Learning to Generate Molecules From Small Datasets Using Neural Markov Logic Networks

Martin Svatoš[1], Peter Jung[1], Filip Železný[1], Giuseppe Marra[2], and Ondřej Kuželka[1]

[1] CTU in Prague, Czech Republic
{svatoma1,jungpete,zelezny,kuzelon2}@fel.cvut.cz
[2] KU Leuven, Belgium giuseppe.marra@kuleuven.be

Abstract. Neural Markov Logic networks are a statistical relational model capable of generating relational structures. In this paper, we investigate how this particular model behaves in the setup of few-shot learning and show that Neural Markov Logic Networks are able to learn to generate small molecules from a handful of training examples without any pre-training.

Keywords: relation data · NMLNs · few-shot learning · generative model

1 Introduction

Drug discovery [1] is an active research field in which one typically uses a huge molecular dataset to learn a model for generating new, unseen, molecules with desired properties [2]. Few-shot learning [3] aims to learn a model using only a handful of training samples. Usually, this is done by adding some additional knowledge into the process, e.g., by pre-training. Motivated by the drug discovery task, in this paper, we investigate the problem of learning a generator of molecules in a few-shot setup. We show that Neural Markov Logic networks (NMLN) [4] are able to learn such a generator of molecules which is able to generate molecules from a held-out test set while using only a small training dataset and without adding any additional knowledge into the model, e.g. pre-training.

2 Neural Markov Logic Networks

Markov logic networks [7] (MLNs) are an exponential-family model for modelling distributions over possible worlds. MLNs work by learning parameters of a relational potential function over fragments of possible worlds, which will henceforth also be called simply *training examples*. Specifically, a *fragment*, in this context, is a subset of ground atoms from a possible world restricted to a given set of constants. NMLNs were designed to exploit symmetries which occur in the data, e.g. molecular data, hence they learn on all fragments of given possible worlds which are induced by size-k subsets of constants.

The probability of a possible world ω is modelled by an NMLN as:

$$P(\omega) = \frac{\exp \sum_i \beta_i \Phi_i(\omega; \mathbf{w}_i)}{\sum_{\omega' \in \Omega_{\mathcal{L}}} \exp \sum_i \beta_i \Phi_i(\omega'; \mathbf{w}_i)} \tag{1}$$

Here, β_i's are real-valued weights and Φ_i's are potential functions represented by neural networks parameterized by the weight vectors $\mathbf{w}$. Each neural potential function Φ_i is a sum of *local potential functions* over fragments of the possible world and the weights $\mathbf{w}$ are shared among all these local fragments (much like weights are shared in convolutional neural networks in computer vision). Due to this weight sharing scheme, NMLNs are invariant w.r.t. permutations of domain elements, which is a desirable property when learning from relational data.

3 Experiments

Data: We evaluate few-shot learning abilities of NMLNs on a case study involving molecular data from ChEMBL [5] database. An NMLN is a generative model for a specific number of constants,[3] thus we derived several datasets from the ChEMBL database, each one consisting of molecules with the same number of heavy atoms; namely, datasets of seven, eight, nine, and ten molecules. We refer to each of these datasets as *n-atoms* dataset. Finally, we split these n-atoms molecular datasets into train and test sets of equal size. The three datasets have equally sized train and test splits: 7-atoms, 8-atoms, and 9-atoms datasets have 297, 515, and 796 training samples respectively. The one remaining dataset, 10-atoms, has 1356 training and 1355 testing samples.

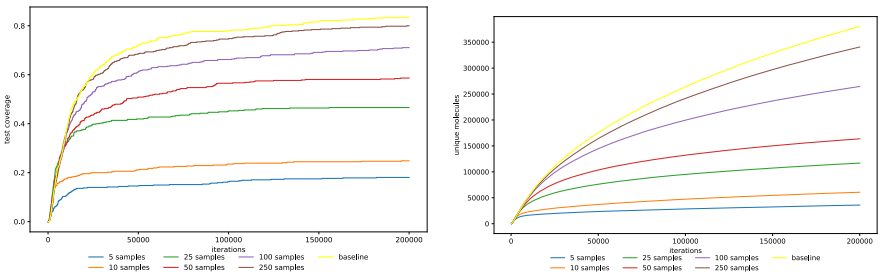
Methodology: To compare how well an NMLN works with only a handful of training examples, we first have to set up a baseline which, in this particular case, is learning on the whole training data. Furthermore, we set training set sizes of 5, 10, 25, 50, 100, 250, 500, and 1000 to see how an NMLN scores in absence of most, or only a part of the training data. On each of these training sets we learn an NMLN[4]. In the end, we compare such learned models against the baseline.

Evaluation: To evaluate how well an NMLN scores against a baseline, i.e. a model learned on a subset of training data versus all the training data, we use the criterion of test coverage, i.e. the percentage of test set samples generated by an NMLN model from the start of the learning and generation process. Although the test coverage is our main metric, we also look at the number of unique generated molecules. Furthermore, we also manually investigate most frequently generated molecules.

[3] Like MLNs, NMLNs are also not a projective model.

[4] See appendix at *http://ida.fel.cvut.cz/~svatos/nmlsfewshot.pdf* for experiments' setup.

Results: Figure 1a displays test coverage for the 8-atom dataset. With increasing size of training sets, the test coverage improves significantly. Test coverage of NMLNs learned with very small support set sizes, e.g. 5 or 25, plateaus early because the models tend to overfit; they discover some molecules from the test sets but mostly generate memorized training data. This defect fades out quickly with increasing training set size; test coverage of such NMLNs plateaus as well but with a higher score and the models generate more new, unseen, molecules, which is shown in fig. 1b.



(a) (b)

Fig. 1: Test coverage given learning step (a) and the number of unique generated molecules (b) for 8-atoms dataset.

The first row of Figure 2 shows nine randomly selected molecules for selected training sets; the second row shows nine most frequent molecules generated for these support sets by the learned NMLNs. [5]

4 Conclusion

In this paper, we investigated few-shot learning abilities of NMLNs for generation without pre-training. On the molecular case study, we showed that NMLNs can learn to generate molecules from small training datasets. However, having very tiny training sets, NMLNs tended to rather memorize training data and did not manage to generate many molecules from the held-out dataset. This quickly improved when using datasets containing more than just a handful of training examples, though. The main reason behind NMLNs success in this few-shot setting is likely their inductive bias—they are invariant w.r.t. permutations and learn regularities on the fragment level, which allows them to generalize even from very few examples.

Acknowledgements *MS and OK were supported by the Czech Science Foundation project "Generative Relational Models" (20-19104Y). MS was also supported by SGS20/178/OHK3/3T/13.*

[5] See appendix for figures of metrics and molecules of the remaining datasets.

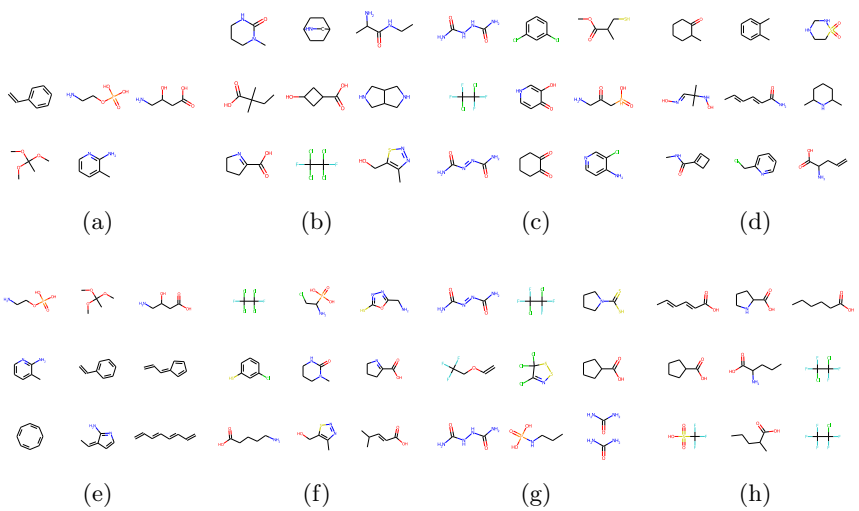


Fig. 2: A sample of train (the first row) and most frequent generated molecules (the second row) on the 8-atoms dataset for support set sizes of 5 (a, e), 25 (b, f), 100 (c, g), and the baseline (d, h).

References

1. Sliwoski, Gregory, et al. "Computational methods in drug discovery." Pharmacological reviews 66.1 (2014): 334-395.
2. Xu, Youjun, et al. "Deep learning for molecular generation." Future medicinal chemistry 11.6 (2019): 567-597.
3. Snell, Jake, Kevin Swersky, and Richard Zemel. "Prototypical networks for few-shot learning." Advances in neural information processing systems 30 (2017).
4. Marra, Giuseppe, and Ondřej Kuželka. "Neural markov logic networks." Uncertainty in Artificial Intelligence. PMLR, 2021.
5. Gaulton, Anna, et al. "The ChEMBL database in 2017." Nucleic acids research 45.D1 (2017): D945-D954.
6. Li, Yujia, et al. "Learning deep generative models of graphs." arXiv preprint arXiv:1803.03324 (2018).
7. Richardson, Matthew, and Pedro Domingos. "Markov logic networks." Machine learning 62.1 (2006): 107-136.