Learning with Molecules beyond Graph Neural Networks

Gustav Sourek
Czech Technical University
souregus@fel.cvut.cz

Filip Zelezny
Czech Technical University
zelezny@fel.cvut.cz

Ondrej Kuzelka
Czech Technical University
kuzelon2@fel.cvut.cz

1 Introduction

Deep learning has registered a tremendous success in the recent past. However, most of the applications are still limited to data in the form of fixed-size feature vectors (tensors). There are nevertheless many domains where the learning examples are highly structured and do not succumb themselves easily to the precanned form of numeric tensors. Particularly, important problems arise around molecular data, which can be understood as attributed graphs of atoms connected via chemical bonds.

To address the problem of deep learning from such structured data (i.e. without preprocessing into feature vectors) Graph Neural Networks (GNNs) have been proposed [7, 12]. GNNs can be viewed as a continuous, differentiable version of the famous Weisfeiler-Lehman (WL) label propagation algorithm used for graph isomorphism refutation checking [11]. In GNNs however, instead of discrete labels, a continuous node representation (embedding) is being successively propagated into nodes' neighborhoods, and vice versa for the corresponding gradient updates, which can be derived w.r.t. some learning target, such as atom or molecule classification. This paradigm has recently become highly popular [14]. Nevertheless, there are still considerable limitations to this class of models, stemming from the limited expressiveness of the WL test which is only based on the immediate neighborhood information gathered in each iteration [13, 5]. Consequently, information about more complex relational substructures, such as atom rings in molecules, cannot be properly extracted.

In this paper we demonstrate a deep learning framework which is inherently based in the highly expressive language of relational logic, enabling to, among other things, capture arbitrarily complex graph structures. We show how GNNs and similar models can be easily covered in the framework by specifying the underlying propagation rules in the relational logic. The declarative nature of the used language then allows to easily modify and extend the propagation schemes into complex structures, such as the molecular rings which we choose for a short demonstration in this paper.

2 Lifted Relational Neural Networks

We follow up on the framework of Lifted Relational Neural Networks (LRNNs) [8] allowing for *templated* modeling of diverse neural architectures oriented towards relational data[1]. It can be understood as a differentiable version of simple Datalog [10] programming, where the learning templates, encoding various neuro-relational architectures, take the form of *parameterized* logic programs [1]. It differs from the commonly used frameworks (e.g. TensorFlow) in its declarative nature, which is particularly useful for relational learning problems, such as learning with molecules [9].

[1] the framework is available at https://github.com/GustikS/NeuraLogic

Machine Learning for Molecules Workshop at NeurIPS 2020. https://ml4molecules.github.io

2.1 Learning Examples

In LRNNs, the learning examples are commonly represented with weighted ground logical facts. A learning example is then a set $E = \{(V_1, e_1), \ldots, (V_j, e_j)\}$, where each V_i is a real-valued tensor and each e_i is a ground fact, i.e. expression of the form

$$\begin{array}{l} \mathbf{V_1} :: \ \mathrm{p}_1(c_1^1, \ldots, c_q^1). \\ \ldots \\ \mathbf{V_j} :: \ \mathrm{p}_n(c_1^n, \ldots, c_r^n). \end{array}$$

where $p_1, \ldots, p_n$ are predicates with corresponding arities $q, \ldots, r$, and c_i^j are arbitrary constants. Note that this representation allows to encode arbitrary information about atoms (e.g. $2.35::ionEnergy(c_1, level_2)$) and their conformations (e.g. $[2.7, -1]::bond(c_1, o_2)$) in molecules, as demonstrated in the left part of Fig. 1.

2.2 Learning Template

The learning program, i.e. the declarative *template*, is then set of parameterized rules $\mathcal{T} = \{\alpha_i, \{W_j^{\alpha_i}\}\} = \{(W^i, c) \leftarrow (W_1^i, b_1), \ldots, (W_k^i, b_k)\}$, i.e. expression of the form

$$\begin{array}{l} \mathbf{W^1} :: \ \mathrm{h}_1^1(\ldots) \ \mathbf{:-} \ \mathbf{W_1^1} : \mathrm{b}_1^1(\ldots), \ \ldots, \mathbf{W_j^1} : \mathrm{b}_i^1(\ldots). \\ \mathbf{W^2} :: \ \mathrm{h}_1^2(\ldots) \ \mathbf{:-} \ \mathbf{W_1^2} : \mathrm{b}_1^2(\ldots), \ \ldots, \mathbf{W_k^2} : \mathrm{b}_j^2(\ldots). \\ \ldots \\ \mathbf{W^n} :: \ \mathrm{h}_p^q(\ldots) \ \mathbf{:-} \ \mathbf{W_1^n} : \mathrm{b}_1^n(\ldots), \ \ldots, \mathbf{W_l^n} : \mathrm{b}_k^n(\ldots). \end{array}$$

where h_i^j's and b_i^j's are predicates forming positive, not necessarily different, literals, and $\mathbf{W_i^j}$'s are the associated tensors (also possibly reused in different places). Note that the template does not have to encode a particular model or knowledge about the problem. Instead, it can merely encode a generic mode of computation such as, for instance, the GNN propagation scheme.

Example 1 ***(GNN)*** *Consider a simple template for learning with molecules, encoding a generic idea that the representation ($h(.)$) of a chemical atom (e.g. $a(o_1)$) is dependent on the atoms adjacent to it. Given that a molecule can be represented by the set of contained atoms (e.g. $a(h_1)$) and bonds between them (e.g. $b(h_1, o_1)$), we can encode this idea by a following rule (the gnn rule):*

$$\mathbf{W_{h_1}} :: h(X) \ \mathbf{:-} \ \mathbf{W_a} : a(Y), \mathbf{W_b} : b(X, Y).$$

where X, Y are free variables. Moreover, one might be interested in using the representation of all atoms ($h(X)$) for deducing the representation of the whole molecule (q), for which we can write

$$\mathbf{W_q} :: \ q \ \mathbf{:-} \ \mathbf{W_{h_2}} : h(X).$$

2.3 Computation Graphs Defined by LRNNs

Here we briefly outline the mapping from a learning template $\mathcal{T}$ and example E_l onto a (differentiable) computation graph $\mathcal{G}_l$. For that, we take $\mathcal{N}_l = \mathcal{T} \cup E_l$ and construct the least Herbrand model $\overline{\mathcal{N}_l}$ of $\mathcal{N}_l$, which can be done using standard theorem proving techniques [2]. Next we project the derived logical constructs onto specific node types in the computation graph $\mathcal{G}_l$, the structure of which, broadly speaking, reflects the structure of the derived proof paths. An overview of the node types and their correspondence to common GNN terminology is in Tab 1. For further details we refer to [8].

Example 2 ***(GNN cont'd)*** *Let us extend the template from Ex. 1 with descriptions of two example molecules of hydrogen and water. The derived computation graphs are then displayed in Fig. 1.*

GNN terminology	Logical constructs	Type of node	Notation
Input data	Ground fact h	Fact node	$F_{(h,\vec{w})}$
Convolution	Ground rule's $\alpha\theta$ body	Rule node	$R^{c\theta}_{(W^c_0 c\theta \leftarrow W^\alpha_1 b_1\theta \wedge \cdots \wedge W^\alpha_k b_k\theta)}$
Pooling	Rule's α ground head h	Aggregation node	$G^{h=c\theta_i}_{(W^c_0 c \leftarrow W^\alpha_1 b_1 \wedge \cdots \wedge W^\alpha_k b_k)}$
Combination	Ground atom h	Atom node	A_h

Table 1: Correspondence between the common GNN terminology and LRNN transformation of a logical ground (Herbrand) model into a computation graph.

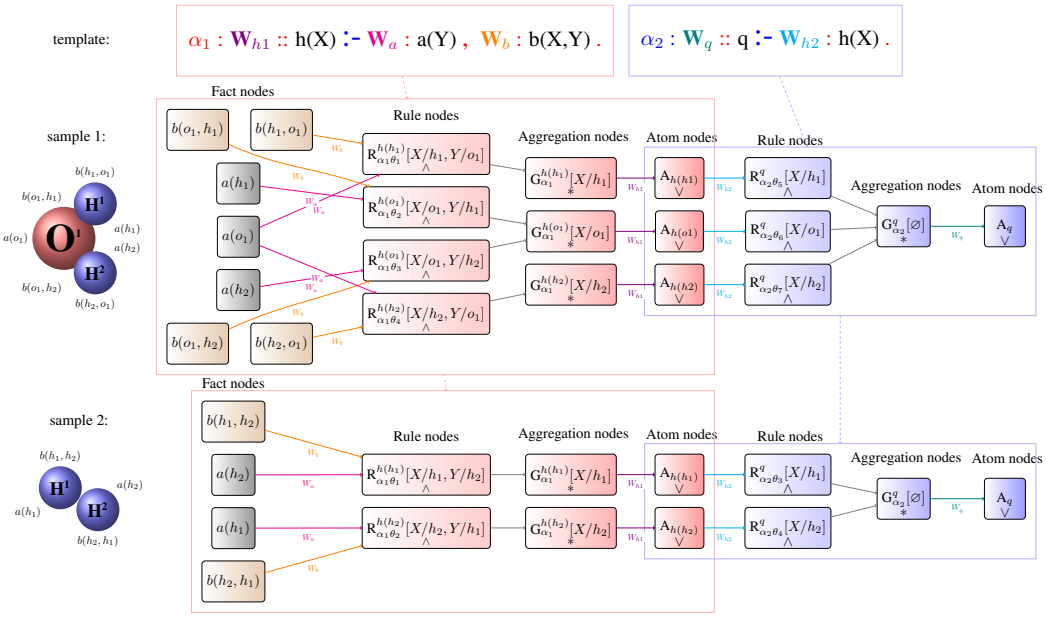

Figure 1: A simple LRNN template with 2 rules described in Ex. 1. Upon receiving 2 molecules, 2 neural computation graphs get created, as prescribed by the LRNN semantics (Sec. 2.3).

3 Extending GNNs with Rings

We have introduced how standard GNNs can be easily encoded in LRNNs in Ex. 1. Note that the example templates discussed in this paper are actual code that can be run very efficiently. For a more detailed description and comparison with existing GNN frameworks we refer to [9]. Now we provide a short demonstration on how to reach beyond GNNs with the molecular rings. Using the language of relational logic, a ring can be easily defined as a crisp pattern based on the existing bonds as

$$\texttt{\_ring}_6(A,\ldots,F) \texttt{ :- } \texttt{bond}(A,B),\ \ldots,\ \texttt{bond}(E,F),\ \texttt{bond}(F,A).$$

A distributed representation of a ring can then be declared by aggregating all the contained atoms:

$$\mathbf{W_r}\texttt{::ring}_6^{(n)}(A,\ldots,F) \texttt{ :- } \texttt{\_ring}_6(A,\ldots,F),\ \mathbf{W_a}\texttt{:atom}^{(n)}(A),\ \ldots,\ \mathbf{W_f}\texttt{:atom}^{(n)}(F).$$

These can be further stacked hierarchically, replacing the crisp $\texttt{\_ring}_6$ pattern by the respective distributed representation $\texttt{ring}_6^{(n-1)}$ from the preceding layer. Similarly, we can then also propagate the representation from the ring back into all the contained atoms by simply adding the following rule

$$\mathbf{W_{a'}}\texttt{::atom}^{(n)}(A) \texttt{ :- } \mathbf{W_{r'}}\texttt{: ring}_6^{(n-1)}(A,\ldots,F).$$

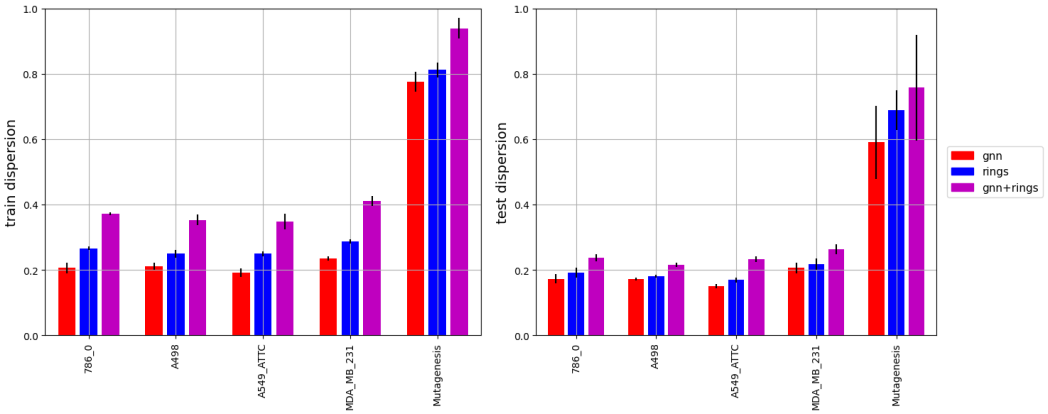

Figure 2: Comparison of the gnn, $rings$, and the joint $gnn + rings$ learners across several molecule classification datasets w.r.t. training (left) and testing (right) dispersion (higher is better).

Note that each ring pattern will be automatically instantiated in all possible revolutions, and so it is enough to specify propagation into the "first" atom (A) only. We further refer to this template as $rings$. We note that it is a good practice to add some basic transformation to produce at least some output should a molecule contain no rings at all, e.g. simply aggregating all pairs of connected atoms:

```
W_q::molecule :- W_1:atom(A), W_2:atom(B), bond(A,B).
```

Naturally, this can be further combined with the standard GNN propagation scheme (Ex. 1), where we aggregate the direct neighbor representations:

```
W_h::atom^(n)(X) :- W_1':atom^(n-1)(Y), bond(X,Y).
```

In each layer n, the representation of an atom will thus get updated based on the neighbors as well as all the atoms occupying the same rings. The particular contributions to the representation update are then parameterized separately. We further refer to this joint template as $rings + gnn$.

3.1 Experiments

For the experiments we encoded both 5 and 6 rings, added 3 layers for both the gnn and rings embedding propagation, and set all the learnable tensor to randomly initialized 3×3 matrices. We then evaluated the architectures across several datasets for molecule classification [6, 4, 3], ranging from 188 to app. 3500 molecules. We note we only used the basic information on atom types and their conformations. We left all the hyperparameters of the framework on their defaults, learning with $tanh$ activations, avg aggregations, and $ADAM$ optimizer, trained for 2000 steps, and evaluated with a 5-fold cross-validation. The results are displayed in Fig. 2. We can see that the $rings$ template indeed performs better than the basic gnn learner, while both benefit from their mutual combination.

4 Discussion

We have demonstrated a deep relational learning framework of LRNNs [8] on a small learning scenario designed to extend the basic GNN idea to capture molecular rings. Note that all we had to do was to declare a couple of simple rules specifying (i) what a ring is and (ii) how to update representation of its contained atoms. The derivation of the particular dynamic computation graphs, encoding the respective propagation scheme for each of the differently structured molecules, the corresponding gradients, training and evaluation were then all performed completely automatically. This allows for quick prototyping of diverse relational modelling ideas, where one can stack complex structural patterns to be trained end-to-end with gradient descend. In the particular showcase just demonstrated, we were then able to quickly evaluate the contribution of atom rings to basic GNNs.

Acknowledgments

This work was supported by project no. 20-19104Y granted by the Czech Science Foundation.

References

[1] Ivan Bratko. *Prolog programming for artificial intelligence*. Pearson education, 2001.

[2] Jean H Gallier. *Logic for computer science: foundations of automatic theorem proving*. Courier Dover Publications, 2015.

[3] Christoph Helma, Ross D. King, Stefan Kramer, and Ashwin Srinivasan. The predictive toxicology challenge 2000–2001. *Bioinformatics*, 17(1):107–108, 2001.

[4] Huma Lodhi and Stephen Muggleton. Is mutagenesis still challenging. *ILP-Late-Breaking Papers*, 35, 2005.

[5] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4602–4609, 2019.

[6] Liva Ralaivola, Sanjay J. Swamidass, Hiroto Saigo, and Pierre Baldi. Graph kernels for chemical informatics. *Neural Netw.*, 18(8):1093–1110, 2005.

[7] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

[8] Gustav Sourek, Vojtech Aschenbrenner, Filip Zelezny, Steven Schockaert, and Ondrej Kuzelka. Lifted relational neural networks: Efficient learning of latent relational structures. *Journal of Artificial Intelligence Research*, 62:69–100, 2018.

[9] Gustav Sourek, Filip Zelezny, and Ondrej Kuzelka. Beyond graph neural networks with lifted relational neural networks. *arXiv preprint arXiv:2007.06286*, 2020.

[10] Jeffrey D Unman. Principles of database and knowledge-base systems. *Computer Science Press*, 1989.

[11] Boris Weisfeiler. *On construction and identification of graphs*, volume 558. Springer, 2006.

[12] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[13] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[14] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*, 2018.