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

Ananjan Nandi **Navdeep Kaur** **Parag Singla** **Mausam**
Indian Institute of Technology
New Delhi, India
{cs1190325, navdeepk.cstaff, parags, mausam}@cse.iitd.ac.in

Abstract

High-quality rules are imperative to the success of Neuro-Symbolic Knowledge Graph (KG) models because they equip the models with the desired qualities of interpretability and generalizability. Though necessary, learning logic rules is an inherently hard task. The past rule learning models in KGs have either struggled to learn rules at the scale required for large KGs or the rules learnt by such models are of poor-quality and not diverse. To address this limitation, we propose two novel ways of feeding more high-quality rules to Neuro-Symbolic Knowledge Graph reasoning models. Specifically, we augment the existing set of logic rules of a KG model by exploiting the concepts of abduction and rule inversion. Our proposed approach is generic and is applicable to most neuro-symbolic KG models in literature. Experiments on three datasets prove that our proposed approach obtains upto 23% MR and 7% MRR gains over the base models.

1 Introduction

Knowledge Graphs (KG) (Bollacker et al., 2008; Miller, 1995) comprise of everyday facts that are critical for solving downstream NLP tasks such as question answering (Sun et al., 2022), relation extraction (Wang et al., 2014) etc. Due to ever-increasing size of knowledge graphs, they are unavoidably incomplete. The most dominant methodology to solve the task of knowledge graph completion (KGC) in the past decade has been based on learning knowledge graph embeddings (KGE) (Bordes et al., 2013; Trouillon et al., 2016; Sun et al., 2019) where entities and relations are represented as learnable vectors in a high-dimensional vector space. These techniques have shown remarkable success with state-of-the-art performance.

Very recently, the research focus has shifted towards designing advanced Neuro-Symbolic (NS) models (Zhang et al., 2020; Qu et al., 2021) for knowledge graph completion where KG embeddings are enhanced by learning first-order logic rules along with them. The strength of these models is that they bring together best of both the worlds - generalizability and interpretability of first-order logic rules with the scalability and superior performance of KG embeddings.

Though quite effective, developing such models present challenges. One of the major roadblocks in developing such models is the rule learning required for them at large scale. For example, past end-to-end differentiable models such as NeuralLP (Yang et al., 2017), DRUM (Sadeghian et al., 2019) search their potential rules in exponentially large space. Although, a more recent model RNN-Logic (Qu et al., 2021) reduces its search complexity by developing a model that has independent rule generator and rule scorer, however, empirically the rules generated by this model are not high-quality.

Another line of work includes RLogic (Cheng et al., 2022) and Path-RNN (Neelakantan et al., 2015), which relies on random walks for structure learning, capturing only the local structure in the neighborhood of a given node. Finally, some models such as RUGE (Guo et al., 2018), Express-GNN (Zhang et al., 2020) avoid the burden of learning rules altogether and depend upon external ILP models (Muggleton and de Raedt, 1994) for their rules. To conclude, the past neuro-symbolic KGC models have either learnt rules in huge search space which is challenging or the quality of the rules produced by these models have not been good enough.

Because high-quality rules are hard to generate and hold vital knowledge about KG in them, the objective of this work is to extract more knowledge from these rules and assimilate it into supplementary rules that our approach produces from these original rules. These supplementary rules are produced by applying two approaches on each existing rule: (i) abduction: it aims at inferring cause from effect. It also helps in determining the smallest set of assumptions required to be made in the cause in order to get the required effect. We build on this and design one supplementary rule for each assumption made by abductive reasoning on the original rule (ii) rule inversion: for each existing rule in KG, we obtain a supplementary rule by inverting each relation in the rule.

Our proposed approach is generic and can be integrated with most of the existing NS-KG models. Specifically, we employ it on RNNLogic (2021) which is a state-of-the-art NS-KG model. Experiments over three datasets show that models trained with augmented rules outperform base models by significant margins, obtaining comparable results to the current state-of-the-art in several cases.

2 Background and Related Work

We are given an incomplete KB $\mathcal{K} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$ consisting of set $\mathcal{E}$ of entities, set $\mathcal{R}$ of relations 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 KB. We consider the setting of neuro-symbolic KB embeddings in our work where a combination of logical rules and knowledge base embeddings is utilized to reason about the validity of a triple. In order to explain our proposed method, we consider RNNLogic (Qu et al., 2021) as our base model.

2.1 RNNLogic+

RNNLogic+ model utilized in our experiments builds upon the RNNLogic (2021) model which produces a set of high-quality rules, which are further used by the RNNLogic+ model to compute the score of a given triple. The scoring function of RNNLogic+ model is described below.

RNNLogic+: Given an incomplete triple $(\mathtt{h}, \mathtt{r}, ?)$ and a set of rules $\mathcal{L}$ learnt by RNNLogic, candidate answer $\mathtt{o}$ is scored by RNNLogic+ as follows:

$$\mathtt{score}(\mathtt{o}) = \mathtt{MLP}\big(\mathtt{PNA}(\{\mathbf{v}_{\mathtt{l}} \mid \#(\mathtt{h}, \mathtt{l}, \mathtt{o})\}_{\mathtt{l} \in \mathcal{L}})\big) \tag{1}$$

where the learnable embedding $\mathbf{v}_{\mathtt{l}}$ of a given rule $\mathtt{l} \in \mathcal{L}$ is weighed by the number of times (#) a triple $(\mathtt{h}, \mathtt{r}, \mathtt{o})$ satisfies the rule $\mathtt{l}$'s body. The resulting weighted embeddings of all the rules are aggregated by employing PNA aggregator (Corso et al., 2020) and this aggregated embedding is passed through an MLP in order to obtain the final scalar score of a triple as in equation (1).

The authors further designed another scoring function that incorporates RotatE into equation (1) where the goal is to exploit the knowledge encoded in the KG embeddings. The new scoring function is given as follows:

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

where $\mathtt{score}(\mathtt{o})$ is the score in equation (1), $\mathtt{RotatE}\,(\mathtt{h}, \mathtt{r}, \mathtt{o})$ is the score of the triple obtained from RotatE (Sun et al., 2019) model, and η is hyper-parameter of the model.

2.2 Related Work

The past work on neuro-symbolic knowledge graph completion can roughly be characterized along four axes. First set of models like RNNLogic (Qu et al., 2021), ExpressGNN (Zhang et al., 2020) exploit variational inference to assess the plausibility of a given triple. Another line of work that includes Minerva (Das et al., 2018), DeepPath (Xiong et al., 2017) explore reinforcement learning that trains an agent to find reasoning paths for KG completion. Third popular approach is to exploit attention over relations in a given KG in order to learn end-to-end differentiable models (Yang et al., 2017; Sadeghian et al., 2019).

The final set of work which includes models like UNIKER (Cheng et al., 2021), RUGE (Guo et al., 2018) integrate the traditional rules learnt via ILP models with KG embeddings models such that the two type of models enhance each other's performance. Our approach is generic and exploits the rules learnt by most of the aforementioned models in order to enhance them further through abduction and rule inversion approaches.

3 Abduction and Rule Inversion in Neuro-Symbolic Knowledge Graph Models

With the aim of harnessing maximal knowledge out of a given high-quality rule, we propose two rule augmentation techniques here, namely abduction and rule inversion, that produce more rules from the already learnt rules obtained from a rule generator. We now explain the two approaches in detail below.

3.1 Abduction

The goal of abductive reasoning (or abduction) is to find the best explanation from a given set of observations (Pierce, 1935). Traditionally, abductive reasoning has been based on first order logic rules. For instance, consider the following rule learnt from the FB15K-237 knowledge base:

$\texttt{BornIn}(\texttt{X},\texttt{U}) \land \texttt{PlaceInCountry}(\texttt{U},\texttt{Y}) \Rightarrow \texttt{Nationality}(\texttt{X},\texttt{Y})$. We are also given the following observations (or facts) $\texttt{BornIn}(\texttt{Oprah},\texttt{Mississippi})$ and $\texttt{Nationality}(\texttt{Oprah},\texttt{US})$ in the knowledge graph. While deductive inference would only determine that the above rule failed to fire for a KG, abductive reasoning would provide an exact explanation for failure of the above rule: the fact $\texttt{PlaceInCountry}(\texttt{Mississippi},\texttt{US})$ is currently missing from the knowledge graph.

Consequently, abduction helps in determining the smallest set of *assumptions* that could be made which will be sufficient to deduce the observations. For instance, in the above example, if we *assume* $\texttt{PlaceInCountry}(\texttt{Mississippi},\texttt{US})$ to be true in KG, the above rule would get fired given the other two observations. In our proposed formulation, we explicitly encode an *assumption* in what we call as abductive rule. While designing an abductive rule, the inverse of assumed relation forms the head and the remaining relations in the original rule form the body.

Continuing with our previous example, the abductive rule with respect to *assumption* $\texttt{PlaceInCountry}(\texttt{Mississippi},\texttt{US})$ would be: $\texttt{Nationality}^{-1}(\texttt{US},\texttt{Oprah}) \land \texttt{BornIn}(\texttt{Oprah},\texttt{Mississippi}) \Rightarrow \texttt{PlaceInCountry}^{-1}(\texttt{US},\texttt{Mississippi})$ where $\texttt{Nationality}^{-1}$ and $\texttt{PlaceInCountry}^{-1}$ are the inverses of $\texttt{Nationality}$ and $\texttt{PlaceInCountry}$ relations respectively and have been introduced to maintain the consistency in the path connecting the entities in the new head relation. Note that the above rule for $\texttt{PlaceInCountry}$ can also be formulated as: $\texttt{BornIn}^{-1}(\texttt{Mississippi},\texttt{Oprah}) \land \texttt{Nationality}(\texttt{Oprah},\texttt{US}) \Rightarrow \texttt{PlaceInCountry}(\texttt{Mississippi},\texttt{US})$. Finally, in order to produce a generic abductive rule for the above example, we replace the constants with variables to obtain the final abductive rule as: $\texttt{Nationality}^{-1}(\texttt{Y},\texttt{X}) \land \texttt{BornIn}(\texttt{X},\texttt{U}) \Rightarrow \texttt{PlaceInCountry}^{-1}(\texttt{Y},\texttt{X})$.

More formally, if currently there is a given rule with n relations in its body and it fails to ground, but groundings for $(n-1)$ relations in the body are present in KG, abduction infers that the rule failed to fire because of the remaining relation in the rule body that does not have possible grounding in KG. Abduction then *assumes* an abductive rule for all possibilities of this relation with no grounding. Consequently, if the original rule is:

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

the corresponding abductive rules acquired by our approach are:

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

3.2 Rule Inversion

With the focus of distilling more knowledge from a superior rule, we propose our second rule augmentation technique which is based on rule inversion. The key idea is that whenever a forward path between a pair of entities is true, we assume the backward path between them to be true. As an example, if the path $\texttt{Oprah} \xrightarrow{\texttt{BornIn}} \texttt{Mississippi} \xrightarrow{\texttt{PlaceInCountry}} \texttt{US}$ is true in KG for a given rule, we infer that path $\texttt{US} \xrightarrow{\texttt{PlaceInCountry}^{-1}} \texttt{Mississippi} \xrightarrow{\texttt{BornIn}^{-1}} \texttt{Oprah}$ would also hold true in it. Formally, for every original rule present in the KG:

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

we equip the KG with the following inverted rule:

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

Our approach to rule augmentation is generic and can be integrated with any of the existing NS-KG models. For example, once RNNLogic model has trained and has generated the desired rules, our approach would enhance these rules by designing abductive and inverted rules corresponding to each original rule. These augmented rules in turn would be passed on to RNNLogic+ predictor for training.

4 Experiments

Datasets: We choose three datasets for evaluation: WN18RR (Dettmers et al., 2018), FB15k-237 (Toutanova and Chen, 2015) and UMLS (Kok and Domingos, 2007). We use standard splits for WN18RR and FB15k-237. Since there are no standard splits for UMLS, we use the splits used by RNNLogic for evaluation (created by randomly sampling 30% triplets for training, 20% for validation and the rest 50% for testing).

Metrics: For each triplet $(\mathbf{h},\mathbf{r},\mathbf{t})$ in the test set, traditionally queries of the form $(\mathbf{h},\mathbf{r},?)$ and $(?,\mathbf{r},\mathbf{t})$ are created for evaluation, with answers $\mathbf{t}$ and $\mathbf{h}$ respectively. We model the $(?,\mathbf{r},\mathbf{t})$ query as $(\mathbf{t},\mathbf{r}^{-1},?)$ with the same answer $\mathbf{h}$, where $\mathbf{r}^{-1}$ is the inverse relation for $\mathbf{r}$. In order to train the

Table 1: Rules per Dataset. `INV` and `ABD` represent rule inversion and abduction respectively.

Dataset	#Rules	#Rules + INV	#Rules + ABD	#Rules + INV + ABD
WN18RR	6135	8742	18251	23304
UMLS	91908	171526	322464	564374
FB15k-237	126137	174658	295403	392280

model over the inverse relations, we similarly 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.

Given ranks for all queries, we report the Mean Rank (MR), Mean Reciprocal Rank (MRR) and Hit@k (H@k, k = 1, 3, 10) under the filtered setting (Bordes et al., 2013). 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.

Experimental Setup: We train the rule generator in RNNLogic with optimal hyperparameters provided in the paper 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[1] (which most closely reproduces the numbers stated in their paper). We augment these rules by abduction, and then rule inversion on both the original rules and the rules formed after abduction. We present statistics detailing the number of rules used per dataset after each augmentation step in Table 1.

We use similar methodology for training RNNLogic+ model as in the original work. New rule embeddings are created for all the rules 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 the best over 5 epochs of training.

For RNNLogic+ with RotatE (equation 2), we use the following formulation of $\mathtt{RotatE}(\mathtt{h}, \mathtt{r}, \mathtt{t})$:

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

where $\mathtt{d}$ is the 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 $\mathbf{x}_t$. The hyperparameter η in equation (2) representing the relative weight is set to 0.01, 0.05 and 0.1 for WN18RR, FB15k-237 and UMLS respectively. The RotatE embedding dimension is set to 200, 500 and 1000 for WN18RR, FB15k-237 and UMLS respectively. We keep a consistent batch size of 8, 4 and 32 for WN18RR, FB15k-237 and UMLS respectively.

Results: We present the results on WN18RR and UMLS in Table 2, and on FB15k-237 in Table 3. We consider RNNLogic+ (equation (1)) and RNNLogic+ with RotatE (equation (2)) as our baselines, and present results on the original and augmented rule sets where abductive augmentation is represented as `ABD` and rule inversion is represented as `INV` in both the tables. We use both abduction and rule inversion in the case of WN18RR and UMLS, but only abduction in the case of FB15k-237 due to computational limitations.

We see that in both RNNLogic+ (row 1 vs 2) and RNNLogic+ with RotatE (row 3 vs 4), there is a significant increase in performance across datasets after performing rule augmentation. In particular, we obtain 23% improvement in the MR over RNNLogic+ on UMLS and 3.4 points of improvement in MRR over RNNLogic+ on WN18RR. We claim that the reason for this is that our rule augmentation techniques result in better utilization of the learned high-quality rules from RNNLogic. The groundings from the new rules can be used to create new paths between entities in the knowledge graph, resulting in more informed scores as computed by RNNLogic+. Further, rule augmentation also complements RotatE ensembling (row 4) in capturing more information about the KG for the model.

5 Analysis of Abduction and Inverse Rules

We perform three empirical analyses to answer the following questions: **Q1.** Why do the augmented rules help in improving the model performance? **Q2**. What is the individual effect of each type of augmentation on the performance? **Q3**. What is the need for rule augmentation in the first place ?

Quality of new rules: To answer **Q1**, we conjecture that abduction and rule inversion result in high-quality rules which helps in performance improvement. Following the notation in AMIE (Galárraga et al., 2013), let the rules be of the form $\mathtt{B} \Rightarrow \mathtt{r}(\mathtt{x}, \mathtt{y})$. We consider the set of positive triples P to be the $(\mathtt{h}, \mathtt{r}, \mathtt{t})$ triples in the training and test

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

Table 2: Results of reasoning on WN18RR and UMLS. *RL+ and RE represent RNNLogic+ and RotatE respectively

Algorithm	WN18RR					UMLS				
	MR	MRR	H@1	H@3	H@10	MR	MRR	H@1	H@3	H@10
RNNLogic+	5857.65	0.496	0.455	0.514	0.574	3.66	0.814	0.712	0.903	0.957
RNNLogic + w/ INV + ABD	5156.38	0.527	0.483	0.549	0.613	2.83	0.839	0.752	0.917	0.966
RNNLogic + w/ RotatE	4445.79	0.516	0.474	0.534	0.602	3.17	0.815	0.712	0.901	0.960
RL + w/ INV + ABD + RE*	**4231.77**	**0.550**	**0.510**	**0.572**	**0.631**	**2.82**	**0.842**	**0.753**	**0.918**	**0.971**

Table 3: Results of reasoning on FB15k-237. RL+ and RE represents RNNLogic+ and RotatE respectively.

Algorithm	FB15K-237				
	MR	MRR	H@1	H@3	H@10
RL+	256.14	0.329	0.240	0.361	0.506
RL+ w/ ABD	218.11	0.345	0.257	0.379	0.519
RL+ w/ RE	217.30	0.343	0.256	0.375	0.524
RL+ w/ ABD + RE*	**198.81**	**0.353**	**0.265**	**0.387**	**0.529**

Table 4: Number of high quality rules before and after rule augmentation.

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**

set. To evaluate the quality of the learned rules, we employ the following two metrics:

(a) **FOIL:** Let $\mathtt{Path}(\mathtt{x},\mathtt{B},\mathtt{y})$ be the set of paths from $\mathtt{x}$ to $\mathtt{y}$ that act as groundings for the rule body $\mathtt{B}$. 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 the quality of a rule as:

$$\mathtt{FOIL}(\mathtt{B} \Rightarrow \mathtt{r}) = \frac{\sum_{\mathtt{r}(\mathtt{x},\mathtt{y})\in \mathtt{P}} |\mathtt{Path}(\mathtt{x},\mathtt{B},\mathtt{y})|}{\sum_{(\mathtt{x},\mathtt{y})} |\mathtt{Path}(\mathtt{x},\mathtt{B},\mathtt{y})|}$$

The key difference from FOIL 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.

(b) **AMIE:** We also calculate the rules scores according to the PCA Confidence metric introduced in AMIE (2013). It uses a Partial Closed World assumption and assumes that if we know a positive triplet $\mathtt{r}(\mathtt{x},\mathtt{y})$, then we know all $\mathtt{y}'$ in $\mathtt{r}(\mathtt{x},\mathtt{y}')$ for the same $\mathtt{x}$ and $\mathtt{r}$. To show that the rules are high quality regardless of the number of groundings, we use the score $\mathtt{PCAConf}(\mathtt{B} \Rightarrow \mathtt{r})$:

$$\frac{\#(\mathtt{x},\mathtt{y}) : |\mathtt{Path}(\mathtt{x},\mathtt{B},\mathtt{y})| > 0 \land \mathtt{r}(\mathtt{x},\mathtt{y}) \in \mathtt{P}}{\#(\mathtt{x},\mathtt{y}) : |\mathtt{Path}(\mathtt{x},\mathtt{B},\mathtt{y})| > 0 \land \exists \mathtt{y}' : \mathtt{r}(\mathtt{x},\mathtt{y}') \in \mathtt{P}}$$

Essentially, it is the number of positive examples satisfied by the rule divided by the total number of $(\mathtt{x},\mathtt{y})$ satisfied by the rule such that $\mathtt{r}(\mathtt{x},\mathtt{y}')$ is a positive example for some $\mathtt{y}'$.

Typically, negative sampling is used to calculate these metrics as it is computationally expensive to compute exhaustive negative examples. However, we calculate these metrics considering the entire knowledge graph, while restricting $\mathtt{x}$ in both formulae to the set of entities that occur as a head in the test dataset. Therefore, all 135 entities are considered in the head for UMLS, while 5364 entities are considered in the head for WN18RR.

Table 4 contains 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 notice that there is a massive increase in the number of high-quality rules after the rule augmentation is performed, nearly tripling in the case of abduction (row 1 vs row 3). This is because the augmented rules mostly exploit the same groundings as the original rules, which are already known to be of high-quality. These groundings can be used in the context of different relations in the rule head, after abduction and inversion.

Interestingly, we find that the ratio of high-quality rules to the total number of rules mostly goes down after the rule augmentation. However, the model still gives better performance. This is because the RNNLogic+ score considers only the number of groundings of the rule. If this is nearly 0 for all possible tails given a head, then it will not affect the model. Our rule scoring function assigns low scores to all such rules. Thus, adding rules that have low scores according to our metrics does not affect performance for RNNLogic+. We confirmed this by running the model on the filtered rule set which is obtained by considering only the rules whose score is above 0.1. In all cases,

Table 5: Ablation study performed on WN18RR and UMLS for inversion (`INV`) and abduction (`ABD`). *R+ and RE represent RNNLogic+ and RotatE respectively.

Algorithm	WN18RR					UMLS				
	MR	MRR	H@1	H@3	H@10	MR	MRR	H@1	H@3	H@10
`RNNLogic` + `w/ RotatE`	4445.79	0.516	0.474	0.534	0.602	3.17	0.815	0.712	0.901	0.960
`RNNLogic` + `w/ INV` + `RE`	4401.85	0.523	0.480	0.543	0.608	3.14	0.821	0.72	0.910	0.962
`RNNLogic` + `w/ ABD` + `RE`	4269.02	0.545	0.500	0.569	0.629	3.07	0.831	0.739	0.913	0.964
`R` + `w/ INV` + `ABD` + `RE`*	**4231.77**	**0.550**	**0.510**	**0.572**	**0.631**	**2.82**	**0.842**	**0.753**	**0.918**	**0.971**

we obtained performance within 1 point MRR of the full rule set. This opens up opportunities for parameter efficiency as the number of rule embeddings required by the model goes down by 50-70% when we consider only the filtered rules. **Q1** can now be answered affirmatively: abduction and rule inversion augment the rule set with high-quality rules and this is the reason behind the improved performance.

Ablation: To answer **Q2**, we perform an ablation study for inversion (`INV`) and abduction (`ABD`) on RNNLogic+ with RotatE model for WN18RR and UMLS datasets in order to observe the impact of each type of augmentation. The final results for all metrics are presented in Table 5.

In general, abduction (row 3) by itself gives larger improvements than rule inversion (row 2). This is because as we noticed in the previous section, abduction adds a higher number of high-quality rules into the rule set. However, the effects are incremental, as combined effect of inversion and abduction (`INV + ABD`, row 4) is better than `INV` and `ABD` taken alone on both the datasets.

Also, the improvements with inversion (`INV`) is more significant in UMLS than in WN18RR dataset. This is because the rule set for WN18RR obtained from RNNLogic already contains most of the inverses, with only around 25% new rules being created after inversion. The rule set for UMLS by comparison nearly doubles after inversion. The same holds for `ABD` vs `INV` + `ABD` as well.

Rule Generation vs Rule Augmentation: Since the rule generator of RNNLogic is an LSTM, a natural question is: why should we perform rule augmentation rather than generating more rules from the LSTM ? To answer this (**Q3**), we perform a set of experiments on WN18RR where we train RNNLogic model from scratch for a lesser number of epochs to generate (100, 200, 300) rules per relation and augment the resulting rules by abductive inference. Although the number of abductive rules per relation vary, the average number of abductive

Table 6: Performance of augmentation with larger rule sets on WN18RR. #Rules is the number of rules per relation generated from RNNLogic model.

# Rules	ABD	MR	MRR	H@1	H@3	H@10
100	No	7600.81	0.457	0.430	0.468	0.509
	Yes	6172.53	0.483	0.445	0.500	0.554
200	No	7322.99	0.461	0.433	0.473	0.516
	Yes	6085.07	0.488	0.451	0.504	0.561
300	No	7301.44	0.467	0.438	0.480	0.521
	Yes	**5982.33**	**0.492**	**0.454**	**0.508**	**0.562**

rules per relation are 168, 364, 494 for 100, 200, 300 original relations respectively. The results are presented in Table 6.

We observe from the table that the rule augmentation leads to dramatic improvement in all cases. Additionally, the performance of RNNLogic+ with 300 rules per relation (row 5) is significantly lower than the performance of `ABD` augmented RNNLogic+ with 100 rules per relation (100+168 rules per relation in total) (row 2) even though the number of rules are higher in the first case. Thus, to answer **Q3**, rule augmentation is more beneficial than using more rules from the rule generator. It helps in exploiting a small number of high-quality rules to their fullest potential .

6 Conclusion and Future Work

We demonstrate the utility of rule inversion and abduction as rule augmentation techniques in the context of Neuro-Symbolic Knowledge Graph models. Specifically, we obtain significant increase in performance on RNNLogic+ and RNNLogic+ with RotatE by rule augmentation, two base models which already obtain near-state-of-the-art results on datasets such as WN18RR. Rule augmentation makes better use of high-quality rules generated by rule generators such as RNNLogic.

We then analyze our results by evaluating quality of generated rules and ablation. Through ablation, we showed that the improvements with inversion and abduction are complementary, both to each other and to RotatE ensembling. Finally, we demonstrate that it is more beneficial to begin with a seed set of high-quality rules and then augment them, rather than requiring the rule generator to produce larger number of rules.

Our work opens several avenues for future work. Using the modified FOIL and PCA Confidence scores to filter the rule sets allows us to try out other augmentation methods to discover more high-quality rules, while not increasing the parameters in the model drastically. It might also be of interest to incorporate these scores into RNNLogic, as it would help in better training the rule generator.

Acknowledgements

We thank IIT Delhi HPC facility for compute resources.

References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, page 1247–1250.

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.

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.

G. A. Miller. 1995. Wordnet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41.

Stephen Muggleton and Luc de Raedt. 1994. Inductive Logic Programming: Theory and Methods. *The Journal of Logic Programming*, 19-20:629–679. Special Issue: Ten Years of Logic Programming.

Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. 2015. Compositional Vector Space Models for Knowledge Base Completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 156–166, Beijing, China. Association for Computational Linguistics.

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.

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.

Yueqing Sun, Qi Shi, Le Qi, and Yu Zhang. 2022. JointLK: Joint reasoning with language models and knowledge graphs for commonsense question answering. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5049–5060, Seattle, United States. Association for Computational Linguistics.

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.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In *ICML*, page 2071–2080. JMLR.org.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zhigang Chen. 2014. Knowledge Graph and Text Jointly Embedding. In *EMNLP*, pages 1591–1601.

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.

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*.