NEURAL COMPOSITIONAL RULE LEARNING FOR KNOWLEDGE GRAPH REASONING

Kewei Cheng *
Department of Computer Science, UCLA
viviancheng@cs.ucla.edu

Nesreen K. Ahmed
Intel Labs
nesreen.k.ahmed@intel.com

Yizhou Sun
Department of Computer Science, UCLA
yzsun@cs.ucla.edu

A APPENDIX

A.1 AN EXAMPLE TO ILLUSTRATE HOW NCRL LEARNS RULES

Fig. 3 in the main content gives an example to illustrate how our NCRL learns logical rules. In this example, we consider a sampled path $p = [r_1, r_3, r_4, r_5, r_3]$, and predict the relations that directly connect the sampled paths (i.e.,r_6). First of all, we split path p into short compositions using a sliding window with a size of 2. Then, NCRL reasons over all the compositions and select the second window $w_2 = [r_3, r_4]$ as the first step. After that, NCRL uses a recurrent attention unit to transform w_2 into a single embedding $\widehat{(r_3, r_4)}$. By replacing the embedding sequence $[\mathbf{r_3}, \mathbf{r_4}]$ with the single embedding $\widehat{(r_3, r_4)}$, we reduce the p from $[r_1, r_3, r_4, r_5, r_3]$ to $[r_1, \widehat{(r_3, r_4)}, r_5, r_3]$. Following the same process, we continually reduce p into $[r_1, \widehat{(\widehat{(r_3, r_4)}, r_5)}, r_3]$ and $[r_1, \widehat{(\widehat{(\widehat{(r_3, r_4)}, r_5)}, r_3)}]$. In the final step of the prediction, we compute the attention θ following Eq. 4. θ collects the predicted probability that the rule body can be closed by each of the head relations. We compared the learn θ with the ground truth one-hot vector $[0, 0, 0, 0, 0, 1]$ (i.e., one-hot encoded vector of r_6) to minimize the cross-entropy loss. Algorithm 1 in the main content also outlines the learning procedure of NCRL.

A.2 EXPERIMENTAL SETUP

NCRL is implemented over PyTorch and trained in an end-to-end manner. Adam Kingma & Ba (2014) is adopted as the optimizer. Embeddings of all predicates are uniformly initialized and no regularization is imposed on them. To fairly compare with different baseline methods, we set the parameters for all baseline methods by a grid search strategy. The best results of baseline methods are used to compare with NCRL. All the experiments are run on Tesla V100 GPUs.

A.3 HYPERPARAMETER SETTINGS

Adam (Kingma & Ba, 2014) is adopted as the optimizer. We set the parameters for all methods by a grid search strategy. The range of different parameters is set as follows: embedding dimension $k \in \{128, 256, 512, 1024, 2048\}$, batch size $b \in \{500, 1,000, 5,000, 8,000\}$, learning rate $lr \in \{0.00001, 0.000025, 0.00005, 0.0001, 0.0005\}$ and epochs $i \in \{1,000, 2,000, 5,000, 10,000\}$. Afterward, we compare the best results of different methods. The detailed hyperparameter settings can be found in Table 1. Both the relation embeddings are uniformly initialized and no regularization is imposed on them.

*work was done when author was an intern at Intel Labs

Table 1: The best hyperparameter setting of NCRL on several benchmarks.

Dataset	Batch Size	# Open Paths Sampling Ratio	Embedding Dim	Epoches	Maximum Length	Learning Rate
Family	500	0.1	512	1,000	3	0.0001
Kinship	1,000	0.1	1024	2,000	3	0.00025
UMLS	1,000	0.1	512	2,000	3	0.00025
WN18RR	5,000	0.1	1024	2,000	3	0.0001
FB15K-237	5,000	0.1	1024	5,000	3	0.0005
YAGO3-10	8,000	0.1	1024	5,000	3	0.00025

A.4 Knowledge Graph Completion

A.4.1 Datasets

The detailed statistics of three large-scale real-world KGs are provided in Table 2. FB15K237, WN18RR, and YAGO3-10 are the most widely used large-scale benchmark datasets for the KG completion task, which don't suffer from test triple leakage in the training set. In addition, three small-scale KGs are also included in our experiments. The Family dataset is selected due to better interpretability and high intuitiveness. The Unified Medical Language System (UMLS) dataset is from bio-medicine: entities are biomedical concepts, and relations include treatments and diagnoses. The Kinship dataset contains kinship relationships among members of the Alyawarra tribe from Central Australia. Because inverse relations are required to learn rules, we preprocess the KGs to add inverse links.

Table 2: Data statistics of widely used benchmark knowledge graphs.

Dataset	# Data	# Relation	# Entity
Family	28,356	12	3007
UMLS	5,960	46	135
Kinship	9,587	25	104
FB15K-237	310,116	237	14,541
WN18RR	93,003	11	40,943
YAGO3-10	1,089,040	37	123,182

A.4.2 Ablation Study

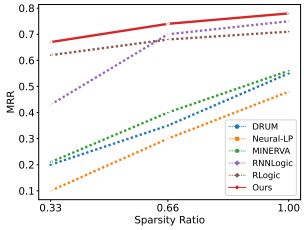

Figure 1: KG completion performance w.r.t. sparsity ratio on UMLS dataset.

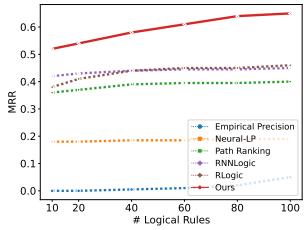

Figure 2: KG completion performance w.r.t. # logical rules on WN18RR dataset.

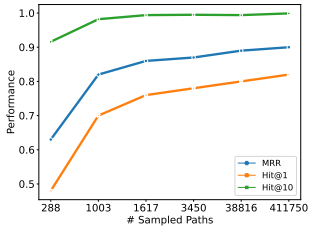

Figure 3: KG completion performance w.r.t. # sampled paths on Family dataset.

Performance w.r.t. Data Sparsity We present more analysis about the performance of NCRL against other baseline methods on UMLS dataset w.r.t. Data Sparsity in Fig. 1. We have a similar observation as we did on Kinship dataset. The performance of NCRL does not vary a lot with different sparsity ratios θ.

Table 3: KG completion performance w.r.t. random deduction order v.s. hierarchical structure learning

Methods	Family			Kinship			UMLS		
	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10
NCRL w/o Hierarchical Structure Learning	0.84	74.6	92.6	0.34	20.1	69.3	0.55	37.7	88.7
NCRL	**0.92**	**85.6**	**99.6**	**0.65**	**49.4**	**93.6**	**0.78**	**66.1**	**95.2**

Table 4: KG completion performance w.r.t. randomness caused by random sampling

	Family			Kinship			UMLS		
	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10
Mean	0.91	85.1	99.1	0.64	48.7	92.5	0.78	65.9	95.0
Standard Deviation	0.005	0.374	0.216	0.000	0.250	0.287	0.005	0.327	0.262
	WN18RR			FB15K-237			YAGO3-10		
	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10
Mean	0.66	56.2	85.0	0.29	20.4	46.8	0.38	27.2	53.3
Standard Deviation	0.005	0.262	0.205	0.005	0.374	0.411	0.005	0.478	0.340

KG completion performance w.r.t. the Number of Learned Rules We present more analysis about the performance of NCRL against other baseline methods on WN18RR dataset w.r.t. the number of learned rules in Fig. 2. We have a similar observation as we did on Kinship dataset.

Performance w.r.t. # Sampled Paths To investigate how the number of sampled paths affects the performance, we vary the number of sampled paths among $\{288, 1003, 1617, 3450, 38816, 411750\}$ and report performance in terms of KG completion task on Family dataset in Fig. 3. We observe that the performances increase severely when the number of sampled closed paths increases from 288 to 1003. After that the performance stay steady. This is attractive in real-world application as a small number of sampled closed paths can already gives great performance.

Hierarchical Structure Learning As stated earlier, learning an accurate hierarchical structure is significant for rule discovering. To investigate how the hierarchical structure learning affects the performance, we follows a random deduction order to induce rules and compare against NCRL in term of KG completion performance. From Table 3, we can observe that the KG completion performance drastically decreases if we didn't follow a correct order to learn the rules.

Performance w.r.t. Randomness Caused by Random Sampling Since the random path sampling may result in unstable performance, to investigate how the randomness affects the performance, we report the results of different runs of the proposed NCRL in terms of KG completion task using the same hyperparameter. From Table 4, we can observe that the KG completion performance is hardly affected by the randomness caused by random sampling, which is attractive in practice.

Performance w.r.t. Size of Sliding Windows To investigate how the size of sliding windows affect the performance, we set the window size to $\{2, 3\}$ and present the performance of NCRL in term of KG completion on Family, Kinship, and UMLS datasets in Table 5. We can see that we consistently achieve the best performance by setting the window size as 2. The major reason is that we apply rules with a maximum length of 3 for the KG completion task. In this case, NCRL cannot leverage the compositionality by setting window size as 3 and thus results in worse performance.

Performance w.r.t. Different Types of Sliding Window Encoder To investigate how different sliding window encoders affect the performance, in addition to RNN, we also encode the sliding window using (1) a Transformer; and (2) a standard MLP, which takes the concatenation of all pred-

Table 5: KG completion performance w.r.t. size of sliding windows

Window Size	Family			Kinship			UMLS		
	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10
2	**0.92**	**85.6**	**99.6**	**0.65**	**49.4**	**93.6**	**0.78**	**66.1**	**95.2**
3	0.90	82.3	99.5	0.60	43.1	88.9	0.72	59.9	89.3

Table 6: KG completion performance w.r.t. types of sliding window encoder

Encoder	Family			Kinship			UMLS		
	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10
MLP	0.88	80.2	98.0	0.56	38.0	84.9	0.69	55.8	87.4
Transformer	0.84	75.6	95.3	0.50	33.7	79.1	0.62	48.8	81.2
RNN	**0.92**	**85.6**	**99.6**	**0.65**	**49.4**	**93.6**	**0.78**	**66.1**	**95.2**

Table 7: Data statistics of CLUTRR datasets.

Dataset	# Relation	# Train	# Test
CLUTRR	22	15,083	823

icates covered by the sliding window as the input and outputs a new embedding $\mathbf{w_i} \in \mathcal{R}^d$. We present the performance of NCRL with different sliding window encoders in terms of KG completion on Family, Kinship, and UMLS dataset in Table 6. We can observe that the RNN encoder, as the most effective and efficient sequence model, gives the best performance due to the sequential nature of the subsequences extracted by the sliding windows. Although Transformer is also a sequence model, it suffers from an overfitting issue caused by the large parameter space, which results in bad performance.

A.5 SYSTEMATICITY

A.5.1 DATASETS

Clean Data CLUTRR Sinha et al. (2019) is a dataset for inductive reasoning over family relations. The goal is to infer the missing relation between two family members. The train set contains graphs with up to 4 hops paths, and the test set contains graphs with up to 10 hops path. The train and test splits share disjoint set of entities. The detailed statistics of CLUTRR is provided in Table 7.

Noisy Data Graphlog Sinha et al. (2020) is a benchmark dataset designed to evaluate systematicity. It contains logical worlds where each world contains graphs that are created using a different set of ground rules. The goal is to infer the relation between a queried node pair. GraphLog contains more bad examples than CLUTRR does. The detailed statistics of Graphlog is provided in Table 8. When the ARL is larger, the dataset becomes noisier and contains more bad cases.

A.5.2 SYSTEMATIC GENERALIZATION ON CLUTRR

Baseline methods. We evaluate our method against several SOTA algorithms, including (1) logical rule learning methods (e.g., CTP (Minervini et al., 2020b), R5 (Lu et al., 2022) and RLogic (Cheng et al., 2022)); (2) sequential models (e.g., Recurrent Neural Networks (RNN) (Schuster & Paliwal, 1997), Long Short-Term Memory Networks (LSTMs) (Graves, 2012), GRU (Chung et al., 2014) and Transformer (Vaswani et al., 2017)); (3) embedding-based models (e.g., GAT (Veličković et al., 2017) and GCN (Kipf & Welling, 2016)); (4) neural theorem provers, including GNTP (Minervini et al., 2020a).

Table 8: Data statistics of GraphLog datasets. NC: number of classes. ND: number of distinct resolution edge sequences (distinct descriptors). ARL: average resolution length. AN: average number of nodes. AE: average number of edges.

Dataset	NC	ND	ARL	AN	AE	#Train	#Test
World 2	17	157	3.21	9.8	11.2	5000	1000
World 3	16	189	3.63	11.1	13.3	5000	1000
World 6	16	249	5.06	16.3	20.2	5000	1000
World 8	15	404	5.43	16.0	19.1	5000	1000

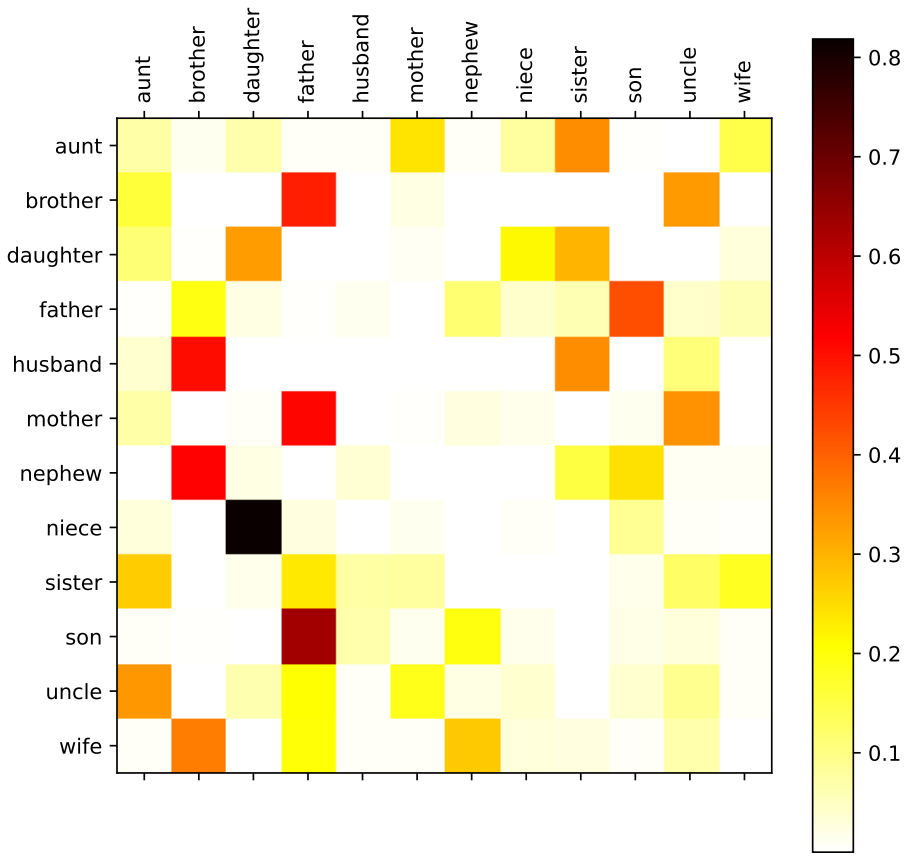

Figure 4: Visualization of the attention learned on Family dataset.

A.5.3 Interpretable Self-Attention

This section discusses whether the attention correctly captures the semantic of relations. The visualization of the attention learned over Family dataset and WN18RR dataset are given in Fig. 4 and Fig. 5.

A.6 Complexity Analysis

To theoretically demonstrate the superiority of our proposed NCRL in terms of efficiency, we compare the space and time complexity of NCRL and backward-chaining methods - NeuralLP as presented in Table 9. We denote $|E|/|R|/|O|/l/a/d$ as the number of entities/relations/observed triples/length of rule body/number of sampled paths/dimension of the embedding space. We can observe that: (1) For space complexity, our proposed NCRL is more efficient compared to NeuralLP since $d \ll |E|^2$; (2) For time complexity, our proposed NCRL is also more efficient than NeuralLP.

References

Kewei Cheng, Jiahao Liu, Wei Wang, and Yizhou Sun. Rlogic: Recursive logical rule learning from knowledge graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge*

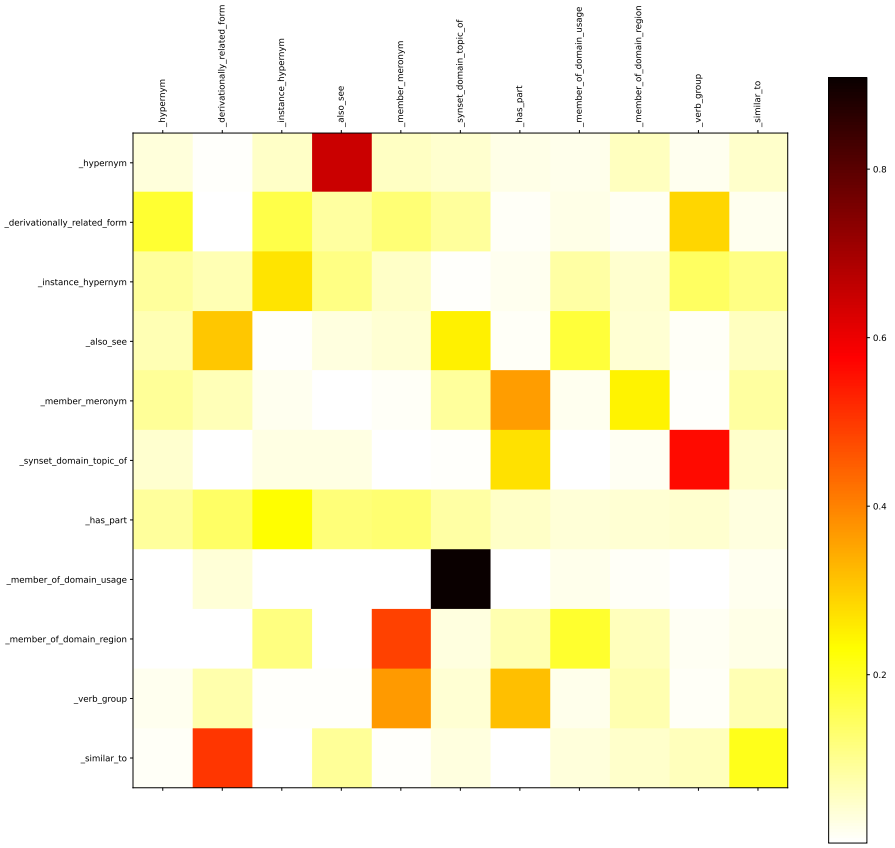

Figure 5: Visualization of the attention learned on WN18RR dataset.

Table 9: Comparison of Space and Time Complexity for Model Training.

Method	Space Complexity	Time Complexity
NeuralLP	$\mathcal{O}(\|R\|\|E\|^2 + 3\|O\|)$	$\mathcal{O}(\|R\|^l\|E\|^{3(l-1)})$
NCRL	$\mathcal{O}(\|R\|d + al)$	$\mathcal{O}(2ad^2)$

Discovery and Data Mining, pp. 179–189, 2022.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Alex Graves. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Shengyao Lu, Bang Liu, Keith G Mills, SHANGLING JUI, and Di Niu. R5: Rule discovery with reinforced and recurrent relational reasoning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=2eXhNpHeW6E.

Pasquale Minervini, Matko Bošnjak, Tim Rocktäschel, Sebastian Riedel, and Edward Grefenstette. Differentiable reasoning on large knowledge bases and natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 5182–5190, 2020a.

Pasquale Minervini, Sebastian Riedel, Pontus Stenetorp, Edward Grefenstette, and Tim Rocktäschel. Learning reasoning strategies in end-to-end differentiable proving. In *International Conference on Machine Learning*, pp. 6938–6949. PMLR, 2020b.

Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L Hamilton. Clutrr: A diagnostic benchmark for inductive reasoning from text. *arXiv preprint arXiv:1908.06177*, 2019.

Koustuv Sinha, Shagun Sodhani, Joelle Pineau, and William L Hamilton. Evaluating logical generalization in graph neural networks. *arXiv preprint arXiv:2003.06560*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.