

Relation patterns. For $x \in \{q, c\}$, let $\mathcal{R}(x)$ be its adjacent one-hop relations and $\mathbf{r}_x = \mathrm{MeanPool}(\{\mathbf{r}_\ell \mid \ell \in \mathcal{R}(x)\})$, where $\mathbf{r}_\ell$ is the encoded relation state for the current query. Then

$$\phi_{\mathrm{rel}}(q, c) = [|\mathbf{r}_q - \mathbf{r}_c|; \mathbf{r}_q \odot \mathbf{r}_c]. \tag{8}$$

Anchor-supported local structure. The entry set $\mathcal{N}_e(x)$ in Eq. (3) includes at most 24 one-hop relation–neighbor entries, including inverse KG relations but excluding bridges. Let I_q and I_c be the distinct query- and candidate-side entry indices in $P(q, c)$, and let $n_q = |\mathcal{N}_e(q)|$ and $n_c = |\mathcal{N}_e(c)|$. We mean-pool the ℓ_2-normalized entity states indexed by I_q and I_c into $\mathbf{z}_{aq}$ and $\mathbf{z}_{ac}$. The six scalar features are $|I_q|/\max(n_q, 1)$, $|I_c|/\max(n_c, 1)$, $|P(q, c)|/\max(\min(n_q, n_c), 1)$, the mean and maximum relation-consistency scores over $P(q, c)$, and the mean $(1 + \cos(\mathbf{h}_{u_i}, \mathbf{h}_{v_j}))/2$ over $P(q, c)$. A relation pair (r, s) has consistency $C(r, s)/\sqrt{C(r, *)C(*, s) + 10^{-12}}$, where C counts paired neighbor relations supported by training anchors. If $P(q, c)$ is empty, both pooled vectors and all six scalars are zero. The anchor-aware view is

$$\phi_{\mathrm{anc}}(q, c) = [\mathbf{z}_{aq}; \mathbf{z}_{ac}; |\mathbf{z}_{aq} - \mathbf{z}_{ac}|; \mathbf{z}_{aq} \odot \mathbf{z}_{ac}; \phi_{\mathrm{anc}}^{\mathrm{scalar}}(q, c)]. \tag{9}$$

L LIMITATIONS

Our method still has several limitations. First, the evaluation is based on existing EA benchmarks, so the conclusions may still depend on the benchmark construction and data quality. Second, the method relies on training anchors to support cross-KG interaction and structural calibration. When anchors are very sparse or noisy, the benefit may become smaller. Third, although the decoder is lightweight, its inference cost still grows with the candidate size k.

M BROADER IMPACTS

This work mainly improves transferable EA on public KG benchmarks. A possible positive impact is that better EA can support knowledge fusion and improve the quality of downstream knowledge-driven systems. A possible risk is that stronger alignment tools may also be used to link entities across data sources in settings where data integration should be carefully controlled. We therefore view this work as a general technical contribution, and responsible use should depend on the application context and data governance requirements.