Learning Inference Rules with Neural TP-Reasoner

Kezhen Chen
Northwestern University
kzchen@u.northwestern.edu

Qiuyuan Huang
Microsoft Research Redmond
qihua@microsoft.com

Paul Smolensky
Microsoft Research Redmond
Johns Hopkins University
psmo@microsoft.com

Kenneth Forbus
Northwestern University
forbus@northwestern.edu

Jianfeng Gao
Microsoft Research Redmond
jfgao@microsoft.com

Abstract

Most standard deep learning models do not perform logical rule-based reasoning like human and are hard to understand. We present a novel neural architecture, ***T**ensor **P**roduct **Reasoner** (**TP-Reasoner**)*, for learning inference rules represented with a structured representation. In TP-Reasoner, we aim to integrate symbolic inference and deep learning: we utilize the ability of *Tensor Product Representation* in a neural model for learning and reasoning inference rules, which extracts intermediate representations of logical rules from a knowledge base reasoning task. TP-Reasoner achieves comparable results with baseline models. Analysis of learned inference rules in TP-Reasoner shows the interpretability of logical composition via a strong neuro-symbolic representation, a novel model expressivity, and an explicit tensor product expressions.

1 Introduction

When people perform inference based on their knowledge, they infer new facts using the existing facts via inference rules. Many researchers have used structured representations to model reasoning process and inference rules. A lot of evidence has shown that relational structured representations are important for human cognition, e.g., (Goldin-Meadow and Gentner, 2003; Forbus et al., 2017; Crouse et al., 2018; Chen and Forbus, 2018; Chen et al., 2019; Lee et al., 2019). Recently, many researchers use deep learning models to perform knowledge base reasoning and have achieved impressive results. However, most existing deep learning models do not explicitly represent human-like inference process and these models lack interpretability. In this paper, we propose a novel neural architecture, **TP-Reasoner**, for presenting knowledge base reasoning and inference. Given a set of relational facts in knowledge base, models are required to infer new facts. To model inference process, existing facts in the knowledge base are regarded as the antecedents and the inferred facts are regarded as consequences. TP-Reasoner represents relational facts using *Tensor Product Representation (TPR)* (Smolensky, 1990) and the inputs for this model are TPR of antecedents. TP-Reasoner has many reasoning heads and each head performs TPR 'binding' and TPR 'unbinding' operations to apply an inference rule on antecedents and generates a new fact represented as TPR. Outputs from TP-Reasoner also conformed the form of TPR, which contains the existing relational facts (antecedents) in knowledge base and inferred facts (consequences). By employing TPRs, the model achieves comparable performance and increases the interpretability, i.e., the learned inference rules can be explained. Our contributions in this paper are as follows. (i) We present a new *TP-Reasoner* model, which performs reasoning on relational facts in a knowledge base and infers new facts. To our knowledge, this is the first neural model which uses *Tensor Product Representation* to explicitly model human-like reasoning process. (ii) Comparable results on Kinship dataset show that TP-Reasoner

NeurIPS 2020 Workshop on BabyMind, Vancouver, Canada.

has reasonable reasoning ability. (iii) Analysis of reasoning heads in TP-Reasoner provides strong interpretability to understand learned inference rules.

2 Related Work

Rule learning with neural models is a popular task in research area. Many works have achieved impressive results. For example, Yang et al. (2017); Manhaeve et al. (2018); Raedt (2017); Evans and Grefenstette (2018); Zhang et al. (2019); Ho et al. (2018); Weber et al. (2019) regard this problem as learning embedding for rules. For example, Wang et al. (2014) uses an Iterative Structural Gradient algorithm that alternates gradient-based search for parameters of a probabilistic model. Yang et al. (2017) reduces the rule learning problem to algebraic operations on neural-embedding-based representations of a given knowledge base. In our TP-Reasoner model, each inference rule is described with TPR structures. Relations in each rule are modeled as embedding vectors instead of a matrix operator as in Yang et al. (2017). TPR is a promising technique for encoding symbolic structural information and modeling symbolic reasoning in vector space. TPR 'binding' and 'unbinding' operations have been used in natural-language tasks(Palangi et al., 2018; Huang et al., 2018, 2019; Kezhen et al., 2020). Some researchers also use TPRs for modeling deductive reasoning processes both on a rule-based model and deep learning models in vector space (Lee et al., 2016; Smolensky et al., 2016; Schlag and Schmidhuber, 2018). However, none of these previous models utilizes TPR 'binding' and 'unbinding' to learn inference rules with differentiable learning, as done in our model.

3 TP-Reasoner

3.1 Structured Representations using TPRs

The Tensor Product Representation (TPR) of a symbolic structure $\mathtt{S}$ is determined by (i) decomposing $\mathtt{S}$ into a set of structural roles $\{\mathtt{r}_i\}$, which are respectively bound to the fillers $\{\mathtt{f}_i\}$; (ii) embedding all roles in a role vector space $\mathbb{R}^{d_r}$, and all fillers in a filler vector space $\mathbb{R}^{d_f}$, and (iii) embedding $\mathtt{S}$ as the tensor in $\mathbb{R}^{d_r} \otimes \mathbb{R}^{d_f}$ (or equivalently the matrix in $\mathbb{R}^{d_r \times d_f}$):

$$\mathcal{T} = \sum_i f_i \otimes r_i = \sum_i f_i r_i^\top \tag{1}$$

For example, adopting linear positional roles, the string $\mathtt{ABC}$ has TPR $\mathcal{T}_{\mathtt{ABC}} = a \otimes r_1 + b \otimes r_2 + c \otimes r_3$, where a is the vector embedding $\mathtt{A}$, r_1 is the vector embedding $\mathtt{R}_1$, etc.

When the role vectors $\{r_i\}$ are linearly independent, there is a set of dual or **unbinding** vectors $\{u_j\}$ defined by the property that $r_i^\top u_j = \delta_{ij}$, from which it follows that any role r_k can be **unbound** from $\mathcal{T}$ to give its filler f_k simply by computing $\mathcal{T} u_k$. For example, $\mathcal{T}_{\mathtt{ABC}} u_2 = b$. When the role vectors are orthonormal, $u_i = r_i$.

In a knowledge base, facts are written as binary relational tuples $(rel, subj, obj)$ where $subj, obj$ indicate the subject and object of a relation rel. Suppose that $rel, subj, obj$ are embedded as vectors r, s, o. This relational tuple can be regarded as the filler $subj$ bound to a complex role $\tilde{r} = (rel, obj)$; $\tilde{r}$ is itself a structure in which the inner filler obj is bound to an inner role rel. Then $\tilde{r}$ is embedded as the vector $\tilde{r} = o \otimes r$; therefore the triple as a whole is embedded as $s \otimes \tilde{r} = s \otimes (o \otimes r)$. The tensor product is associative, so we can omit parentheses, and to align with the ordering $(rel, subj, obj)$ we can reorder the factors and use the equivalent tensor $r \otimes s \otimes o$. Given a knowledge base with m facts $\{(rel_i, subj_i, obj_i)\}_{i=1}^m$, and vector embeddings r_i, s_i, o_i for $rel_i, subj_i, obj_i$, the TPR for the entire knowledge base is:

$$\mathbf{H} = \sum_{i=1}^{m} r_i \otimes s_i \otimes o_i \tag{2}$$

Let the unbinding vectors for r_i, s_i, o_i be r_i', s_i', o_i'. These can be used to recover individual elements and pairs of elements from the knowledge base via the tensor inner product; for example:

$$r_i' \cdot \mathbf{H} \equiv \mathbf{H} \cdot_1 r_i' = \Sigma_{j:r_j = r_i} s_j \otimes o_j \text{ where } [\mathbf{H} \cdot_1 r_i']_{\beta\gamma} \equiv \Sigma_\alpha [\mathbf{H}]_{\alpha\beta\gamma} [r_i']_\alpha; \text{ then} \tag{3}$$

$$s_i' \cdot \Sigma_{j:r_j = r_i} s_j \otimes o_j = \Sigma_{j:s_j = s_i \wedge r_j = r_i} o_j \tag{4}$$

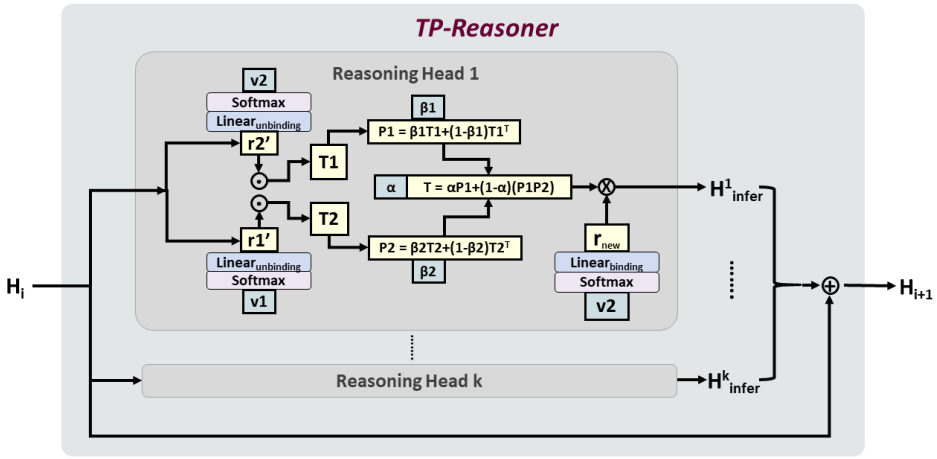

Figure 1: Overview of TP-Reasoner. Rule: $(r_1, ?s, ?x) \wedge (r_2, ?x, ?o) \Rightarrow (r_{new}, ?s, ?o)$ when $\beta_1 = 1 = \beta_2, \alpha = 0$; $\beta_k = 0$ reverses the arguments of r_k; $\alpha = 1$ eliminates the r_2 condition.

Eq. 3 provides the TPR for the set of pairs resulting from the query $(rel_i, ?s, ?o)$, and Eq. 4 gives the TPR for the set of entities resulting from $(rel_i, subj_i, ?o)$. When the embedding vectors are orthonormal, the result of the query $(r_i, ?s, ?x) \wedge (r_k, ?x, ?o)$ is the product of the results of the sub-queries, considered as matrices:

$$\left[\Sigma_{j:r_j=r_i} s_j o_j^\top\right]\left[\Sigma_{l:r_l=r_k} s_l o_l^\top\right] = \Sigma_{j,l:r_j=r_i \wedge r_l=r_k} s_j (o_j \cdot s_l) o_l^\top = \Sigma_{j,l:r_j=r_i \wedge r_l=r_k \wedge o_j=s_l} s_j o_l^\top \quad (5)$$

In next section, we will describe how the TP-Reasoner model uses the structured representations to learn and apply inference rules.

3.2 Model Architecture

TP-Reasoner architecture has k reasoning heads. These heads aim to learn inference rules for knowledge base reasoning using TPR binding and unbinding operations. Given a knowledge base with n_f facts, n_r relations and n_e entities, TP-Reasoner uses a relation embedding layer and an entity embedding layer to convert the relations, subjects and objects of all facts to embedding vectors firstly. Using the TPR binding operation from Eq.2, these facts are encoded as a TPR $\mathbf{H}_i$. The reasoning head j of TP-Reasoner takes $\mathbf{H}_i$ as input and applies inference rules on $\mathbf{H}_i$ to generate the TPR of inferred facts $\mathbf{H}^j_{infer}$. As each reasoning head learns different rules, each head generates a TPR of inferred facts. The TPR of inferred facts from all heads and TPR of antecedents are summed together to generate the output TPR $\mathbf{H}_{i+1}$ as following:

$$\mathbf{H}_{i+1} = \mathbf{H}_i + \sum_{j=1}^{k} \mathbf{H}^j_{infer} \quad (6)$$

TP-Reasoner can perform multi-step inference by passing the output TPR $\mathbf{H}_{i+1}$ back to the model again. Figure 1 shows an overview of this model.

Each reasoning head of TP-Reasoner has m unbinding relation vectors and a relation binding vector r_{new}. The inner product between $\mathbf{H}_i$ and each unbinding relation vector r'_x of relation rel_x is computed as:

$$\mathbf{T}_x = \mathbf{H}_i \cdot r'_x = s_x \otimes o_x \quad (7)$$

As all relation binding and unbinding vectors are independent with each other, if there are facts with the relation rel_x, the inner product between $\mathbf{H}_i$ and r'_x can unbind the tensor product $\mathbf{T}_x$ of objects and subjects that hold this relation. With the unbinding operation, reasoning heads match the antecedents

of inference rules. To learn the order of the object and subject, we use a coefficient β between 0 and 1 to determine whether the rule needs to switch the order of object and subject. If the order is changed, the transpose of $\mathbf{T}_x$ is used, i.e. $\beta\mathbf{T}_x + (1-\beta)\mathbf{T}_x^\top$. With m unbinding relation vectors, the head can learn inference rules with length m. For each unbound $\mathbf{T}_x$, we use matrix multiplication between them to run logic chaining inference. As entity embedding vectors are independent with each other, intermediate entities are canceled, e.g. $(rel_1, Entity_0, Entity_1) \wedge (rel_2, Entity_1, Entity_2) \wedge ... \wedge (rel_k, Entity_{k-1}, Entity_k) \Rightarrow (rel_{k+1}, Entity_0, Entity_k)$. Finally, the new relation binding vector r_{new} binds the pair of entities after chaining inference to generate the TPR of a new fact. Specifically, in our implementation, we use a coefficient α to control whether the number of conditions in this inference rule is 1 or 2. Figure 1 presents an example with 2 unbinding vectors. The first head in the figure describes the rule $(r_1, ?s, ?x) \wedge (r_2, ?x, ?o) \Rightarrow (r_{new}, ?s, ?o)$ when $\beta_1 = 1 = \beta_2, \alpha = 0$. This process is as follows:

$$\mathbf{H}_{infer} = r_{new} \otimes (\alpha(\beta_1\mathbf{T}_1 + (1-\beta_1)\mathbf{T}_1^\top) + (1-\alpha)(\beta_2\mathbf{T}_2 + (1-\beta_2)\mathbf{T}_2^\top)) \tag{8}$$

We sum the TPRs of inferred facts from each reasoning head and the original $\mathbf{H}_i$ to generate the output $\mathbf{H}_{i+1}$. The output of current inference step is passed to the next step recursively. All inference steps share the same set of reasoning heads.

To simplify the implementation, binding and unbinding vectors of relations and entities are one-hot vectors. Thus, each head is only trained to learn to choose relations and coefficients for rule antecedents and consequent. During training, we use the mini-batch ADAM with batch size 8 and the learning rate initially set to 0.001. The number of training epochs is 200. TP-Reasoner outputs the new TPR after reasoning and inference. Given a testing fact, the unbinding vectors of two ground truth elements from the relation, object and subject in a tuple are used to unbind the other element from $\mathbf{H}_n$, the output from TP-Reasoner at the n recursive step. following the Eq. 3–4. We sum the Cross-Entropy loss on each unbound vector. Also, we add a regularization term in the loss to push the trained parameters in each reasoning head to softly one-hot so the inference rule can be interpreted.

4 Experiments

Model \ Data	Kinship
ISG	59.2
Neural LP	90.2
TP-Reasoner (ours)	90.2+

Figure 2: Results on Kinship dataset.

We conduct experiments on statistical relational learning task to test the reasoning ability of TP-Reasoner. Given a knowledge base, models are required to perform reasoning from the knowledge base, then to infer new facts. The goal is to retrieve a ranked list of entities for $?e$ based on a given query $(rel, entity1, ?e)$ from the inferred facts. The desired answer should be ranked as high as possible. We test TP-Reasoner on Kinship dataset which has 104 entities, 25 relations and 9587 facts. Following Yang et al. (2017), we randomly split the datasets into facts, train, test data with ratio 6:2:1. The evaluation metric is Hits@10. Figure 2 shows the experiment results that we compared TP-Reasoner with two rule-learning baselines, Iterative Structural Gradient (ISG) (Wang et al., 2014) and Neural LP (Yang et al., 2017). For all models, the rule length is picked as 2 and we use 200 reasoning heads in the model. From the results, TP-Reasoner can achieves the competitive performance as Neural LP model. After training, the inference rules can be analyzed from each head. In each head, trained the unbinding vectors have the information for antecedents for the learned rule in the head. The new relation binding vector shows the consequent of the rule. By exploring these trained parameters, we can understand the inference rules learned in these heads. For example, in head 160, the learned rule is $(Rel16, ?a, ?b) \wedge (Rel15, ?c, ?b) \Rightarrow (Rel21, ?a, ?c)$, and in head 176, the learned rule is $(Rel8, ?a, ?b) \wedge (Rel11, ?c, ?b) \Rightarrow (Rel7, ?a, ?c)$.

5 Conclusion and Future Work

In this paper, we propose a novel TP-Reasoner model to learn inference rules and perform reasoning. This model is tested on Kinship dataset. Experiments and analysis show that TP-Reasoner is promising for learning inference rules and performing interpretability reasoning on knowledge base. We will test this model on more larger datasets and construct more types of inference rules in future.

References

Kezhen Chen and Kenneth D. Forbus. 2018. Action recognition from skeleton data via analogical generalization over qualitative representations. In *Thirty-Second AAAI Conference*.

Kezhen Chen, Irina Rabkina, Matthew D. McLure, and Kenneth D. Forbus. 2019. Human-like sketch object recognition via analogical learning. In *Thirty-Third AAAI Conference*, volume 33, pages 1336–1343.

Maxwell Crouse, Clifton McFate, and Kenneth D. Forbus. 2018. Learning from unannotated qa pairs to analogically disanbiguate and answer questions. In *Thirty-Second AAAI Conference*.

Richard Evans and Edward Grefenstette. 2018. Learning explanatory rules from noisy data. In *JAIR*.

Kenneth.D. Forbus, Chen Liang, and Irina Rabkina. 2017. Representation and computation in cognitive models. In *Top Cognitive System*.

Susan Goldin-Meadow and Dedre Gentner. 2003. *Language in mind: Advances in the study of language and thought*. MIT Press.

Vinh Thinh Ho, Daria Stepanovaand Mohamed H. Gad-Elrab, Evgeny Kharlamovand, and Gerhard Weikum. 2018. Rule learning from knowledge graphs guided by embedding models. In *ISWC*.

Qiuyuan Huang, Li Deng, Dapeng Wu, chang Liu, and Xiaodong He. 2019. Attentive tensor product learning. In *Thirty-Third AAAI Conference*, volume 33.

Qiuyuan Huang, Paul Smolensky, Xiaodong He, Oliver Wu, and Li Deng. 2018. Tensor product generation networks for deep nlp modeling. In *NAACL*.

Chen Kezhen, Qiuyuan Huang, Hamid Palangi, Paul Smolensky, Kenneth Forbus, and Jianfeng Gao. 2020. Mapping natural-language problems to formal-language solutions using structured neural representations. In *ICML*.

Kuang-Huei Lee, Hamid Palangi, Xi Chen, Houdong Hu, and Jianfeng Gao. 2019. Learning visual relation priors for image-text matching and image captioning with neural scene graph generators. abs/1909.09953.

Moontae Lee, Xiaodong He, Wen-tau Yih, Jianfeng Gao, Li Deng, and Paul Smolensky. 2016. Reasoning in vector space: An exploratory study of question answering. In *ICLR*.

Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt.Deepproblog. 2018. Neural probabilistic logic programming. In *NeurIPS*.

Hamid Palangi, Paul Smolensky, Xiaodong He, and Li Deng. 2018. Question-answering with grammatically-interpretable representations. In *AAAI*.

Luc De Raedt. 2017. Multi-relational data mining. In *Springer*.

Imanol Schlag and Jurgen Schmidhuber. 2018. Learning to reason with third order tensor products. In *Neural Information Processing Systems*.

Paul Smolensky. 1990. Tensor product variable binding and the representation of symbolic structures in connectionist networks. In *Artificial Intelligence*, volume 46, pages 159–216.

Paul Smolensky, Moontae Lee, Xiaodong He, Wen-tau Yih, Jianfeng Gao, and Li Deng. 2016. Basic reasoning with tensor product representations. *arXiv preprint arXiv:1601.02745*.

William Wang, Kathryn Mazaitis, and William W Cohen. 2014. Structure learning via parameter learning. In *CIKM*.

Leon Weber, Pasquale Minervini, Jannes Munchmeyer, Ulf Leser, , and Tim Rocktaschel. 2019. Nlprolog: Reasoning with weak unification for question answering in natural language. In *ACL*.

Fan Yang, Zhilin Yang, and William W. Cohen. 2017. Differentiable learning of logical rules for knowledge base reasoning. In *NeurIPS*.

Wen Zhang, Bibek Paudel, , and Liang Wang. 2019. Iteratively learning embeddings and rules for knowledge graph reasoning. In *WWW.ACM*.