Potential Directions on Neuro Symbolic Reasoning

Jinhao Li
University of Electronic Science and Technology of China
stephlee175@gmail.com

While deep learning is booming with the help of large-scale datasets and computation resources (GPUs, TPUs, and cloud computing), doubts and controversies emerge as well. One main problem is that deep learning is lacking of **interpretability [17] and generalization capability [16].** Therefore, the work by [6] introduces *Neurosymbolic AI*, aiming to combine symbolic reasoning (the mainstream for AI before the hypergrowth of deep learning) with deep learning. Symbolic reasoning benefits from not only representing objects as symbols, but also establishing clear relations among objects with logic. Moreover, once we can conduct reasoning in the deep learning framework, tremendous combinations of symbols can be created for deep learning models to learn, which makes it possible for models to adapt to new domains. Hence, integrating symbolic reasoning with deep learning will bring in interpretability and generalization ability while being able to deal with large-scale data.

1 Neurosymbolic AI itself

Recent works on neurosymbolic AI can be classified into three categories as illustrated in Figure 1:

- As for the **query-like neurosymbolic AI** [14] [2], the structure of symbolic reasoning is first organized by semantic parsing from a certain question. The output of neural networks passes through the executor which makes it similar to a querying system, since the part of symbolic reasoning has already been processed. In addition, these methods are all question-driven. Although it shows a clear process on how a result is obtained (guarantees interpretability), the question-answering (QA) training mode hinders its adaption on other scenarios which requires a domain-specific QA dataset.
- The **logic learning neurosymbolic AI** has been introduced by works such as the logic tensor network (LTN) [18], and semantic loss [22]. The symbolic reasoning is complied into the neural networks by converting discrete logic to a continuous tensor and adding semantic loss functions on the basic neural network, respectively. The former one provides a way to train a neural network to conduct reasoning. However, the transmission from logic to tensor should be well designed. Besides, it cannot handle large-scale data. The latter one,

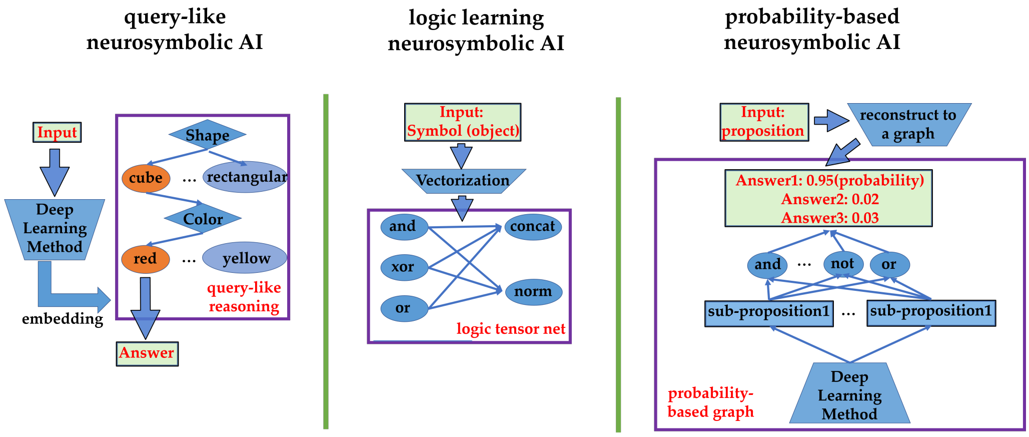

Figure 1: Three categories of neurosymbolic AI in recent works.

adding logical reasoning constraints to the training process, relies on domain knowledge for the semantic loss part, which leads to its weak generalization.

- The **probability-based neurosymbolic AI** can be considered as an extension of the Bayesian Network [13] [21]. One main difference is that deep learning method is adopted to extract features from more complex and unstructured data. Therefore, since the main body of these works is logical reasoning, it is restricted to certain scenarios and cannot deal with large-scale data.

From the above analysis, we can observe that recent works mainly focus on how to interpret a neural network's output, which is all considered in given circumstances like QA. In other words, the generalization ability attracts less attention while it is an extremely essential ability for our models to become intelligent. Therefore, an **object-based Neurosymbolic learning method** is proposed here as illustrated in Figure 2, which can **not only interpret the neural network output via logical reasoning, but also be capable of self-evolution dynamically**.

The object-based method is reasonable since an object is, from the symbolic reasoning perspective, a kind of symbol consisting of several attributes. For instance, a person possesses attributes such as height, weight, gender, and etc.. Furthermore, human usually conducts reasoning by establishing relationships among different objects. The Monet [5] is adopted to decompose a visual scene into several objects blocks in an unsupervised way with their corresponding representations formulated as:

$$\boldsymbol{\mu} = (\boldsymbol{\mu_1}, \boldsymbol{\mu_2}, \cdots, \boldsymbol{\mu_N}), \tag{1}$$

where $\boldsymbol{\mu_i} \in \mathbb{R}^d$ is the representation of object i with dimension d.

Notably, we still preserve the QA mode in the training process. Hence, as for the input question q, a transformer-based method [20] is adopted to get the question's representation formulated as:

$$\boldsymbol{q} = (\boldsymbol{q_1}, \boldsymbol{q_2}, \cdots, \boldsymbol{q_M}), \tag{2}$$

where $\boldsymbol{q_j} \in \mathbb{R}^d$ denotes the word representation from original question q with dimension d.

After obtaining both visual and linguistic object-based representations, we introduce a graph-based method by graph convolutional neural network (GCN) integrating multimodal knowledge (**visual representations for nodes and linguistic representations for edges in a graph**) for later reasoning. From my perspective, a graph neural network can bridge the gap between symbolic reasoning and deep learning since every node can be considered as, to some extent, a symbol and connected by edges. A graph is denoted by $\mathbb{G} = (\mathbb{V}, \mathbb{E})$, where $\mathbb{V}$ and $\mathbb{E}$ represents the set of nodes and edges of $\mathbb{G}$, respectively. Additionally, $\mathbb{G}$ is a weighted graph where its adjacency matrix $\mathbb{A}^{N\times N}$ is formulated as:

$$a_{ij} = \mathrm{softmax}(f(\boldsymbol{q} \mid (\boldsymbol{u_i}, \boldsymbol{u_j})) \times \boldsymbol{u_i}^T \boldsymbol{u_j}), \tag{3}$$

where $f(\cdot \mid \cdot)$ means the effect that the input question q has on two objects, and $softmax$, usually appearing in machine learning and deep learning, is a normalization method [4].

The last module in the proposed method is answering. After compressing the whole graph $\mathbb{G}$ into a single vector $\boldsymbol{g}$ formulated as $\boldsymbol{g} = \mathrm{Compress}(\mathbb{G})$ by the method proposed in [15], we can predict the answer a for the input question q by simply constructing a dense network [8]

$$a = \mathrm{softmax}(\mathrm{Dense}(\boldsymbol{g})). \tag{4}$$

To test whether the constructed object-based graph captures the relationships among objects, a real object-based graph is introduced for calculating the similarity with the constructed object-based graph. Notice that the similarity is calculated at a structure level that checks the number of nodes, edges, and etc. [10]. Besides, the proposed real object-based graph can be determined by expert knowledge where each node represents a real object as shown in Figure 2.

The loss function of our model proposed in Figure 2 can be formulated as:

$$\mathcal{L} = \mathcal{L}_{QA} + \mathcal{L}_{Sim}, \tag{5}$$

where $\mathcal{L}_{QA}$ represents the error for question answering while $\mathcal{L}_{sim}$ demonstrates the performance of constructing the object-based graph.

During the training process, each object's features, together with objects' relationships under the input question, are captured by the proposed GCN. This process is quite similar to human reasoning—several symbols in the brain are triggered when visual images and words come into mind, then these symbols connect with each other to form propositions. Since the proposed

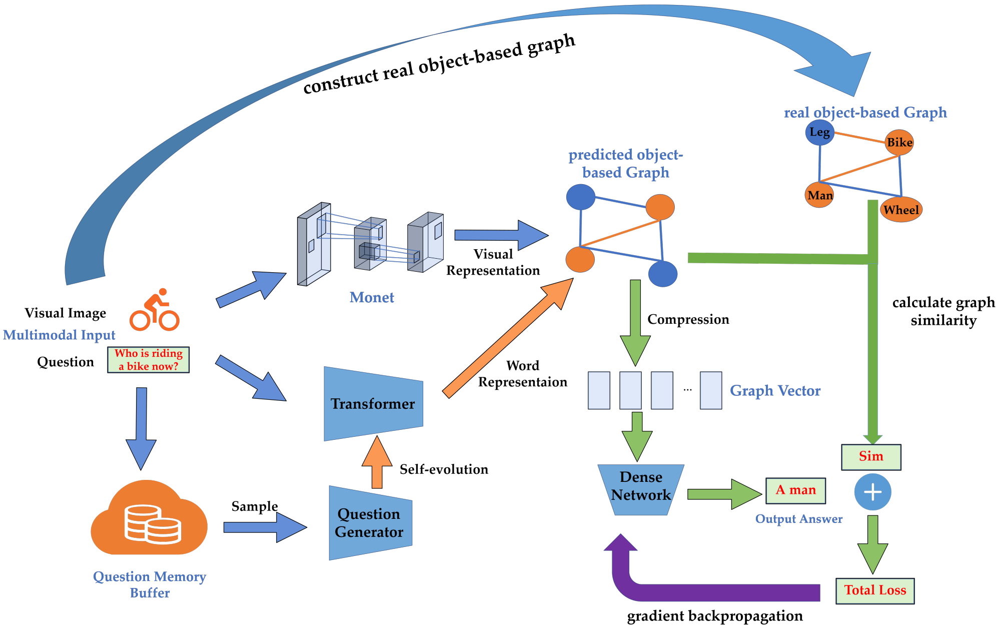

Figure 2: The object-based neurosymbolic learning method framework. In the object-based graph part, the objects triggered by the word representation are shown in orange color.

model learns at an object level, it is reasonable to adapt to other scenarios. New objects in other circumstances can be added to the graph, which will make this *knowledge base* bigger and bigger. Additionally, the main reason we maintain the QA mode is that common reasoning only needs several objects while others are not triggered. To ensure the efficiency of our model when the *knowledge base* is growing, this question-driven method should be adopted for the training process.

The capability of self-evolution is crucial for a model's generalization ability. Due to the question-driven training mode, **we propose a question generator to produce new questions for training based on the available data.** An action dictionary is introduced for the generation of rules which can be found in online verb corpus and formulated as:

$$\text{Act} = \{\text{Move}, \text{Flip}, \cdots\}. \tag{6}$$

Since all the historical questions can be stored in a memory buffer, we can extract entities (objects in other words) from this buffer easily by the approach proposed by [7]. Subsequently, we use these entities with the action dictionary to form various new questions. The Flesch-Kincaid readability test can be adopted to examine these questions' readability. Hence, the proposed model can achieve self-training and self-evolution without requiring more data. In addition, the question generator may generate some data beyond the captured data distribution, which will make our model more robust.

2 Applications of Neurosymbolic AI

It is known that neural network is fragile when dealing with attacks due to its black-box characteristic and bad generalization capability. For instance, the work by [9] has claimed that the output of a neural network can be totally inverted with high confidence when the input is injected with a small permutation. No matter one intruder attacks a certain target purposefully or aimlessly, it is difficult for our network to predict which part will be damaged since the effect may spread gradually, especially for large-scale distributed systems. For instance, the global model can be poisoned by injecting noise to gradients in the federated learning scenario [19], which may cause great damage to the whole system.

Several recent works have focused on how to defend against malicious attacks. A blockchain-based approach has been proposed by [12]. Although blockchain can be considered as a transparent and distributed transaction recorder, there is no real physical system (smart grid, communication system, and so on). The work by [11] has proposed an encryption-based approach to defend against attacks and alleviate the leakage of data. Neither of these two works focuses on the response to the attacks and how to predict attacks' effects on the whole system, which can give the system manager interpretable knowledge on which part of the system should be paid more attention to.

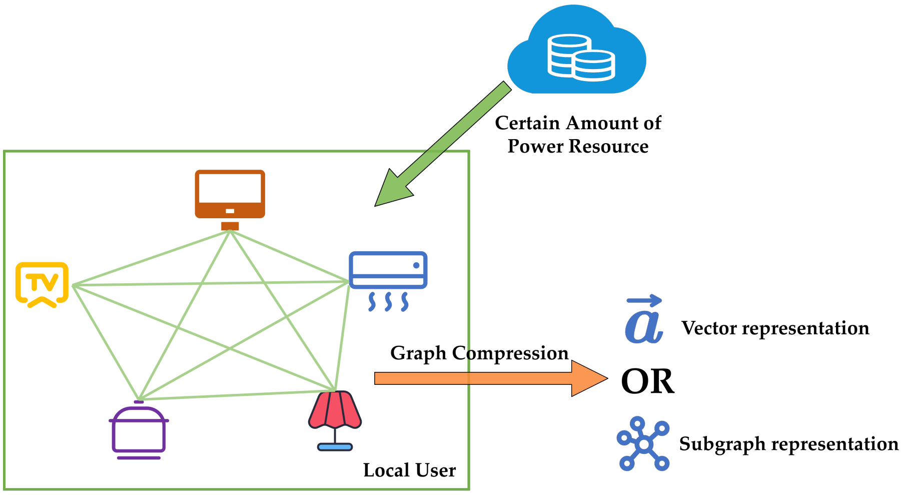
Certain Amount of Power Resource
Graph Compression
Local User
$\vec{a}$ Vector representation
OR
Subgraph representation

Figure 3: The object-based power graph of one local user in the smart grid

Therefore, an adaptive approach proposed in Figure 2 can be adopted for predicting the target of an attack. **Every component in a dynamic system (smart grid, IoT, communication system, and etc.) is abstracted as an object** with its own representation denoted by $\boldsymbol{r_o}$. For instance, in the smart grid, one local user's representation can be extracted from its local object-based power graph as illustrated in Figure 3. In Figure 3, all the power devices form an object-based graph with limited power resources. The local user's representation can be obtained from the power graph by [15] (the vector representation) or [1] (the subgraph representation), which depends on model setting.

We assume that one intruder attacks the infrastructure aimlessly in a smart grid denoted by $\mathbb{G} = (\mathbb{V}, \mathbb{E})$. Since the grid system varies dynamically, the representation of each user changes as well. Notably, each user's representation only shows its inner power characteristic as illustrated in Figure 2. The relationship between two users can be determined by their power transition which can be formulated as:

$$\boldsymbol{p_{ij}} = \{p_{ij}^{\text{resist}}, p_{ij}^{\text{react}}, p_{ij}^{\text{flow}}, \cdots\}, \tag{7}$$

where p_{ij}^{resist} is the total resistance in the power transition between user i and user j, p_{ij}^{react} is the total reactance, and p_{ij}^{flow} is the powerflow. Notice that how many factors should be included for demonstrating two objects' relationship is determined by expert knowledge and also varies under different scenarios. Hence, Eq(3) can be rewritten as:

$$a_{ij} = \text{softmax}(f(\boldsymbol{p_{ij}} \mid (\boldsymbol{r_i}, \boldsymbol{r_j}))). \tag{8}$$

To capture the change of the graph (system level), a graph memory pooling with volume N for storing graphs at different time slots t, which can be formulated as:

$$\text{Graph\_Pooling} = \{\mathbb{G}_t, \mathbb{G}_{t+1}, \cdots, \mathbb{G}_{t+N}\} \tag{9}$$

An attack can be detected by calculating the similarity [3] between the latest graph representation denoted by g_{t+N} and one sample from the graph memory pooling $g_{t'}$ where $t < t^{'} < t + N$.

The core idea of the proposed neurosymbolic method is to conduct reasoning after obtaining an attacked system graph. The system manager (experts) asks specific questions about the proposed model. For instance, the manager may ask whether the powerflow in a certain area has some problems. The question will be preprocessed into word representations as illustrated in Figure 2. All the objects related to this question are triggered and gathered to conduct logical reasoning. With the answer of reasoning, the manager can ask more questions to narrow suspected areas or just check suspected area's security. Notably, the proposed method can be adapted to any kind of system where objects are connected with each other, especially for IoT. Moreover, the proposed method is quite flexible since the definition of an object depends on different scenario settings.

References

[1] Bijaya Adhikari, Yao Zhang, Naren Ramakrishnan, and B. Aditya Prakash. Sub2vec: Feature learning for subgraphs. In Dinh Phung, Vincent S. Tseng, Geoffrey I. Webb, Bao Ho, Mo-

hadeseh Ganji, and Lida Rashidi, editors, *Advances in Knowledge Discovery and Data Mining*, pages 170–182, Cham, 2018. Springer International Publishing.

[2] Saeed Amizadeh, Hamid Palangi, Oleksandr Polozov, Yichen Huang, and Kazuhito Koishida. Neuro-symbolic visual reasoning: Disentangling "visual" from "reasoning", 2020.

[3] Yunsheng Bai, Hao Ding, Song Bian, Ting Chen, Yizhou Sun, and Wei Wang. Simgnn: A neural network approach to fast graph similarity computation, 2020.

[4] John Bridle. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann, 1990.

[5] Christopher P. Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation, 2019.

[6] Artur d'Avila Garcez and Luís C. Lamb. Neurosymbolic AI: the 3rd wave. *CoRR*, abs/2012.05876, 2020.

[7] John Giorgi, Xindi Wang, Nicola Sahar, Won Young Shin, Gary D. Bader, and Bo Wang. End-to-end named entity recognition and relation extraction using pre-trained language models, 2019.

[8] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. http://www.deeplearningbook.org.

[9] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015.

[10] Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. Graph matching networks for learning the similarity of graph structured objects, 2019.

[11] Haizhou Liu, Xuan Zhang, Xinwei Shen, and Hongbin Sun. A federated learning framework for smart grids: Securing power traces in collaborative learning, 2021.

[12] Yi Liu, Jialiang Peng, Jiawen Kang, Abdullah M. Iliyasu, Dusit Niyato, and Ahmed A. Abd El-Latif. A secure federated learning framework for 5g networks. *IEEE Wireless Communications*, 27(4):24–31, 2020.

[13] Robin Manhaeve, Sebastijan Dumančić, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Deepproblog: Neural probabilistic logic programming, 2018.

[14] Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B. Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision, 2019.

[15] Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. graph2vec: Learning distributed representations of graphs, 2017.

[16] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning, 2017.

[17] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead, 2019.

[18] Luciano Serafini and Artur d'Avila Garcez. Logic tensor networks: Deep learning and logical reasoning from data and knowledge, 2016.

[19] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. Data poisoning attacks against federated learning systems. In Liqun Chen, Ninghui Li, Kaitai Liang, and Steve Schneider, editors, *Computer Security – ESORICS 2020*, pages 480–501, Cham, 2020. Springer International Publishing.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[21] Leon Weber, Pasquale Minervini, Jannes Münchmeyer, Ulf Leser, and Tim Rocktäschel. Nl-prolog: Reasoning with weak unification for question answering in natural language, 2019.

[22] Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van den Broeck. A semantic loss function for deep learning with symbolic knowledge, 2018.