Simple Augmentations of Logical Rules for Neuro-Symbolic Knowledge Graph Completion

Ananjan Nandi Navdeep Kaur Parag Singla Mausam
Indian Institute of Technology, Delhi
{tgk.ananjan, navdeepkjohal}@gmail.com {parags, mausam}@cse.iitd.ac.in

Abstract

High-quality and high-coverage rule sets are imperative to the success of Neuro-Symbolic Knowledge Graph Completion (NS-KGC) models, because they form the basis of all symbolic inferences. Recent literature builds neural models for generating rule sets, however, preliminary experiments show that they struggle with maintaining high coverage. In this work, we suggest three simple augmentations to existing rule sets: (1) transforming rules to their abductive forms, (2) generating equivalent rules that use inverse forms of constituent relations and (3) random walks that propose new rules. Finally, we prune potentially low quality rules. Experiments over four datasets and five ruleset-baseline settings suggest that these simple augmentations consistently improve results, and obtain up to 7.1 pt MRR and 8.5 pt Hits@1 gains over using rules without augmentations.

1 Introduction

Knowledge Graphs (KGs) comprise important world knowledge facts, but are typically incomplete, due to their ever-increasing size. KG embeddings (Wang et al., 2017) has been the dominant methodology for knowledge graph completion (KGC). A KG embedding approach represents entities and relations as learnable dense vectors and computes a score for an unseen fact as a function over them. These generally have state-of-the-art performance, especially for large KGs.

Recently, neuro-symbolic (NS-KGC) approaches for the task have been proposed, where KG embeddings are enhanced by inferences over an explicit first-order logic rule set (Zhang et al., 2020; Qu et al., 2021). The resulting models bring together best of both worlds – generalizability and interpretability of explicit logical rules, and the scalability and representation power of embeddings. Unfortunately, a key roadblock for success of NS-KGC is the availability of a high-coverage rule set.

Early NS-KGC methods, such as NeuralLP (Yang et al., 2017) and DRUM (Sadeghian et al., 2019), learn rules as part of a single model, but do not have performance competitive with embedding models such as RotatE (Sun et al., 2019). A recent NS-KGC model, RNNLogic (Qu et al., 2021), matches empirical performance with embedding approaches. It has a separate neural component that outputs a set of rules, which is then used to train inference parameters, in an EM-based approach. Preliminary experiments on RNNLogic suggest that its ruleset has limited coverage, due to which symbolic inferences do not fire for many queries, and the model gets limited to using its embedding part only. The goal of this work is to strengthen the symbolic inferences in NS-KGC models for better overall performance.

In this work, we propose simple augmentations that take an existing ruleset (such as one output by RNNLogic) and proposes additional (related) rules to improve coverage and quality. We propose three augmentations. First, we convert each deductive rule into its abductive counterparts. Second, we supplement each rule via an equivalent rule that uses inverses for all constituent relations. Third, we generate additional high-quality rules independently by local random walks and subsequent PCA filtering (Galárraga et al., 2013). These increase size of ruleset drastically; we balance runtimes by additionally pruning rules from existing set using our filtering approach. Overall, this results in a comparable number of high-quality and high-coverage rules, for use in NS-KGC.

On four KGC datasets, over three NS-KGC models, we find that our augmentations consistently improve KGC performance, outperforming no augmentation baselines by up to 7.1 MRR and 8.5 Hits@1 pts. We believe our augmentations should become standard practice over any ruleset for NS-KGC. We release our code [1] and rulesets.

[1] https://github.com/dair-iitd/NS-KGC-AUG

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 256–269

2 Background and Related Work

We are given an incomplete KG $\mathcal{K} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$ consisting of entities $\mathcal{E}$, relation set $\mathcal{R}$ and set $\mathcal{T} = \{(\mathtt{h}, \mathtt{r}, \mathtt{t})\}$ of triples. Our goal is to predict the validity of any triple not present in $\mathcal{T}$.

Related Work: Existing work on NS-KGC can roughly be characterized into four types. One approach is to use attention over relations to learn end-to-end differentiable models (Yang et al., 2017; Sadeghian et al., 2019). A second approach, which includes Minerva (Das et al., 2018) and DeepPath (Xiong et al., 2017), uses RL to train an agent to find reasoning paths for KG completion. These approaches are not yet competitive to KG embedding models for large datasets. Thirdly, models like ExpressGNN (Zhang et al., 2020) and RNN-Logic use variational inference to assess plausibility of a given triple. We experiment with both these models in this paper. The final type includes UNIKER (Cheng et al., 2021) and RUGE (Guo et al., 2018), which integrate embeddings alongside traditional rules learnt via ILP models. We believe that our augmented rules can benefit these works too. Since our experiments are based on RNN-Logic, ExpressGNN and we utilize PCA scores for filtering, we describe these in some detail next.

RNNLogic+: As a pre-processing step, for every $r \in \mathcal{R}$, RNNLogic adds a relation r^{-1} to $\mathcal{R}$, and corresponding facts using inverse relations to $\mathcal{T}$. RNNLogic first produces a set of first order rules ($\mathcal{L}$) using an LSTM which are used by the RNN-Logic+ predictor to compute the score of a given triple. Given a query $(\mathtt{h}, \mathtt{r}, ?)$, the candidate answer $\mathtt{o}$ is scored by RNNLogic+ as:

$$\mathtt{scor}(\mathtt{o}) = \mathtt{MLP}\big(\mathtt{PNA}(\{\mathbf{v}_\mathtt{l} \,|\, \#(\mathtt{h}, \mathtt{l}, \mathtt{o})\}_{\mathtt{l} \in \mathcal{L}})\big) \quad (1)$$

where the learnable embedding $\mathbf{v}_\mathtt{l}$ of a given rule $\mathtt{l} \in \mathcal{L}$ is weighted by the number of groundings (#) that triple $(\mathtt{h}, \mathtt{r}, \mathtt{o})$ satisfies in the rule $\mathtt{l}$'s body. The resulting weighted embeddings of all rules are aggregated by employing PNA aggregator (Corso et al., 2020) and this aggregated embedding is passed through an MLP to obtain a final score.

The authors designed another scoring function that incorporates RotatE (Sun et al., 2019) into the scoring function, $\mathtt{scor}(\mathtt{o})$, in equation (1) where the goal is to exploit the knowledge encoded in the KG embeddings. The resulting scoring function is:

$$\mathtt{score}_{\mathtt{KGE}}(\mathtt{o}) = \mathtt{scor}(\mathtt{o}) + \eta\, \mathtt{RotatE}\,(\mathtt{h}, \mathtt{r}, \mathtt{o}) \quad (2)$$

where $\mathtt{RotatE}\,(\mathtt{h}, \mathtt{r}, \mathtt{o})$ is the score of the triple obtained from RotatE , and η is a hyper-parameter. $\mathtt{RotatE}\,(\mathtt{h}, \mathtt{r}, \mathtt{o})$ is the negation of the value obtained by rotating the embedding for $\mathtt{h}$ by the rotation transformation defined by the embedding of $\mathtt{r}$ in complex space and computing the distance from the embedding of $\mathtt{t}$. Please refer to Appendix B for further details.

ExpressGNN: It is a novel model that integrates Markov Logic Networks (MLN) (Richardson and Domingos, 2006) and Graph Neural Networks (GNN) (Kipf and Welling, 2017) to exploit their complementary strengths. An open-world paradigm is adopted in which a fact that is unknown in KG is assumed to be hidden (not false). The joint distribution of the observed and hidden triples of the KG in the MLN is optimized by employing a variational EM framework where the variational posterior distribution of the hidden variables is encoded as a GNN. Please refer to (Zhang et al., 2020) for further details about the model.

PCA Score: It is a symbolic rule confidence metric proposed in AMIE (2013) – see Appendix M for details. Broadly, it is the number of positive examples satisfied by a rule, divided by the total number of tails reached by the rule from heads occurring in the training dataset. Its performance in the context of AMIE was not as good due to its purely symbolic approach, and we are likely the first to show its utility in the context of NS-KGC.

3 Rule Augmentation in NS-KGC Models

With the aim of maximal utilization of a given rule $\mathtt{l} \in \mathcal{L}$, we first propose two rule augmentation techniques: abduction and rule inversion. The other two techniques prune low-quality rules from $\mathcal{L}$, and independently add new rules to increase coverage. All augmentations are generic and can be integrated with any existing ruleset, and NS-KGC model.

Abduction: The goal of abductive reasoning (or abduction) is to find the best explanation from a given set of observations (Pierce, 1935). It has seen limited use in the context of KBs (Yoshikawa et al., 2019). In our approach, for every rule in $\mathcal{L}$, we introduce several abductive rules with one of the antecedants, appearing as a consequent. As an example, consider the rule:

$$\mathtt{R1}(\mathtt{X}, \mathtt{Y}) \wedge \mathtt{R2}(\mathtt{Y}, \mathtt{Z}) \wedge \mathtt{R3}(\mathtt{Z}, \mathtt{W}) \Rightarrow \mathtt{RH}(\mathtt{X}, \mathtt{W})$$

Our augmentation will generate abductive rules, one for each relation in the body, as:

$$\texttt{R2}(\texttt{Y},\texttt{Z}) \wedge \texttt{R3}(\texttt{Z},\texttt{W}) \wedge \texttt{RH}^{-1}(\texttt{W},\texttt{X}) \Rightarrow \texttt{R1}^{-1}(\texttt{Y},\texttt{X})$$
$$\texttt{R3}(\texttt{Z},\texttt{W}) \wedge \texttt{RH}^{-1}(\texttt{W},\texttt{X}) \wedge \texttt{R1}(\texttt{X},\texttt{Y}) \Rightarrow \texttt{R2}^{-1}(\texttt{Z},\texttt{Y})$$
$$\texttt{RH}^{-1}(\texttt{W},\texttt{X}) \wedge \texttt{R1}(\texttt{X},\texttt{Y}) \wedge \texttt{R2}(\texttt{Y},\texttt{Z}) \Rightarrow \texttt{R3}^{-1}(\texttt{W},\texttt{Z})$$

As an example, let's say a learned rule is $\texttt{BornIn}(\texttt{X},\texttt{U}) \wedge \texttt{PlaceInCountry}(\texttt{U},\texttt{Y}) \Rightarrow \texttt{Nationality}(\texttt{X},\texttt{Y})$. If in the KG, we know that Oprah has nationality U.S., and that she is born in Mississippi, then abduction allows the model to hypothesize that Mississippi might be in U.S. Of course, not all abductions are accurate, for instance, just because Alabama is known to be in U.S., does not mean that Oprah was born in Alabama. Abductive rules increase rule coverage at the cost of precision. We expect the predictor scorer to automatically handle which (abductive) rules can and cannot be trusted.

Rule Inversion: Our second rule augmentation takes an existing rule and rewrites it by referring to inverses of all relations. As an example, if a rule uses the path $\texttt{Oprah} \xrightarrow{\texttt{BornIn}} \texttt{Mississippi} \xrightarrow{\texttt{PlaceInCountry}} \texttt{US}$, then it could also use the equivalent path $\texttt{US} \xrightarrow{\texttt{PlaceInCountry}^{-1}} \texttt{Mississippi} \xrightarrow{\texttt{BornIn}^{-1}} \texttt{Oprah}$. Formally, for every original rule:

$$\texttt{R1}(\texttt{X},\texttt{Y}) \wedge \texttt{R2}(\texttt{Y},\texttt{Z}) \wedge \texttt{R3}(\texttt{Z},\texttt{W}) \Rightarrow \texttt{RH}(\texttt{X},\texttt{W})$$

we add to the ruleset the following inverted rule:

$$\texttt{R3}^{-1}(\texttt{W},\texttt{Z}) \wedge \texttt{R2}^{-1}(\texttt{Z},\texttt{Y}) \wedge \texttt{R1}^{-1}(\texttt{Y},\texttt{X}) \Rightarrow \texttt{RH}^{-1}(\texttt{W},\texttt{X})$$

Rule Filtering: Augmentations increase the size of the ruleset. In order to reduce the number of parameters and the training/test times of the NS-KGC model, we prune seemingly low-quality rules from the augmented rulebase. For this, we compute the PCA score for each original and augmented rule and prune all the rules that have score less than a threshold (set at 0.01 in experiments) and have less than 10 groundings. So, all low-coverage rules with seemingly low quality are pruned out. As experiments show, this results in up to 70% reduction in the number of rules, while preserving KGC performance.

Random Walk Augmentation: Motivated by the empirical success of PCA scores for finding good rules in the previous step, we further augment our ruleset with new, high scoring rules generated independently via local random walks. Starting at each entity in the KG, we perform a number of random walks of fixed length. Each such random walk constitutes the body of the rule and the relation connecting the end entities in the KG form the head of the discovered rule. We score these rules by the PCA score and retain all such rules that have PCA score above the threshold (of 0.1).

4 Experiments

Datasets: We use four datasets for evaluation: WN18RR (Dettmers et al., 2018), FB15K-237 (Toutanova and Chen, 2015), Kinship and UMLS (Kok and Domingos, 2007). For each triple in test set, we answer queries $(\texttt{h},\texttt{r},?)$ and $(\texttt{t},\texttt{r}^{-1},?)$ with answers $\texttt{t}$ and $\texttt{h}$. We report the Mean Reciprocal Rank (MRR) and Hit@k (H@1, H@10) under the filtered measures (Bordes et al., 2013). Details and data stats are in Appendix A.

Baselines: We first experiment with two base models: `RNNLogic+` ([`RNN`] in tables), and `RNNLogic+` with `RotatE` ([`RNN+RotE`]) (Eqn 2). We have reproduced the numbers published by the original authors for these models (details in Appendix D). We run these models with two rulesets: (1) `Orig`, rules generated by RNNLogic (around 300 rules per relation for WN18RR and FB15k-237, and 1000 rules per relation for Kinship and UMLS), and (2) `RW`, only the rules discovered by our random walks. This second setting can only evaluate the value of abduction, inversion, and pruning since random walks are anyways used in generating rules. More details in Appendix C, F and G.

In order to assess the generality of our augmentations, we also experiment with ExpressGNN (Zhang et al., 2020). We choose top five rules for each relation from RNNLogic's `Orig` ruleset according to PCA confidence and provide them as input ruleset to ExpressGNN ([`ExpGNN`] in tables). ExpressGNN does not scale up to the augmented ruleset for FB15K-237, hence we test it for the other three datasets. Refer to Appendix E for more details. We use `AUG` to denote the performance of rule augmentations for all baselines.

We also tried rulesets from NeuralLP (2017), but they are too small to be useful with RNN-Logic+. The only other NS-KGC model that has reported performance similar to RNNLogic+ is RLogic (2022). Unfortunately, their code is not publicly available.[2]

Results: We report the results in Table 1 for the RNNLogic baselines (further details in Appendix

[2]Our reimplementation could not match reported results, and sending several emails to original authors was not helpful.

Table 1: Results of reasoning on four datasets with RNNLogic+ (RNN). Orig represent RNNLogic rules. RotE represents RotatE. AUG represents our proposed augmentations. RW denotes rules discovered by random walks.

Algorithm	WN18RR			FB15K-237			Kinship			UMLS		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
[RNN]-(RW)	44.2	41.6	48.7	26.4	19.8	39.9	63.2	47.8	93.7	74.7	63.1	93.0
[RNN]-(RW+AUG)	47.7	44.3	54.3	29.5	21.5	45.3	65.7	50.9	94.8	79.7	69.5	95.7
[RNN+RotE]-(RW)	48.7	45.1	55.9	30.8	22.8	46.9	71.4	58.0	95.7	82.0	73.5	95.3
[RNN+RotE]-(RW+AUG)	51.1	47.4	58.5	31.4	23.3	47.9	71.9	58.9	96.2	83.8	75.8	96.4
[RNN]-(Orig)	49.6	45.5	57.4	32.9	24.0	50.6	61.6	46.3	91.8	81.4	71.2	95.7
[RNN]-(Orig+AUG)	52.7	48.3	61.3	34.5	25.7	51.9	68.7	54.8	95.7	84.0	75.2	96.4
[RNN+RotE]-(Orig)	51.6	47.4	60.2	34.3	25.6	52.4	68.9	54.9	94.6	81.5	71.2	96.0
[RNN+RotE]-(Orig+AUG)	**55.0**	**51.0**	**63.5**	**35.3**	**26.5**	**52.9**	**72.9**	**59.9**	**96.4**	**84.2**	**76.1**	**96.5**

H). We observe that in all settings, there is a notable increase in performance using augmented rules. In particular, we obtain 7.1 pt and 8.5 pt increase in MRR and Hits@1 in [RNN]-(Orig) setting on Kinship, and 3.5 pt and 5.6 pt increase in MRR and Hits@10 in [RNN]-(RW) setting for WN18RR dataset. We also find that rule augmentations complement RotatE scores in capturing more information about the KG, leading to improved performance in those settings too. To the best of our knowledge, our best results for WN18RR are state-of-the-art for NS-KGC models.

Next, we present the results of our proposed augmentations with ExpressGNN[3] as baseline in Table 2. We note that ExpressGNN assumes the knowledge of test queries while it constructs the MLN during training. Therefore, the results presented in Table 2 are not directly comparable with the results of other models presented in the paper, which do not make this assumption. We observe substantial gains on all datasets and all metrics, notably a 22.4 pt MRR, 17.9 pt Hits@1 and 29.9 pt Hits@10 improvement on WN18RR dataset with our augmentations (AUG). This experiment demonstrates that AUG can help other neuro-symbolic settings as well. Refer to Appendix E for more details.

Table 2: Results of reasoning on three datasets with ExpressGNN (ExpGNN). AUG represents our proposed augmentations[4].

Dataset	Model	MRR	H@1	H@10
WN18RR	[ExpGNN]	52.3	44.1	63.6
	[ExpGNN+AUG]	**74.7**	**62.0**	**93.5**
UMLS	[ExpGNN]	58.1	44.4	77.6
	[ExpGNN+AUG]	**60.9**	**49.2**	**83.4**
Kinship	[ExpGNN]	52.7	41.7	79.8
	[ExpGNN+AUG]	**64.1**	**49.5**	**93.2**

[3] https://github.com/expressGNN/ExpressGNN

[4] Please note that results in this table are not directly comparable to results in Table 1

5 Analysis of Augmented Rules

We perform five further analyses to answer the following questions. **Q1.** Are the rules created by abduction and rule inversion of high quality? **Q2**. What is the individual effect of each type of augmentation on the performance? **Q3**. How do the rule augmentations affect the training time of a model? **Q4**. Can we get the same performance as augmentation by generating more rules from the LSTM in RNNLogic? **Q5**. Are the augmented rules interpretable by a human?

Quality of New Rules: To answer **Q1**, we employ two metrics to assess quality of rules, (PCA-metric and FOIL-metric) before and after abduction and rule inversion. The rules obtained from random walks have high scores by construction since they are filtered based on PCA score. Therefore, they are of high quality as per our definition. (Details in Appendix M and N)

Table 3: Number of high quality rules before and after augmentations on rules generated by RNNLogic.

Rule Set	WN18RR		UMLS	
	FOIL	PCA	FOIL	PCA
Original	2286	2647	25079	28982
Original w/ INV	3157	3577	42188	46908
Original w/ ABD	7141	7607	68693	84554
Original w/ INV + ABD	**8502**	**9155**	**100146**	**125019**

Table 3 presents the number of rules that have a score of at least 0.1 according to each metric, which we regard as criteria for defining a high-quality rule. We observe that there is a large increase in the number of high-quality rules after abduction and rule inversion, nearly tripling in the case of abduction (row 1 vs row 3). This is because the augmented rules exploit the same groundings as the original rules, in the form of new rules. Thus, augmented counterparts of high-quality rules are likely to be high-quality. Overall, we find that abduction and rule inversion does indeed produce high-quality rules.

Ablation: To answer **Q2**, we perform an ablation

Table 4: Ablation study on WN18RR and Kinship for filtering (FIL), inversion (INV), abduction (ABD) and PCA-filtered random walk augmentation (RW).

Algorithm	WN18RR			Kinship		
	MRR	H@1	H@10	MRR	H@1	H@10
AUG	**55.0**	**51.0**	**63.5**	**72.9**	**59.9**	**96.4**
AUG minus ABD	52.2	47.8	61.0	71.3	57.8	96.2
AUG minus INV	54.4	50.0	62.7	71.3	57.7	**96.4**
AUG minus FIL	**55.0**	50.6	63.3	72.5	59.5	**96.4**
AUG minus RW	54.6	50.1	63.2	70.7	57.1	95.6

Table 5: Table showing performance/time trade-off per epoch on two datasets. T/T(min) represents training time per epoch in minutes.

Dataset	Modification	#Rules	T/T	MRR	H@1	H@10
WN18RR	Orig	6135	334	51.6	47.4	60.2
	Orig + AUG	25729	1520	55.0	50.6	63.3
	Orig + AUG + FIL	20053	931	**55.0**	**51.0**	**63.5**
Kinship	Orig	49994	5	68.9	54.9	94.6
	Orig + AUG	315865	36	72.5	59.5	96.4
	Orig + AUG + FIL	97331	11	**72.9**	**59.9**	**96.4**

study for inversion (INV), abduction (ABD), random walk augmentation (RW) and rule filtering (FIL) on [RNN+RotE]-(Orig) setting for WN18RR and Kinship datasets to observe the impact of each type of augmentation. The results are presented in Table 4 (further details are in Appendix I).

In general, abduction (row 3) gives larger improvements than rule inversion (row 2) because as we noticed in the previous section, abduction adds a larger number of high-quality rules to the rule set. We also find that adding the PCA-based random walk rules results in performance improvement, even with only 5% new rules being added (as in Kinship) as compared to original rule set. Finally, we find that filtering based on the PCA metric results in marginal performance improvement, along with lower running times (see below).

Performance vs Training Time Trade-off: To answer **Q3**, we report training time per epoch (in minutes), size of ruleset and performance metrics after augmentation through ABD, INV and RW (denoted as AUG) and filtering (AUG + FIL) with [RNN + RotE] as the baseline model in Table 5.

Our proposed augmentations (INV, ABD and RW) result in substantial performance gains, at the cost of 5-6 times increase in the training time. After filtering (FIL), there is no decrease in performance, and the training time goes down by 2-3× compared to AUG. Therefore, we obtain substantial performance gains through our augmentations, at the cost of only 2-3 times increase in training time.

Rule Generation vs Rule Augmentation: Our augmentations result in 100-200% increase in the number of rules across datasets after filtering. As a control experiment to answer **Q4**, we train RNN-Logic to generate 80 rules per relation (R/R) and augment resulting rules without filtering (for a fair comparison). We further train RNNLogic with 500 rules per relation without augmentation and compare performance of both rulesets (which now have comparable size) using [RNN+RotE] on WN18RR and Kinship in Table 6 (see Appendix J).

Table 6: Performance of augmentation on WN18RR and Kinship. R/R and TR is number of rules per relation and total rules generated from RNNLogic respectively.

Dataset	R/R	TR	AUG	MRR	H@1	H@10
WN18RR	80	9867	Yes	**49.0**	**44.9**	**56.7**
	500	11000	No	47.7	43.7	55.2
Kinship	80	18432	Yes	**69.5**	**56.1**	**94.6**
	500	25000	No	66.1	52.1	93.1

We observe that rule augmentations lead to large improvement over rule generation in all cases, even when rule generation creates more rules. Thus, we find that rule augmentation is more beneficial than simply using more rules from the rule generator. Augmentations exploit a small number of high-quality rules to their full potential.

Qualitative Analysis: To answer **Q5**, we randomly sample 50 rules from the Orig and RW rules for the FB15K-237 dataset and score them as 0 (gibberish), 1 (logically dubious but statistically plausible) and 2 (logically correct) for each ruleset. The reported numbers are averages of scores obtained from two human annotators. We do not include INV and ABD in this comparison as they are generated from Orig rules utilizing the same groundings and thus we expect them to be as interpretable. The scores are 0.90 (Orig) and 1.23 (RW). RW rules are more interpretable due to their high PCA scores. One example of an interpretable rule added by RW is Friends(A, C), Inverse_Producer(C, D), Writer(D , B) :- Friends(A, B). We provide additional rule examples for each type of augmentation in Appendix K.

6 Conclusion and Future Work

We present simple rule augmentation techniques in the context of Neuro-Symbolic Knowledge Graph models and obtain substantial increase in performance over strong base models. We believe our augmentations can become standard for all subsequent NS-KGC models. We release code and rulesets for further research. Future work includes using our augmentation technique during the iterative learning of rules in algorithms such as RNNLogic, potentially further improving their performance.

Acknowledgements

This work is supported by grants by Google, IBM, Verisk, and 1MG, and the Jai Gupta chair fellowship by IIT Delhi. We also acknowledge travel support from Google travel grant. We thank the IIT Delhi HPC facility for its computational resources.

Limitations

Since rule abduction and inversion utilize the same groundings as the original rules, Neuro-Symbolic KGC models that are based on grounding the entire rule will not benefit from these augmentations. Abduction and inversion also require the model to be trained on a knowledge graph that contains the inverse relations $\mathtt{r}^{-1}$ for each relation $\mathtt{r}$. Finally, since RNNLogic+ has a separate rule embedding for each rule, performing rule augmentation increases the number of parameters in the model and leads to longer training times and larger GPU memory consumption.

Ethics Statement

We anticipate no substantial ethical issues arising due to our work on rule augmentation for Neuro-Symbolic KGC. Our work relies on a set of rules generated from another source to perform augmentation. This may result in the augmented rule set exaggerating the effect of malicious or biased rules in the original rule set.

Acknowledgements

This work is supported by IBM AI Horizons Network grant, grants by Google, Verisk, and 1MG, an IBM SUR award, and the Jai Gupta chair fellowship by IIT Delhi. We acknowledge travel support by Google and Yardi School of AI travel grants. We thank the IIT Delhi HPC facility for its computational resources.

References

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *NeurIPS*. Curran Associates, Inc.

Kewei Cheng, Jiahao Liu, Wei Wang, and Yizhou Sun. 2022. RLogic: Recursive Logical Rule Learning from Knowledge Graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 179–189, New York, NY, USA. Association for Computing Machinery.

Kewei Cheng, Ziqing Yang, Ming Zhang, and Yizhou Sun. 2021. UniKER: A Unified Framework for Combining Embedding and Definite Horn Rule Reasoning for Knowledge Graph Inference. In *EMNLP*, pages 9753–9771, Online and Punta Cana, Dominican Republic. ACL.

Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. 2020. Principal Neighborhood Aggregator for Graph Nets. In *NeuRIPS*, pages 13260–13271.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. 2018. Go for a Walk and Arrive at the Answer: Reasoning Over Paths in Knowledge Bases using Reinforcement Learning. In *ICLR*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. 2013. AMIE: Association Rule Mining under Incomplete Evidence in Ontological Knowledge Bases. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13, page 413–422, New York, NY, USA. Association for Computing Machinery.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2018. Knowledge Graph Embedding with Iterative Guidance from Soft Rules. In *AAAI*, pages 4816–4823.

Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR '17.

Stanley Kok and Pedro Domingos. 2007. Statistical Predicate Invention. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, page 433–440, New York, NY, USA. Association for Computing Machinery.

C. S. Pierce. 1935. *The Collected Papers of Charles Sanders Peirce*. Harvard University Press, Harvard, US.

Meng Qu, Junkun Chen, Louis-Pascal A. C. Xhonneux, Yoshua Bengio, and Jian Tang. 2021. RNNLogic: Learning Logic Rules for Reasoning on Knowledge Graphs. In *ICLR*, pages 1–21.

J. R. Quinlan. 1990. Learning Logical Definitions from Relations. *Machine Learning*, 5(3):239–266.

Matthew Richardson and Pedro Domingos. 2006. Markov logic networks. *Machine Learning*, 62(1–2):107–136.

Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. 2019. DRUM: End-To-End Differentiable Rule Mining On Knowledge Graphs. In *NeuRIPS*, volume 32. Curran Associates, Inc.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In *ICLR*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus Latent Features for Knowledge Base and Text Inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, Beijing, China. Association for Computational Linguistics.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743.

Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In *EMNLP*, pages 564–573, Copenhagen, Denmark. ACL.

Fan Yang, Zhilin Yang, and William W Cohen. 2017. Differentiable Learning of Logical Rules for Knowledge Base Reasoning. In *NeuRIPS*, volume 30. Curran Associates, Inc.

Masashi Yoshikawa, Koji Mineshima, Hiroshi Noji, and Daisuke Bekki. 2019. Combining Axiom Injection and Knowledge Base Completion for Efficient Natural Language Inference. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 7410–7417. AAAI Press.

Yuyu Zhang, Xinshi Chen, Yuan Yang, Arun Ramamurthy, Bo Li, Yuan Qi, and Le Song. 2020. Efficient Probabilistic Logic Reasoning with Graph Neural Networks. In *ICLR*.

A Data Statistics and Evaluation Metrics

Table 7 summarizes the statistics of the data used in the experiments of our work. We utilize the standard train, validation and test splits for WN18RR and FB15k-237 datasets. Since there are no standard splits for UMLS and Kinship datasets, for consistency, we employ the splits used by RNN-Logic (2021) for evaluation (created by randomly sampling 30% triplets for training, 20% for validation and the rest 50% for testing).

Metrics: For each triplet $(\mathtt{h}, \mathtt{r}, \mathtt{t})$ in the test set, traditionally queries of the form $(\mathtt{h}, \mathtt{r}, ?)$ and $(?, \mathtt{r}, \mathtt{t})$ are created for evaluation, with answers $\mathtt{t}$ and $\mathtt{h}$ respectively. We model the $(?, \mathtt{r}, \mathtt{t})$ query as $(\mathtt{t}, \mathtt{r}^{-1}, ?)$ with the same answer $\mathtt{h}$, where $\mathtt{r}^{-1}$ is the inverse relation for $\mathtt{r}$. In order to train the model over the inverse relations, we augment the training data with an additional $(\mathtt{t}, \mathtt{r}^{-1}, \mathtt{h})$ triple for every triple $(\mathtt{h}, \mathtt{r}, \mathtt{t})$ present in KG.

Given ranks for all queries, we report the Mean Reciprocal Rank (MRR) and Hit@k (H@k, k = 1, 10) under the filtered setting in the main paper and two additional metrics: Mean Rank (MR) and Hits@3 in the appendices. MRR and Hits@k metrics are reported after multiplying with 100. To maintain consistency with RNNLogic, in cases where the model assigns same probability to other entities along with the answer, we compute the rank as $(m + \frac{(n+1)}{2})$ where m is the number of entities with higher probabilities than the correct answer and n is the number of entities with same probability as the answer.

B RotatE

`RotatE` is a knowledge graph embedding model that embeds entities and relations in complex space. Relation embeddings are modeled as rotations in complex vector space. Formally, $\mathtt{RotatE}(\mathtt{h}, \mathtt{r}, \mathtt{t})$ is calculated using the following equation:

$$\mathtt{RotatE}\,(\mathtt{h}, \mathtt{r}, \mathtt{t}) = -\mathtt{d}(\mathbf{x}_\mathtt{h} \circ \mathbf{x}_\mathtt{r}, \mathbf{x}_\mathtt{t}) \qquad (3)$$

where $\mathtt{d}$ is the cosine distance in complex vector space, RotatE embedding of $\mathtt{r}$ is $\mathbf{x}_\mathtt{r}$, and $\circ$ is the Hadamard product. Intuitively, we rotate $\mathbf{x}_h$ by the rotation defined by $\mathbf{x}_r$ and consider the distance between the result and $\mathbf{x}_t$. For our experiments, `RotatE` is trained separately and the trained embeddings are used to calculate scores for the [`RNN` + `RotE`] baseline.

C Experimental Setup for RNNLogic

In order to obtain main results in Table 1, we train the rule generator in RNNLogic with optimal hyperparameters obtained after communication with the original authors and generate a set of high-quality Horn rules to use for training RNNLogic+. For our best results, we utilize optimal rules provided by the authors of RNNLogic[5]. We augment these rules by abduction (`ABD`), and then rule inversion (`INV`) on both the original rules and the rules formed after abduction. We further augment the rulebase with the rules discovered by random walks (`RW`). Finally, we filter (`FIL`) superior rules from these rules by

[5] https://github.com/DeepGraphLearning/RNNLogic

Table 7: Statistics of Knowledge Graph datasets

Datasets	#Entities	#Relations	#Training	#Validation	#Test
FB15K-237	14541	237	272,115	17,535	20,446
WN18RR	40,943	11	86,835	3,034	3,134
Kinship	104	25	3,206	2,137	5,343
UMLS	135	46	1,959	1,306	3,264

Table 8: RNNLogic rules used per dataset. `INV` and `ABD`, `RW` represent rule inversion and abduction and PCA-based walk rule augmentation respectively. The last column represents the rule filtering (`FIL`) applied on all the rules.

Datasets	#Rules	#Rules + INV	#Rules + ABD	#Rules + INV + ABD	#Rules +INV + ABD + RW	#Rules +INV+ ABD + RW + FIL
FB15K-237	126137	174658	295403	392280	394967	298446
WN18RR	6135	8742	18251	23304	25729	20053
Kinship	49994	91544	171302	301646	315865	97331
UMLS	91908	171526	322464	564374	574687	204504

Table 9: Results of reasoning on four datasets: WN18RR, FB15K-237, Kinship and UMLS with RNNLogic+ (`RNN`). `Orig` represents rules acquired from RNNLogic. `RotE` represents RotatE. `AUG` represents all the proposed approaches in our work. `RW` represents rules obtained only from PCA-filtered random walk augmentation.

Algorithm	**WN18RR**					**FB15K-237**				
	MR	MRR	H@1	H@3	H@10	MR	MRR	H@1	H@3	H@10
[RNN]-(RW)	8218.73	44.2	41.6	45.5	48.7	808.32	26.4	19.8	28.9	39.9
[RNN]-(RW+AUG)	7241.14	47.7	44.3	49.2	54.3	481.58	29.5	21.5	32.3	45.3
[RNN+RotE]-(RW)	4679.70	48.7	45.1	49.8	55.9	521.06	30.8	22.8	33.5	46.9
[RNN+RotE]-(RW+AUG)	4431.75	51.1	47.4	52.6	58.5	279.65	31.4	23.3	34.3	47.9
[RNN]-(Orig)	5857.65	49.6	45.5	51.4	57.4	256.14	32.9	24.0	36.1	50.6
[RNN]-(Orig+AUG)	5156.38	52.7	48.3	54.9	61.3	218.11	34.5	25.7	37.9	51.9
[RNN+RotE]-(Orig)	4445.79	51.6	47.4	53.4	60.2	217.30	34.3	25.6	37.5	52.4
[RNN+RotE]-(Orig+AUG)	**4231.77**	**55.0**	**51.0**	**57.2**	**63.5**	**198.81**	**35.3**	**26.5**	**38.7**	**52.9**
Algorithm	**Kinship**					**UMLS**				
	MR	MRR	H@1	H@3	H@10	MR	MRR	H@1	H@3	H@10
[RNN]-(RW)	3.6	63.2	47.8	73.5	93.7	5.17	74.7	63.1	83.6	93.0
[RNN]-(RW+AUG)	3.36	65.7	50.9	75.8	94.8	3.65	79.7	69.5	87.8	95.7
[RNN+RotE]-(RW)	2.99	71.4	58.0	81.6	95.7	3.46	82.0	73.5	88.9	95.3
[RNN+RotE]-(RW+AUG)	2.89	71.9	58.9	81.7	96.2	3.20	83.8	75.8	90.0	96.4
[RNN]-(Orig)	4.45	61.6	46.3	71.7	91.8	3.66	81.4	71.2	90.3	95.7
[RNN]-(Orig+AUG)	3.15	68.7	54.8	78.9	95.7	**2.81**	84.0	75.2	**91.5**	96.4
[RNN+RotE]-(Orig)	3.28	68.9	54.9	78.8	94.6	3.17	81.5	71.2	90.1	96.0
[RNN+RotE]-(Orig+AUG)	**2.80**	**72.9**	**59.9**	**82.6**	**96.4**	2.83	**84.2**	**76.1**	91.3	**96.5**

Table 10: Ablation study performed on Kinship and UMLS for filtering (`FIL`), inversion (`INV`), abduction (`ABD`) and random walk augmentation (`RW`). `AUG` represents all proposed approaches in our work taken together.

Algorithm	**Kinship**					**UMLS**				
	MR	MRR	H@1	H@3	H@10	MR	MRR	H@1	H@3	H@10
AUG	**2.80**	**72.9**	**59.9**	**82.6**	**96.4**	**2.83**	**84.2**	**76.1**	**91.3**	**96.5**
AUG minus ABD	2.90	71.3	57.8	81.4	96.2	3.16	82.6	72.9	90.8	96.5
AUG minus INV	2.89	71.3	57.7	81.5	96.4	2.98	83.8	74.8	91.9	96.5
AUG minus FIL	2.84	72.5	59.5	82.3	96.4	3.01	83.9	75.1	91.5	96.5
AUG minus RW	2.99	70.7	57.1	80.8	95.6	3.05	82.8	73.2	91.1	96.5

Table 11: Ablation study performed on WN18RR for abduction (`ABD`), inversion (`INV`), filtering (`FIL`) and PCA-based random walk augmentation (`RW`). `AUG` represents represents all the approaches proposed in our work.

Algorithm	**WN18RR**				
	MR	MRR	H@1	H@3	H@10
AUG	4231.77	**55.0**	**51.0**	**57.2**	**63.5**
AUG minus ABD	4406.95	52.2	47.8	54.1	61.0
AUG minus INV	4302.04	54.4	50.0	56.8	62.7
AUG minus FIL	**4224.20**	**55.0**	50.6	57.1	63.3
AUG minus RW	4263.43	54.6	50.1	57.0	63.2

PCA score. We present statistics detailing the number of rules used per dataset after each augmentation step in Table 8. These rules are utilized in RNNogic+ ([RNN]-(Orig)) and RNNLogic+ with RotatE ([RNN+RotE]-(Orig)) baselines. For the other results: [RNN]-(RW) and [RNN+RotE]-(RW), we employ only the rules obtained by RW augmentation and train RNNLogic+ model with them (Appendix F). The goal of these set of results is to test the utility of abduction and rule inversion with a different set of rules. The details of training RNNLogic+ model is provided in Appendix G.

D RNNLogic Results Reproduction

We have reproduced the results of RNNLogic+ with and without RotatE and obtained similar results to the original RNNLogic paper (Qu et al., 2021), however the numbers reported in this paper for [RNN] and [RNN + RotE] are our own reproductions. In this section, we report a comparison between the original results and our reproduced results for RNNLogic+ model ([RNN]) on the WN18RR and FB15K-237 datasets. As can be observed in the Table 12, our reproduced results are better than the published results of RNNLogic model for both the datasets, using hyperparameters obtained after communication with the authors of the RNNLogic paper.

Table 12: Comparison of the results reported in original RNNLogic paper with the results reproduced by the authors of this paper.

Dataset	Numbers	MR	MRR	H@1	H@3	H@10
WN18RR	Reported	7204	48.9	45.3	50.6	56.3
	Reproduced	**5858**	**49.6**	**45.5**	**51.4**	**57.4**
FB15K-237	Reported	480	29.9	21.5	32.8	46.4
	Reproduced	**256**	**32.9**	**24.0**	**36.1**	**50.6**

E ExpressGNN Training and Hyperparameter Setting

As already discussed in Section 4, in order to prove the broad applicability of proposed augmentations (AUG) in our work, we perform experiments with ExpressGNN model as another baseline in Table 2. In this section we provide the details of this experiment. The current implementation of ExpressGNN model scales poorly with the number of rules, necessitating the use of a much smaller ruleset size. We generate ruleset for each dataset by selecting the top $5 - 10$ rules per relation (in the rule head) from RNNLogic rules for that dataset (ORIG) based on the PCA score. This results in 417 rules for WN18RR, 500 rules for Kinship and 460 rules for UMLS. We perform augmentations on these rules and further maintain a threshold of the PCA score to be 0.95 while filtering RW rules. After augmentation, we obtain 1734 rules for Kinship, 2058 rules for UMLS and 828 rules for WN18RR. We also augment the training and the test set of ExpressGNN datasets with the inverse triples $(\mathtt{t}, \mathtt{r}^{-1}, \mathtt{h})$ for each original $(\mathtt{h}, \mathtt{r}, \mathtt{t})$ triple. Hyperparameters used for training are the optimal ones from the original paper. Results for FB15k-237 are omitted since ExpressGNN does not scale up to the augmented ruleset.

ExpressGNN assumes the knowledge of test queries at training time to construct its Markov Logic Network. For the test triple $(\mathtt{h}, \mathtt{r}, \mathtt{t})$, this informs the model that h is a potential head and t is a potential tail entity for given relation r, even though this information might not be present in the training data. Hence, results presented in Table 2 are not directly comparable to results in Table 1.

F Rule Generation via Random Walks

Because rules generated by employing random walks form a distinct ruleset in the main paper ([RNN]-(RW)), we explain the statistics of these rules in detail in a dedicated section here. In order to determine the number of rules generated from the random walks, we calculate the difference of the column '**#Rules** + INV + ABD' and '**#Rules** + INV + ABD + RW' in the Table 8 and summarize the resulting statistics of the number of RW rules created for each dataset in the Table 13. When compared to Table 8, we note that although random walk rules (RW) comprise less than 8% of the augmented ruleset for all the datasets, these rules are still pivotal. This is because we notice a considerable decrease in performance after removing these rules as observed in Table 4, Table 10 and Table 11.

Table 13: Number of random walk rules (RW) generated per dataset in the experiments

Dataset	FB15K-237	WN18RR	Kinship	UMLS
#RW Rules	2687	2425	14219	10313

G RNNLogic+ Training and Hyperparameter Setting

Here we describe the training of RNNLogic+ model that is utilized in Table 1 and complementary Table 9. We use the same methodology for training RNNLogic+ model as in the original work (Qu et al., 2021). New rule embeddings are created for

all the rules that are added to the rule set after rule augmentation. Rule embedding dimension is set to 16 (compared to 32 in original RNNLogic+) across datasets to mitigate the effect of the increased number of parameters in the model due to new rule embeddings. Results reported are for a single run with fixed seed over 5 epochs of training.

The hyperparameter η in Equation (2) representing the relative weight is set to 0.01, 0.05, 0.1 and 0.5 for WN18RR, FB15k-237, UMLS and Kinship respectively. The RotatE embedding dimension is set to 200, 500, 1000 and 2000 for WN18RR, FB15k-237, UMLS and Kinship respectively. We keep a consistent batch size of 8, 4, 32 and 16 for WN18RR, FB15k-237, UMLS and Kinship respectively. The number of parameters for RNNLogic+ scales with the rule embedding size and the number of rules, reaching a maximum of 16*298446 = 4775136 for FB15k-237 after augmentations and filtering (leading to a training time of around 23 hours). As we can see, augmentation adds new rules leading to increase in the parameters of the model. All training was carried out on a single Tesla V100 GPU. The optimal values of all the hyper-parameters was found by tuning on validation set on each dataset.

H Detailed Results on Proposed Augmentations

Results in Table 9 are supplementary to results already presented in Table 1. In addition to MRR, Hits@1 and Hits@10 presented in the Table 1 in the Experiment section, we also present Mean Rank (MR) and Hits@3 here. As discussed already in Section 4, `AUG` includes abduction (`ABD`), inversion (`INV`), rule filtering (`FIL`) and random walk augmentation (`RW`).

In Table 9, we observe that there is a consistent improvement in the performance of the model for all the metrics after rule augmentation and filtering (`AUG`). Notably, for the two new metrics introduced in Table 9, we obtain a performance gain of 3.7 point on Hits@3 and 40.4% on MR for FB15K-237 dataset and [`RNN`]-(`RW`) baseline. Since the original rules for the random walk baseline are lesser in number, [`RNN`]-(`RW`) and [`RNN + RotE`] - (`RW`) benefit more from augmentation. We also observe that for Kinship and UMLS, [`RNN + RotE`] - (`RW`) gives better performance than [`RNN + RotE`] - (`Orig`), highlighting the quality of the rules discovered by local random walks followed by PCA filtering.

I Detailed Results of Ablation Study

Results in Table 10 are supplementary to results already presented in Table 4. Besides the three metrics presented in Table 4, we present Hits@3 and MR in this table. Additionally, we also demonstrate results of ablation on WN18RR dataset in Table 11. Ablation is not performed on FB15K-237 due to computational constraints. As with the other metrics, Hits@3 and MR is the most affected by abductive rules in UMLS and WN18RR because abduction results in augmenting the ruleset with a large number of high-quality rules (see Table 3). Furthermore, Hits@3 and MR gets most affected by PCA-based random walk augmentation in Kinship dataset. This is because Kinship is a dense dataset, and a large number of high-quality rules are quickly discovered by the random walks.

J Detailed Results of Rule Generation vs Rule Augmentation

Results in Table 14 are supplementary to the results already presented in Table 6. Here we present Hits@3 and MR as two additional metrics for analyzing the need for rule augmentation.

We generate rules by training RNNLogic model. We consider 80 rules per relation for each dataset from these rules and expand them by performing three augmentations and filtering. This results in total of 9867 rules for WN18RR and 18432 rules for Kinship data. Then, we train RNNLogic+ with RotatE ([`RNN+RotE`]) on these rules and compare the results with RNNLogic+ with RotatE model trained on 500 rules per relation without augmentations. We observe that model trained with augmented rules consistently performs better than model trained by merely increasing the number of rules generated, even for a comparable number of rules. Specifically, we observe that model trained with augmented rules shows 4 point Hit@1 gain in Kinship dataset over the model trained with merely increased rules. These results strengthens the hypothesis that it is more helpful to leverage a few high-quality augmented rules rather than exploiting a plethora of lower-quality rules for Neuro-Symbolic KG Completion.

K Qualitative Analysis of the Augmented Rules

In this section, we present one logical rule generated after each augmentation step as examples. The rules are taken from the FB15K-237 dataset.

Table 14: Comparison of performance by rule augmentation with performance on the original rules on WN18RR and Kinship. R/R and TR is number of rules per relation and total rules generated from RNNLogic respectively. ABD represents abduction performed on original rules.

Dataset	R/R	TR	ABD	MR	MRR	Hits@1	Hits@3	Hits@10
WN18RR	80	9867	Yes	**4701.61**	**49.0**	**44.9**	**50.5**	**56.7**
	500	11000	No	4848.39	47.7	43.7	49.8	55.2
Kinship	80	18432	Yes	**3.21**	**69.5**	**56.1**	**79.4**	**94.6**
	500	25000	No	3.62	66.1	52.1	75.3	93.1

1. **ABD**: $\texttt{LivesIn}(\texttt{PersonA}, \texttt{LocationB}) : -\ \texttt{PlayFor}(\texttt{PersonA}, \texttt{TeamC}), \texttt{Inverse\_Team\_Location}(\texttt{TeamsC}, \texttt{LocationB})$

2. **INV**: $\texttt{Inverse\_Person\_Language}(\texttt{LanguageA}, \texttt{PersonB}) : -\ \texttt{Inverse\_Film's\_Language}(\texttt{LanguageA}, \texttt{FilmC}), \texttt{StoryWrittenBy}(\texttt{FilmC}, \texttt{PersonB})$

3. **RW**: $\texttt{Friends}(\texttt{PersonA}, \texttt{PersonB}) : -\ \texttt{Friends}(\texttt{PersonA}, \texttt{PersonC}), \texttt{Inverse\_Producer}(\texttt{PersonC}, \texttt{FilmD}), \texttt{Writer}(\texttt{FilmD}, \texttt{PersonB})$

For example, the rule in **ABD** category states that a person will live in the same city as the team he plays for is located. Therefore, we conclude that the rules captured through augmentations can be human interpretable.

L An Alternative Augmentation Strategy

Recall that in our proposed methodology (Orig+AUG) in Section 3, we consider original rules (ORIG) and perform abduction (ABD) on the original rules. This is followed by rule inversion (INV) over the original rules and abductive rules. Then, we introduce the random walk rules (RW) as the final augmentation step in the proposed augmentations (AUG) for the original (Orig) ruleset. In this section, we consider an alternative sequence of augmenting the ruleset where we consider both the original (Orig) and the random walk rules (RW) and apply abduction and rule inversion on both of them. We denote this setting as $(\texttt{Orig} + \texttt{AUG2})$. We report a comparison of $(\texttt{Orig} + \texttt{AUG2})$ with $(\texttt{Orig} + \texttt{AUG})$ (Table 1) with $[\texttt{RNN} + \texttt{RotE}]$ as the baseline model in Table 15. From the results in the table, we conclude that Orig + AUG2 does not result in improvement over our original methodology of Orig + AUG. It also creates a larger ruleset, further slowing down the training of the model.

Table 15: Comparison of performance by exploring two methodologies of augmentations: $(\texttt{Orig} + \texttt{AUG})$ and $(\texttt{Orig} + \texttt{AUG2})$.

Dataset	Augmentation	MRR	H@1	H@10
WN18RR	Orig + AUG	**55.0**	**51.0**	**63.5**
	Orig + AUG2	54.4	50.2	62.9
Kinship	Orig + AUG	**72.9**	**59.9**	**96.4**
	Orig + AUG2	71.1	58	95.8

M PCA-Confidence Metric

In this section, we explain in detail, the PCA-confidence metric that has been employed to score the rules discovered through random walk in our third augmentation approach. This metric has also been used to score the augmented rules in Table 3.

PCA: The calculation of the metric utilizes a Partial Closed World assumption (Galárraga et al., 2013) and assumes that if we know one $\texttt{t}$ for a given $\texttt{r}$ and $\texttt{h}$ in $\texttt{r}(\texttt{h}, \texttt{t})$, then we know all $\texttt{t}'$ for that $\texttt{h}$ and $\texttt{r}$. Let the rules under consideration be of the form $\texttt{B} \Rightarrow \texttt{r}(\texttt{h}, \texttt{t})$. Then the PCA-score $\texttt{PCAConf}(\texttt{B} \Rightarrow \texttt{r})$ is:

$$\frac{\#(\texttt{h}, \texttt{t}) : |\texttt{Path}(\texttt{h}, \texttt{B}, \texttt{t})| > 0 \wedge \texttt{r}(\texttt{h}, \texttt{t}) \in \texttt{P}}{\#(\texttt{h}, \texttt{t}) : |\texttt{Path}(\texttt{h}, \texttt{B}, \texttt{t})| > 0 \wedge \exists \texttt{t}' : \texttt{r}(\texttt{h}, \texttt{t}') \in \texttt{P}}$$

Essentially, it is the number of positive examples, $\texttt{P}$, satisfied by the rule divided by the total number of $(\texttt{h}, \texttt{t})$ satisfied by the rule such that $\texttt{r}(\texttt{h}, \texttt{t}')$ is a positive example for some $\texttt{t}'$.

N FOIL-Score Metric

We employ a modification of FOIL as one of the evaluation metrics to assess the quality of rules produced by augmentation techniques (**Q1**) in Table 3. FOIL-scoring metric is discussed in detail below.
FOIL: Let the rules be of the form $\texttt{B} \Rightarrow \texttt{r}(\texttt{h}, \texttt{t})$. Let $\texttt{Path}(\texttt{h}, \texttt{B}, \texttt{t})$ be the set of paths from $\texttt{h}$ to $\texttt{t}$ that act as groundings for the rule body $\texttt{B}$. Under the Closed World assumption, we assume that all triples not in the training and test set are false. Inspired by the First-Order Inductive Learner algorithm (Quinlan, 1990), we define FOIL score to

assess the quality of a rule as follows:

$$\texttt{FOIL}(\texttt{B} \Rightarrow \texttt{r}) = \frac{\sum_{\texttt{r(h,t)} \in \texttt{P}} |\texttt{Path}(\texttt{h}, \texttt{B}, \texttt{t})|}{\sum_{\texttt{(h,t)}} |\texttt{Path}(\texttt{h}, \texttt{B}, \texttt{t})|}$$

In the above equation, P represents the set of positive examples in the given KG. The key difference between the FOIL score proposed originally (Quinlan, 1990) and ours is that instead of considering the number of examples satisfied by the rule, we calculate the number of groundings of the rule. This is more in line with the score calculated by RNNLogic+, which considers the number of groundings as well. Ideally the rules should have larger number of groundings for positive triples in comparison to the other triples.

Typically, negative sampling is used to calculate these metrics (PCA in Appendix M and FOIL here) as it is computationally expensive to compute exhaustive negative examples. However, we calculate these metrics by considering the entire knowledge graph, which is enabled by utilizing batching and sparse matrix operations on the adjacency graph.

We highlight that we are the first to show the utility of PCA Confidence and FOIL in the context of neuro-symbolic models. This makes our specific approach distinct from AMIE (Galárraga et al., 2013) and FOIL (Quinlan, 1990), and more targeted to our setting due to the changes in the method of computation.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section or justification Section Limitations, Page number 6

☑ A2. Did you discuss any potential risks of your work?
Section Ethics Statement, Page number 6

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section Abstract and 1 (Introduction), Page number 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Left blank.

☑ B1. Did you cite the creators of artifacts you used?
Section References, Page number 6 and Appendix C, Page number 7

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
We have used publicly available code and rule files released by the authors of RNNLogic on Github, which has not been explicitly licensed. We have mentioned source of rule files in Appendix C, Page number 7.

☒ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
The authors of the code for RNNLogic and ExpressGNN have not explicitly stated their intended use of code on Github. They only require potential users to cite their paper if they use the code, which we have done.

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
All the datasets that we have used in our experiments are standard datasets and we have cited the creators for each one of them.

☒ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
We have not created any new artifacts through our work. We have provided original and augmented rule sets used in our experiments in the submitted code.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Appendix A, Page number 7

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

C ☑ Did you run computational experiments?

Left blank.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix G, Page number 10

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix G, Page number 10

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Appendix G, Page number 10

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix D, Page number 9

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.