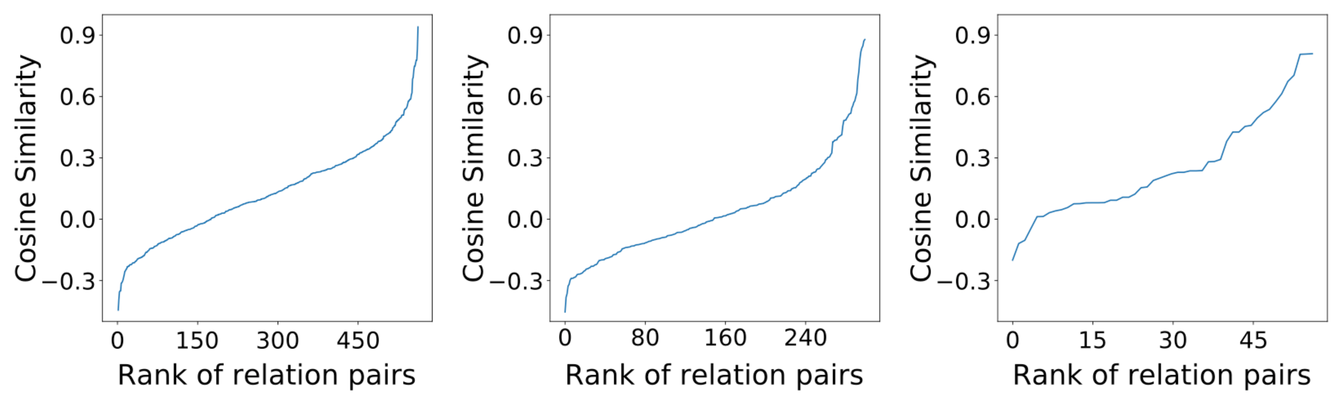
Context-Aware Adapter Tuning for Few-Shot Relation Learning in Knowledge Graphs

Ran Liu[1], Zhongzhou Liu[1], Xiaoli Li[2,3], Yuan Fang[1]

Singapore Management University[1]
Institute for Infocomm Research, A*STAR, Singapore[2]
A*STAR Centre for Frontier AI Research, Singapore[3]

SMU
SINGAPORE MANAGEMENT UNIVERSITY

MOTIVATION



(a) WIKI (b) FB15K-237 (c) UMLS

Figure 1: Pairwise cosine similarity of relations.

Contributions

1. Integrate a **lightweight adapter module** into the meta-learning framework to **enable relation-specific adaptations** of global prior to local task in the meta-testing stage.

2. Inject additional **contextual information** about the target relation into meta-testing to **enrich the few-shot relation instances** for better adaptation to novel relations.

Problem

- Assumption in conventional meta-learning: meta-training and meta-testing tasks are **independently and identically distributed (i.i.d.)**.
- Due to the **data distribution shift**, the learned function from meta-training relation tasks would not be able to make accurate predictions for downstream novel relation tasks in meta-testing.

Preliminaries & Methodology

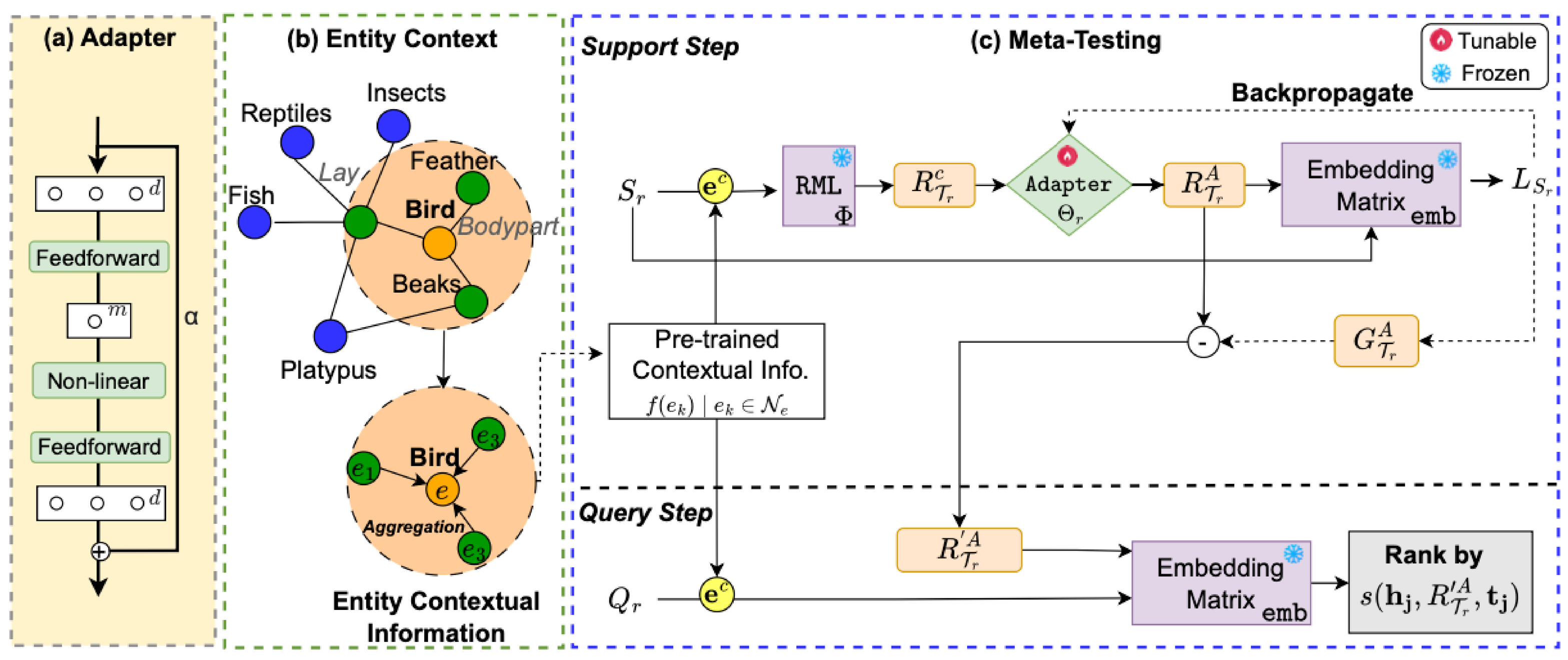


Figure 2: Illustration of key concepts in RelAdapter, hinging on an entity-aware adapter (a, b) in the meta-testing stage (c). Note that we omit the meta-training stage, which is similar to meta-testing but with backpropagation of the query loss to update the model parameters (emb and Φ).

Adapter Structure

$$R^A_{\mathcal{T}_r} = \mathtt{Adapter}(R_{\mathcal{T}_r};\Theta_r) = \alpha\cdot\mathtt{FFN}(R_{\mathcal{T}_r};\Theta_r) + (1-\alpha)\cdot R_{\mathcal{T}_r}$$

Contextual information

$$\mathbf{e}^c = \mu\cdot\mathtt{Mean}(\{f(e_k)\mid e_k\in\mathcal{N}_e\}) + (1-\mu)\cdot\mathtt{emb}(e)$$

Context-aware relation meta

$$R^c_{\mathcal{T}_r} = \mathtt{Mean}(\{\mathtt{RML}(\mathbf{h}^c,\mathbf{t}^c;\Phi)\mid\mathcal{S}_r\})$$

Relation-specific adaptation

$$R^A_{\mathcal{T}_r} = \mathtt{Adapter}(R^c_{\mathcal{T}_r};\Theta_r)$$

Gradient update

$$G^A_{\mathcal{T}_r} = \nabla_{R^A_{\mathcal{T}_r}} L_{\mathcal{S}_r},$$
$$R'^A_{\mathcal{T}_r} = R^A_{\mathcal{T}_r} - \beta G^A_{\mathcal{T}_r}.$$

Support & Query loss

$$L_{\mathcal{S}_r} = \sum_{(h,r,t)\in\mathcal{S}_r}[\gamma + s(\mathtt{emb}(h),R_{\mathcal{T}_r},\mathtt{emb}(t)) - s(\mathtt{emb}(h),R_{\mathcal{T}_r},\mathtt{emb}(t'))]_+$$
$$L_{\mathcal{Q}_r} = \sum_{(h,r,t)\in\mathcal{Q}_r}[\gamma + s(\mathtt{emb}(h),R'_{\mathcal{T}_r},\mathtt{emb}(t)) - s(\mathtt{emb}(h),R'_{\mathcal{T}_r},\mathtt{emb}(t'))]_+$$

Ranking score

$$s\left(\mathrm{emb}(h),R'^A_{\mathcal{T}_r},\mathrm{emb}(t)\right) = \|\mathrm{emb}(h)+R'^A_{\mathcal{T}_r}-\mathrm{emb}(t)\|$$

Experiments & Conclusion

Table 2: Performance comparison against baselines in the 3-shot setting. (Best: bolded, runners-up: underlined).

	Models	WIKI MRR	WIKI Hit@10	WIKI Hit@1	FB15K-237 MRR	FB15K-237 Hit@10	FB15K-237 Hit@1	UMLS MRR	UMLS Hit@10	UMLS Hit@1
Supervised Relation Learning	TransE	.031±.007	.043±.012	.021±.014	.294±.005	.437±.011	.204±.014	.178±.036	.310±.051	.146±.068
	DistMult	.047±.003	.082±.009	.031±.011	.234±.008	.364±.007	.208±.010	.231±.035	.337±.049	.214±.067
	ComplEx	.093±.004	.166±.011	.071±.012	.239±.007	.359±.010	.205±.013	.251±.038	.351±.041	.227±.058
	RGCN	.217±.012	.363±.023	.188±.031	.332±.011	.495±.013	.241±.031	.409±.059	.549±.072	.389±.089
Few-shot Relation Learning	GMatching	.133±.017	.331±.013	.114±.026	.309±.019	.441±.015	.245±.019	.296±.059	.532±.040	.257±.087
	FSKGC	.131±.003	.267±.010	.104±.016	.355±.005	.523±.004	.217±.011	.525±.031	.682±.024	.490±.038
	GANA	.291±.014	.384±.012	.272±.015	<u>.388</u>±.004	.553±.008	**.301**±.017	.541±.045	.721±.076	.502±.047
	FAAN	.278±.018	.421±.020	.275±.024	.363±.009	.542±.007	.279±.013	.545±.034	.746±.120	.505±.068
	HiRe	.300±.028	<u>.444</u>±.012	<u>.282</u>±.015	.378±.013	<u>.571</u>±.011	.281±.015	<u>.577</u>±.060	<u>.752</u>±.066	<u>.533</u>±.089
	MetaR	<u>.314</u>±.013	.420±.016	.274±.028	.368±.007	.536±.005	.251±.012	.435±.075	.601±.095	.417±.103
	RelAdapter	**.347**±.006	**.454**±.012	**.317**±.013	**.405**±.012	**.575**±.014	<u>.297</u>±.019	**.608**±.067	**.780**±.044	**.555**±.062

Compared to backbone : The big performance improvement of RelAdapter compared to MetaR[1] shows the importance of adding context-ware adapter in the meta-learning framework, enabling relation-specific and context aware adaptation.

Table 4: Number of parameters of our adapter w.r.t. MetaR.

	WIKI	FB15K-237	UMLS
MetaR	241,967,556	1,650,206	234,306
Our adapter	5,125	5,125	5,125
% of MetaR	0.002	0.311	2.187

Negligible parameter overhead for Adapter

Table 5: Runtime (in seconds) for meta-training and meta-testing.

Meta-learning stage	Model	WIKI	FB15K-237	UMLS
Meta-train (total runtime)	MetaR	22,691	16,504	8,802
	RelAdapter	24,085	17,529	9,656
Meta-test (per prediction)	MetaR	0.012	0.005	0.043
	RelAdapter	0.045	0.008	0.053

Parameter efficiency : Compared to MetaR, the only new parameters of RelAdapter belong to the adapter module, which is only a small overhead

ACKNOWLEDGEMENT & REFERENCES

[1] Mingyang Chen, et al. MetaR: Meta relational learning for few-shot link prediction in knowledge graphs. EMNLP 2019.

For complete references please refer to https://arxiv.org/abs/2410.09123

Paper

SCHOOL OF COMPUTING AND INFORMATION SYSTEMS