Simon Fraser University
Faculty of Applied Sciences
School of Computing Science

Semi-Supervised Junction Tree Variational Autoencoder for Molecular Property Prediction

CMPT 416 Final Report

Author
Tony Shen

Supervisor
Dr. Martin Ester
Atia Hamidizadeh

May 2, 2021

Semi-Supervised Junction Tree Variational Autoencoder for Molecular Property Prediction

ABSTRACT

Recent advances in machine learning have enabled accurate prediction of chemical properties. However, supervised machine learning methods in this domain often suffer from the label scarcity problem, due to the expensive nature of labelling chemical property experimentally. This research modifies state-of-the-art molecule generation method - Junction Tree Variational Autoencoder (JT-VAE) to facilitate semi-supervised learning on chemical property prediction. Furthermore, we force some latent variables to take on consistent and interpretable purposes such as representing toxicity via this partial supervision. We leverage JT-VAE architecture to learn an interpretable representation optimal for tasks ranging from molecule property prediction to conditional molecule generation, using a partially labelled dataset.

1 INTRODUCTION

1.1 Problem Motivation

One of the early challenges in the drug development pipeline is to discover non-toxic small molecules with the right properties. Scientists need to test hundreds of candidate molecules experimentally in wet lab; This process is in part why drug development is time consuming and extremely expensive. Our ultimate goal is to develop accurate In silico methods for small molecules generation and property prediction computationally, thus speeding up the drug discovery process and reducing its cost.

1.2 Addressing Label Scarcity

The task of molecular property prediction, the main focus of this paper, has often been addressed using supervised learning methods. However, experimentally labelling sufficient number of molecule's property required for supervised training is often infeasible due to the high cost and time required. To this end, transfer learning using embedding of graph neural network (GNN) pre-trained on large public chemical datasets has been proposed to mitigate label scarcity problem and improve performance for downstream tasks (Hu et al., 2020). Semi-supervised methods also aim to improve label prediction performance by leveraging both the abundant unlabelled data and the limited labelled data for training. In particular, Semi-supervised Learning with Deep Generative Models (Semi-DGM) proposed by (Kingma et al., 2014) has been designed to optimize its prediction by leveraging the generative description, the latent variable, of the entire dataset. Semi-DGM optimize the classifier jointly with a Variational Autoencoder (VAE) over the input data and achieved state-of-the-art performance on image classification with partially labelled data. Semi-supervised learning is particularly suitable for chemistry applications due to its ability to leverage unlabelled data to model the vast chemical space.

1.3 Mapping Chemicals to Vector Space

Applying VAE on chemical data requires the non-trivial task of mapping chemical structures to a meaningful continuous representation through the encoding process, and subsequently decoding the continuous representation back to a molecular graph. Morgan Fingerprint (Rogers and Hahn, 2010) is commonly used to map chemicals to a bit-vector molecular fingerprint for prediction tasks. However, it is not possible to reconstruct the original molecule structure from Morgan Fingerprint. Therefore, it is an unsuitable representation in the context of VAE. Another method is to use SMILE representation, a linear string used to describe molecular structure, in conjunction with recurrent neural network units to address the aforementioned encoding and decoding task (Gómez-Bombarelli et al., 2018). Futhermore, Grammar based VAE for SMILE (Kusner et al., 2017) has been developed to incorporate rules that enforces the model to generate valid molecules. However, SMILE representation continues to suffer from two limitations: Firstly, its linear format fail to capture the graphical nature of chemical; Secondly, similar molecules can be mapped into drastically different strings representations. These limitations create challenge in learning a continuous and meaningful latent space. To this end, Junction Tree Variational Autoencoder (JTVAE) (Jin et al., 2019) has been designed to directly encode and decode the molecular graph directly without resorting to SMILE representation. JTVAE achieved the state of the art performance on generating chemically valid and novel molecules.

1.4 Leveraging JTVAE for Semi-supervised Property Prediction

In this paper, we propose a semi-supervised version of JTVAE that is capable of predicting molecule properties training on partially labelled data. While semi-supervised learning with generative models has been previously explored in other domains, such as speech synthesis (Habib et al., 2019), state-of-the-art representation learnt from unlabelled chemical data remains largely untapped for molecular property classification. It is this gap we wish to address through the following contributions:

(1) Disentangle the property factor from the latent factor of JTVAE for label prediction.
(2) Exploit the representation learnt from JTVAE encoder to improve property prediction performance on partially labelled Tox21 dataset.

2 METHOD

2.1 Problem Statement

Given a set of labelled molecular graphs $G_l = \{G_1, G_2, \dots G_n\}$ of size n with corresponding property labels $\{y_1, y_2, \dots y_n\}$ and a set of unlabeled molecular graphs $G_u = \{G_{n+1}, G_{n+2}, \dots G_{n+m}\}$

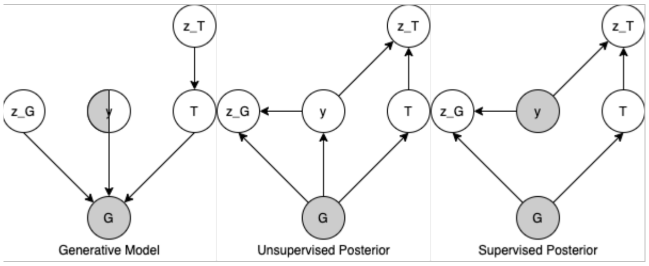

Figure 1: Left: The graphical model showing the generation dependence of random variables. Centre: The structure of the variational distribution used to approximate the posterior for supervised data points and Right: fully unobserved points

of size m, our goal is to learn a model that predicts the set of labels $\{y_{n+1}, y_{n+2}, \ldots yn + m\}$ for the unlabeled molecular graphs.

2.2 Semi-supervised Junction Tree Variational Autoencoder

Semi-JTVAE consists of three latent variables – z_G, z_T and y. Of which, the partially observed latent variable y represents a high-level molecular property, such as toxicity, that we want to make prediction about. Identical to JTVAE, there are two unobserved latent variable z_T and z_G, responsible for representing the junction tree scaffold and its cluster labels T, and cluster connection orientation required for reconstructing molecular graph G respectively. The chemical molecule is modelled to be generated through the following process:

$$p_\theta(y) = Cat(y|\pi); \quad p_\theta(z) = N(z|0, I);$$
$$p_\theta(G|z_G, T, y); \quad p_\theta(T|z_T, y);$$

As shown above, $Cat(y|\pi)$ is the multinomial class label distribution; z_T and z_G are generated from a multivariate normal distribution. Although molecular graph generation - $p_\theta(G|z_G, T, y)$ depends on junction tree T, which in turn depends on y already, a deliberate choice was made to make both molecular graph and junction tree generation process dependent on y. This is justified because some steroisomers (and constitutional isomers) pairs can have drastically different molecular properties. It is the responsibility of the graph decoder to select the right connectivity of the clusters that give rise to the isomer with the desired property. For that reason, latent variable y is also provided to graph decoder. Figure 1 shows the corresponding graphical models of our proposed Semi-JTVAE.

2.3 Molecular Property Prediction

Semi-supervised methods has the unfair advantage of learning data distribution from the abundance of unlabelled data to develop a better data representation. We use the hidden layer of Tree Encoder and Graph Encoder (H_G, H_T) as the input feature for the prediction of y, since these features likely benefit from the descriptive power learned from these partially labelled data. H_G and H_T are used to generate Z_G and Z_T subsequently. Our tree and graph encoder are jointly optimized for both encoding and property classification task. Our proposed Semi-JTVAE is modelled in Figure 2.

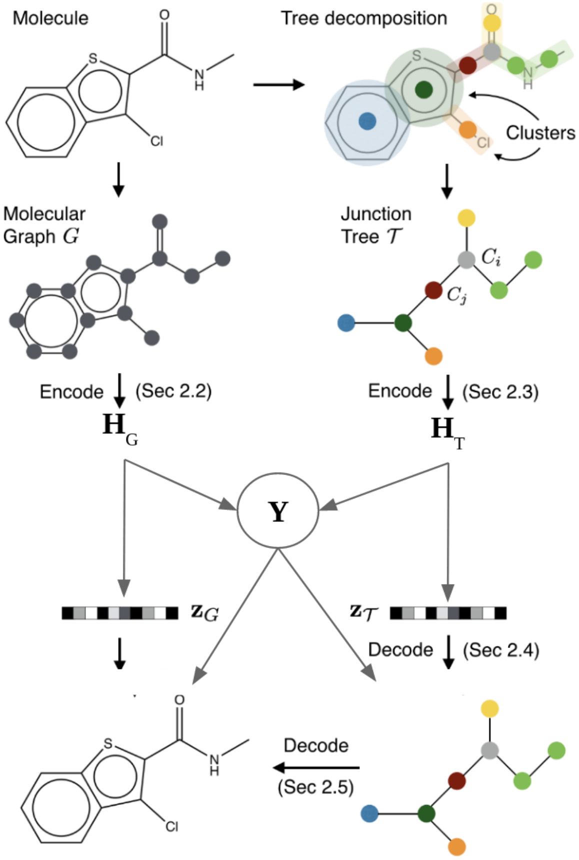

Figure 2: Overview of our modified model: Property factor y depends on hidden layer of both the tree and graph encoder H_T, H_G. We reconstruct the junction tree using both z_T and y, then assemble the molecular graph using T, z_G and y.

2.4 Semi-JTVAE Model Objective

There are two cases of model objective to consider. We maximize the expected lower bound, with the labelled dataset objective function as follows:

$$\begin{aligned}\log p_\theta(G, T, y) \geq E_{q_\phi(z_T, z_G|T,G,y)} \big[&\log p_\theta(G|T, z_G, y) + \log p_\theta(T|z_T, y) \\ &+ \log p_\theta(z_T) + \log p_\theta(z_G) + 2 * \log p_\theta(y) \\ &- \log q_\phi(z_T|T) - \log q_\phi(z_G|G) \big] \\ = -&\mathcal{L}(T, G, y)\end{aligned} \tag{1}$$

For the case where the label corresponding to a data point is unobserved, we approximate $p_\theta(z_G, z_T, y|T, G)$ using a posterior function $q_\phi(z_G, z_T, y|T, G)$ modelled by neural networks. The detailed derivation can be found in subsection 6.1. The unlabelled dataset objective is as follows:

Table 1: Frequency of Special Characters

Method	***Multi-Layer Perceptron***			Semi-Supervised VAE			***Junction Tree VAE***		
Metrics	*Accuracy*	*f1-pos*	*f1-neg*	*Accuracy*	*f1-pos*	*f1-neg*	*Accuracy*	*f1-pos*	*f1-neg*
Labelled %									
1% (N=78)	84%	**12%**	91%	84%	2%	92%	84%	0%	92%
5% (N=391)	82%	**27%**	90%	84%	19%	91%	84%	20%	91%
25% (N=1957)	86%	47%	92%	86%	44%	92%	86%	**48%**	92%
Original (N=5810)	**88%**	**54%**	93%	87%	46%	93%	84%	44%	90%

$$
\begin{aligned}
\log p_\theta(G,T) \geq E_{q_\phi(z_T,z_G,y|T,G)}\big[& \\
& \log p_\theta(G|T,z_G,y) + \log p_\theta(T|z_T,y) \\
& + \log p_\theta(z_T) + \log p_\theta(z_G) + 2 * \log p_\theta(y) \\
& - \log q_\phi(z_T|T) - \log q_\phi(z_G|G) - \log q_\phi(y|T,G)\big] \\
= & \sum_y \big[-\mathcal{L}(T,G,y)\big] + \mathcal{H}\big(q(y|T,G)\big) \\
= & -\mathcal{U}(T,G)
\end{aligned} \tag{2}
$$

As noted in Semi-DGM's original paper (Kingma et al., 2014), it is desirable to add a classification loss so the distribution $q_\phi(y|T,G)$ can be learnt with the labelled dataset. This yields our final objective function:

$$
\mathcal{J}^a = \sum_{(T,G,y)\sim\mathcal{D}} \mathcal{L}(T,G,y) + \sum_{(T,G)\sim\mathcal{D}} \mathcal{U}(T,G) + a \cdot E\big[\log q_\phi(y|T,G)\big] \tag{3}
$$

2.5 Optimization

The semi-JTVAE's encoder, classifier and decoder components are optimized jointly using the previous objective function. The following is the pseudo code of semi-JTVAE's training loop:

```
$(\phi, \theta, W_G, W_T) \leftarrow initializeModelParameters()$;
while $training()$ do
    $\mathcal{D} \leftarrow getRandomMiniBatch()$;
    $T_i \leftarrow decomposeToTree(G_i)$;
    $H_{Gi} \leftarrow encodeGraph(G_i, W_G)$;
    $H_{Ti} \leftarrow encodeTree(T_i, W_T)$;
    $y_i \sim q_\phi(y_i|H_{Gi}, H_{Ti})$;
    $z_{Ti} \sim q_\phi(z_{Ti}|H_{Ti}, y_i)$;    $z_{Gi} \sim q_\phi(z_{Gi}|H_{Gi}, y_i)$;
    $\mathcal{J}^a \leftarrow Equation\ 2.4$;
    $(g_\phi, g_\theta, g_{W_G}, g_{W_T}) \leftarrow (\frac{\partial \mathcal{J}^a}{\partial \phi}, \frac{\partial \mathcal{J}^a}{\partial \theta}, \frac{\partial \mathcal{J}^a}{\partial W_G}, \frac{\partial \mathcal{J}^a}{\partial W_T})$;
    $(\phi, \theta, W_G, W_T) \leftarrow (\phi, \theta, W_G, W_T) + \Gamma(g_\phi, g_\theta, g_{W_G}, g_{W_T})$;
end
```

Algorithm 1: Learning in JTVAE

For each iteration, a random mini batch is drawn from our dataset, either labelled or unlabelled; Molecular graphs are then decomposed into junction trees. Note that none of $decomposeToTree()$, $encodeGraph()$ and $encodeTree()$ are modified in this project. Our added classifier network - $q_\phi(y_i|H_{Gi}, H_{Ti})$ is modelled as a 2-layer feed forward neural network that takes the input from the result of tree and graph encoding. Latent variable z_T and z_G are drawn from the normal distribution generated conditioned on y and their respective hidden layer H. The model parameters' gradient on objective function is computed each random mini-batch of training samples and optimized using Adam (Kingma and Ba, 2017).

3 RESULT

We test the performance of our model on Tox21 ("Toxicology in the 21st Century"), a partially labelled structure-activity dataset containing 7831 small molecules with 12 toxicity labels. The labels are highly unbalanced with non-toxic being the predominant label. Performance metrics are generated using various percentage of labelled data (1%, 5%, 25%, 75%) in training. Multi-Layer Perceptron and Semi-Supervised VAE with 1024 bits Morgan fingerprint inputs are used as benchmarks for Junction Tree VAE's toxicity classification performance. The performance is summarized in Table 1.

We noticed that Multi-Layer Perceptron provided best performance in most cases. This could be due to the small overall dataset size of Tox21, inability of VAE to handle unbalanced dataset well, and/or the difficulty of training variational autoencoder due to its relatively complex loss functions.

4 FUTURE DIRECTIONS

4.1 M1 pre-trained embedding

In (Kingma and Ba, 2017), it was found that combining a pre-trained model $M1$'s embedding with a Semi-supervised VAE yielded much higher performance than training semi-supervised VAE alone. We believe the prediction accuracy of Semi-JTVAE will also benefit by using the embedding of a pre-trained unsupervised JTVAE model.

4.2 Generate molecule with desired properties

Semi-JTVAE disentangles the property factor of interest from the rest of the latent variable. A natural extension of the project is conditionally generating similar molecules with a different desired property by modifying only property factor and reconstructing the molecule using the decoder networks.

5 CONCLUSION

We have have introduced Semi-Supervised Junction Tree Variational Autoencoder architecture capable of learning on partially labelled molecular graphs. This model has the potential to improve performance for tasks including molecule property prediction and conditional molecule generation.

ACKNOWLEDGMENTS

This work was supported by Ester Lab. We thank Compute Canada's generous contribution of compute resources.

REFERENCES

R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 4(2):268–276, Jan 2018. ISSN 2374-7951. doi: 10.1021/acscentsci.7b00572. URL http://dx.doi.org/10.1021/acscentsci.7b00572.

R. Habib, S. Mariooryad, M. Shannon, E. Battenberg, R. Skerry-Ryan, D. Stanton, D. Kao, and T. Bagby. Semi-supervised generative modeling for controllable speech synthesis, 2019.

W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande, and J. Leskovec. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJlWWJSFDH.

W. Jin, R. Barzilay, and T. Jaakkola. Junction tree variational autoencoder for molecular graph generation, 2019.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling. Semi-supervised learning with deep generative models. *CoRR*, abs/1406.5298, 2014. URL http://arxiv.org/abs/1406.5298.

M. J. Kusner, B. Paige, and J. M. Hernández-Lobato. Grammar variational autoencoder, 2017.

D. Rogers and M. Hahn. Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, 2010. doi: 10.1021/ci100050t. URL http://dx.doi.org/10.1021/ci100050t. PMID: 20426451.

6 APPENDIX

6.1 Derivation of Semi-JTVAE Model Objective for Unlabelled Data

$$
\begin{aligned}
D_{KL}(Q||P) &= \int_{-\infty}^{\infty} q(z_T, z_G, y|T, G) \log(\frac{q(z_T, z_G, y|T, G)}{p(z_T, z_G, y|T, G)})dz \\
&= \int_{-\infty}^{\infty} q(z_T, z_G, y|T, G) \log(\frac{q(z_T, z_G, y|T, G)}{p(z_T, z_G, y, T, G)})dz + \log p(T) + \log p(G) \\
\log p(T) + \log p(G) &\geq -E_{q(z_T, z_G, y|T, G)} \left[\frac{q(z_T, z_G, y|T, G)}{p(z_T, z_G, y, T, G)} \right] \\
&= E_{q(z_T, z_G, y|T, G)} \big[\log p(G|T, z_G, y) + \log p(T|z_T, y) \\
&\quad + \log p(z_T) + \log p(z_G) + 2 * \log p(y) \\
&\quad - \log q(z_T|T) - \log q(z_G|G) - \log q(y|T, G) \big] \\
&= E_{q(y|T,G)} \Big[E_{q(z_T|T)} \big[\log p(T|z_T, y) + \log p(y) + \log(z_T) - \log q(z_T|T) \big] \\
&\quad + E_{q(z_G|G)} \big[\log p(G|T, z_G, y) + \log p(y) + \log p(z_G) - \log q(z_G|G) \big] \\
&\quad - \log q(y|T, G) \Big] \\
&= E_{q(y|T,G)} \Big[E_{q(z_T|T)} \big[\log p(T|z_T, y) \big] - D_{KL}\big(q(z_T|T)||p(z_T)\big) + \log p(y) \\
&\quad + E_{q(z_G|G)} \big[\log p(G|z_G, y, T) \big] - D_{KL}\big(q(z_G|G)||p(z_G)\big) + \log p(y) \\
&\quad - \log q(y|T, G) \Big] \\
&= \sum_y \big[q(y|T, G)[-TreeLoss - GraphLoss - KL_{Tree} - KL_{Graph} + 2\log(y)] \big] + \mathcal{H}\big(q(y|T, G)\big) \\
&= \sum_y \big[-\mathcal{L}(T, G, y) \big] + \mathcal{H}\big(q(y|T, G)\big)
\end{aligned}
\tag{4}
$$