Variational Graph Recurrent Neural Networks

Ehsan Hajiramezanali* † Arman Hasanzadeh* †
Nick Duffield* Krishna Narayanan* Mingyuan Zhou** Xiaoning Qian*

*Department of Electrical and Computer Engineering, Texas A&M University († equal contribution)
**McCombs School of Business, University of Texas at Austin

Introduction

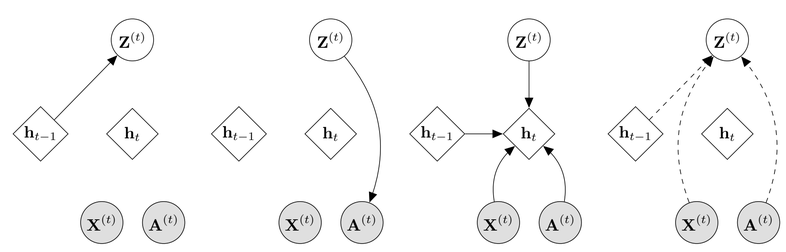
Variational graph recurrent neural network (VGRNN) by adopting high-level latent random variables in GRNN has been proposed to

- achieve more interpretable latent representations for dynamic graphs;
- model uncertainty of node latent representation;
- capture both topology and node attribute changes simultaneously.

To further boost the expressive power and interpretability of our new VGRNN method, we integrate semi-implicit variational inference with VGRNN. The **semi-implicit VGRNN (SI-VGRNN)** is capable of inferring more flexible and complex posteriors.

Methods

VGRNN Architecture



(a) Prior (b) Generation (c) Recurrence (d) Inference

(a) Conditional **prior** distribution

$$p\left(\mathbf{Z}^{(t)}\right)=\prod_{i=1}^{N} p\left(\mathbf{Z}_i^{(t)}\right);\ \mathbf{Z}_i^{(t)} \sim \mathcal{N}\left(\boldsymbol{\mu}_{i,\text{prior}}^{(t)}, \text{diag}((\boldsymbol{\sigma}_{i,\text{prior}}^{(t)})^2)\right),\ \left\{\boldsymbol{\mu}_{\text{prior}}^{(t)}, \boldsymbol{\sigma}_{\text{prior}}^{(t)}\right\}=\varphi^{\text{prior}}(\mathbf{h}_{t-1}),$$

(b) **Generating** distribution

$$\mathbf{A}^{(t)} \mid \mathbf{Z}^{(t)} \sim \text{Bernoulli}\left(\pi^{(t)}\right), \quad \pi^{(t)}=\varphi^{\text{dec}}\left(\mathbf{Z}^{(t)}\right),$$

(c) **Recurrence** equation

$$\mathbf{h}_t = f\left(\mathbf{A}^{(t)}, \varphi^{\mathbf{x}}\left(\mathbf{X}^{(t)}\right), \varphi^{\mathbf{z}}\left(\mathbf{Z}^{(t)}\right), \mathbf{h}_{t-1}\right),$$

(d) **Inference** of the posterior distribution

$$q\left(\mathbf{Z}^{(t)} \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}\right)=\prod_{i=1}^{N} q\left(\mathbf{Z}_i^{(t)} \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}\right)=\prod_{i=1}^{N} \mathcal{N}\left(\boldsymbol{\mu}_{i,\text{enc}}^{(t)}, \text{diag}((\boldsymbol{\sigma}_{i,\text{enc}}^{(t)})^2)\right),$$

$$\boldsymbol{\mu}_{\text{enc}}^{(t)}=\text{GNN}_{\mu}\left(\mathbf{A}^{(t)}, \text{CONCAT}\left(\varphi^{\mathbf{x}}\left(\mathbf{X}^{(t)}\right), \mathbf{h}_{t-1}\right)\right),$$

$$\boldsymbol{\sigma}_{\text{enc}}^{(t)}=\text{GNN}_{\sigma}\left(\mathbf{A}^{(t)}, \text{CONCAT}\left(\varphi^{\mathbf{x}}\left(\mathbf{X}^{(t)}\right), \mathbf{h}_{t-1}\right)\right),$$

Learning

- The objective function of VGRNN is derived from the variational lower bound at each snapshot

$$\mathcal{L}=\sum_{t=1}^{T}\Big\{\mathbb{E}_{\mathbf{Z}^{(t)} \sim q(\mathbf{Z}^{(t)} \mid \mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)}, \mathbf{Z}^{(<t)})} \log p\left(\mathbf{A}^{(t)} \mid \mathbf{Z}^{(t)}\right) - \mathbf{KL}\Big(q\left(\mathbf{Z}^{(t)} \mid \mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)}, \mathbf{Z}^{(<t)}\right) \| p\left(\mathbf{Z}^{(t)} \mid \mathbf{A}^{(<t)}, \mathbf{X}^{(<t)}, \mathbf{Z}^{(<t)}\right)\Big)\Big\}.$$

- The inner-product decoder is adopted in VGRNN

$$p\left(\mathbf{A}^{(t)} \mid \mathbf{Z}^{(t)}\right)=\prod_{i=1}^{N}\prod_{j=1}^{N} p\left((A_{i,j}^{(t)} \mid \mathbf{z}_i^{(t)}, \mathbf{z}_j^{(t)}\right); p\left(A_{i,j}^{(t)}=1 \mid \mathbf{z}_i^{(t)}, \mathbf{z}_j^{(t)}\right)=\text{sigmoid}\left(\mathbf{z}_i^{(t)}(\mathbf{z}_j^{(t)})^T\right),$$

Semi-implicit VGRNN (SI-VGRNN)

We impose a mixing distributions on the variational distribution parameters to further increase the expressive power of the variational posterior through a semi-implicit hierarchical construction

$$\mathbf{Z}^{(t)} \sim q(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t), \qquad \boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \mid \mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)}, \mathbf{Z}^{(<t)})=q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}).$$

$$\boldsymbol{\ell}_j^{(t)}=\text{GNN}_j(\mathbf{A}^{(t)}, \text{CONCAT}(\mathbf{h}_{t-1}, \boldsymbol{\epsilon}_j^{(t)}, \boldsymbol{\ell}_{j-1}^{(t)})); \ \boldsymbol{\epsilon}_j^{(t)} \sim q_j(\boldsymbol{\epsilon}) \text{ for } j=1, \ldots, L, \ \boldsymbol{\ell}_0^{(t)}=\varphi_\tau^{\mathbf{x}}\left(\mathbf{X}^{(t)}\right)$$

$$\boldsymbol{\mu}_{\text{enc}}^{(t)}(\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})=\text{GNN}_{\mu}(\mathbf{A}^{(t)}, \boldsymbol{\ell}_L^{(t)}), \quad \boldsymbol{\Sigma}_{\text{enc}}^{(t)}(\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})=\text{GNN}_{\Sigma}(\mathbf{A}^{(t)}, \boldsymbol{\ell}_L^{(t)}),$$

$$q(\mathbf{Z}_i^{(t)} \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}, \boldsymbol{\mu}_{i,\text{enc}}^{(t)}, \boldsymbol{\Sigma}_{i,\text{enc}}^{(t)})=\mathcal{N}(\boldsymbol{\mu}_{i,\text{enc}}^{(t)}(\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1}), \boldsymbol{\Sigma}_{i,\text{enc}}^{(t)}(\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})),$$

Learning

$$\mathcal{L}=\sum_{t=1}^{T}\Big\{\mathbb{E}_{\boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} \mathbb{E}_{\mathbf{Z}^{(t)} \sim q(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t)} \log\left(p(\mathbf{A}^{(t)} \mid \mathbf{Z}^{(t)}, \mathbf{h}_{t-1})\right) - \mathbf{KL}\Big(\mathbb{E}_{\boldsymbol{\psi}_t \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi}_t \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})} q\left(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t\right) \| p(\mathbf{Z}^{(t)} \mid \mathbf{h}_{t-1})\Big)\Big\}.$$

Results

- Given partially observed snapshots of a dynamic graph with node attributes $\{G^{(1)}, \ldots, G^{(T)}\}$, dynamic link prediction problems are defined as follows:

Dynamic link detection Detect unobserved edges in $G^{(T)}$

Metrics	Methods	Enron	COLAB	Facebook	Social Evo.	HEP-TH	Cora
AUC	VGAE	88.26 ± 1.33	70.49 ± 6.46	80.37 ± 0.12	79.85 ± 0.85	79.31 ± 1.97	87.60 ± 0.54
	DynAE	84.06 ± 3.30	66.83 ± 2.62	60.71 ± 1.05	71.41 ± 0.66	63.94 ± 0.18	53.71 ± 0.48
	DynRNN	77.74 ± 5.31	68.01 ± 5.50	69.77 ± 2.01	74.13 ± 1.74	72.39 ± 0.63	76.09 ± 0.97
	DynAERNN	91.71 ± 0.94	77.38 ± 3.84	81.71 ± 1.51	78.67 ± 1.07	82.01 ± 0.49	74.35 ± 0.85
	GRNN	91.09 ± 0.67	86.40 ± 1.48	85.60 ± 0.59	78.27 ± 0.47	89.00 ± 0.46	91.35 ± 0.21
	VGRNN	**94.41 ± 0.73**	**88.67 ± 1.57**	**88.00 ± 0.57**	**82.69 ± 0.55**	**91.12 ± 0.71**	**92.08 ± 0.35**
	SI-VGRNN	**95.03 ± 1.07**	**89.15± 1.31**	**88.12 ± 0.83**	**83.36 ± 0.53**	**91.05 ± 0.92**	**94.07 ± 0.44**
AP	VGAE	89.95 ± 1.45	73.08 ± 5.70	79.80 ± 0.22	79.41 ± 1.12	81.05 ± 1.53	89.61 ± 0.87
	DynAE	86.30 ± 2.43	67.92 ± 2.43	60.83 ± 0.94	70.18 ± 1.98	63.87 ± 0.21	53.84 ± 0.51
	DynRNN	81.85 ± 4.44	73.12 ± 3.15	70.63 ± 1.75	72.15 ± 2.30	74.12 ± 0.75	76.54 ± 0.66
	DynAERNN	93.16 ± 0.88	83.02 ± 2.59	83.36 ± 1.83	77.41 ± 1.47	85.57 ± 0.93	79.34 ± 0.77
	GRNN	93.47 ± 0.35	88.21 ± 1.35	84.77 ± 0.62	76.93± 0.35	89.50 ± 0.42	91.37 ± 0.27
	VGRNN	**95.17 ± 0.41**	**89.74 ± 1.31**	**87.32 ± 0.60**	**81.41 ± 0.53**	**91.35 ± 0.77**	**92.92 ± 0.28**
	SI-VGRNN	**96.31 ± 0.72**	**89.90 ± 1.06**	**87.69 ± 0.92**	**83.20± 0.57**	**91.42 ± 0.86**	**94.44 ± 0.52**

Dynamic link prediction

Predict edges in $G^{(T+1)}$

Metrics	Methods	Enron	COLAB	Facebook	Social Evo.
AUC	DynAE	74.22 ± 0.74	63.14 ± 1.30	56.06 ± 0.29	65.50 ± 1.66
	DynRNN	86.41 ± 1.36	75.7 ± 1.09	73.18 ± 0.60	71.37 ± 0.72
	DynAERNN	87.43 ± 1.19	76.06 ± 1.08	76.02 ± 0.88	73.47 ± 0.49
	VGRNN	**93.10 ± 0.57**	**85.95 ± 0.49**	**89.47 ± 0.37**	**77.54 ± 1.04**
	SI-VGRNN	**93.93 ± 1.03**	**85.45 ± 0.91**	**90.94 ± 0.37**	**77.84 ± 0.79**
AP	DynAE	76.00 ± 0.77	64.02 ± 1.08	56.04 ± 0.37	63.66 ± 2.27
	DynRNN	85.61 ± 1.46	78.95 ± 1.55	75.88 ± 0.42	69.02 ± 1.71
	DynAERNN	89.37 ± 1.17	81.84 ± 0.89	78.55 ± 0.73	71.79 ± 0.81
	VGRNN	**93.29 ± 0.69**	**87.77 ± 0.79**	**89.04 ± 0.33**	**77.03 ± 0.83**
	SI-VGRNN	**94.44 ± 0.85**	**88.36 ± 0.73**	**90.19 ± 0.27**	**77.40 ± 0.43**

Dynamic new link prediction

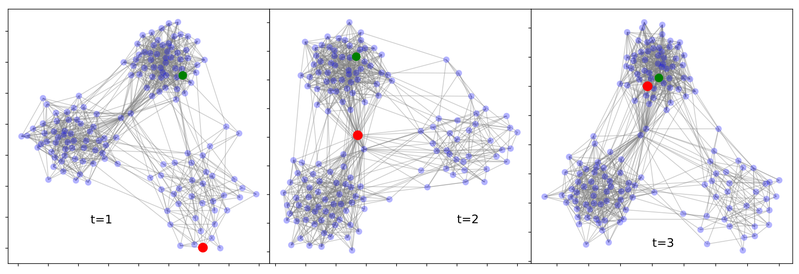
Predict edges in $G^{(T+1)}$ that are not in $G^{(T)}$

Metrics	Methods	Enron	COLAB	Facebook	Social Evo.
AUC	DynAE	66.10 ± 0.71	58.14 ± 1.16	54.62 ± 0.22	55.25 ± 1.34
	DynRNN	83.20 ± 1.01	71.71 ± 0.73	73.32 ± 0.60	65.69 ± 3.11
	DynAERNN	83.77 ± 1.65	71.99 ± 1.04	76.35 ± 0.50	66.61 ± 2.18
	VGRNN	**88.43 ± 0.75**	**77.09 ± 0.23**	**87.20 ± 0.43**	**75.00 ± 0.97**
	SI-VGRNN	**88.60 ± 0.95**	**77.95 ± 0.41**	**87.74 ± 0.53**	**76.45 ± 1.19**
AP	DynAE	66.50 ± 1.12	58.82 ± 1.06	54.57 ± 0.20	54.05 ± 1.63
	DynRNN	80.96 ± 1.37	75.34 ± 0.67	75.52 ± 0.50	63.47 ± 2.70
	DynAERNN	85.16 ± 1.04	77.68 ± 0.66	78.70 ± 0.44	65.03 ± 1.74
	VGRNN	**87.57 ± 0.57**	**79.63 ± 0.94**	**86.30 ± 0.29**	**73.48 ± 1.11**
	SI-VGRNN	**87.88 ± 0.84**	**81.26 ± 0.38**	**86.72 ± 0.54**	**73.85 ± 1.33**

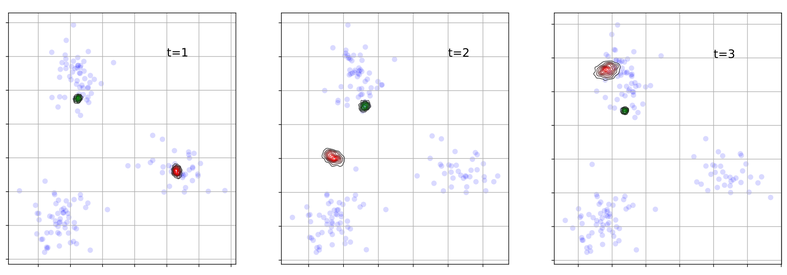
- GRNN outperforms DynAERNN due to the superior capability of GCN in capturing graph topology compared to fully connected layers
- (SI-)VGRNN compared to GRNN and DynAERNN shows that latent random variables carry more information than deterministic hidden states specially for dynamic graphs with complex temporal changes
- Since GRNN is trained as an autoencoder, it cannot predict edges in the next snapshot.
- In (SI-)VGRNN, the prior construction based on previous time steps allows us to predict links in the future.
- The proposed methods improve new link prediction substantially which shows that they can capture temporal trends better than the competing methods.
- Comparing VGRNN with SI-VGRNN shows that the prediction results are almost the same for all datasets. The reason is that although the posterior is more flexible in SI-VGRNN, the prior on which our predictions are based, is still Gaussian, hence the improvement is marginal.

Interpretable Latent Representation

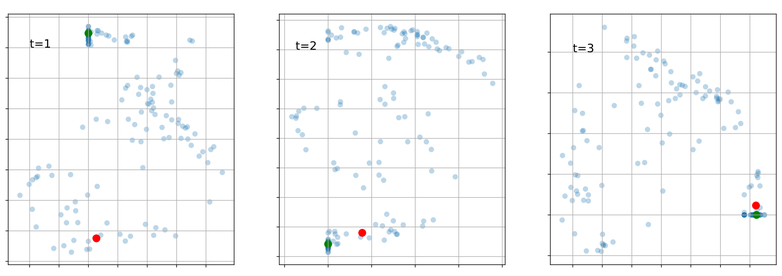
- To show that VGRNN learns more interpretable latent representations, we simulated a dynamic graph with three communities in which a node (red) transfers from one community into another in two time steps.



We embedded the node into 2-d latent space using VGRNN



and DynAERNN



- Uncertainty is directly related to structural evolution of nodes in dynamic graphs.
- The variance of the latent variables for the desired node increases in time (left to right; red contour).
- The variance of a node whose community doesn't change in time (green contour) does not increase over time.
- We argue that the uncertainty helps to better encode non-smooth evolution, in particular abrupt changes, in dynamic graphs.
- VGRNN separates the communities in the latent space more distinctively than DynAERNN.

Conclusion

- We proposed (SI-)VGRNN, the first node embedding methods for dynamic graphs that embed each node to a random vector in latent space.
- We argue that adding high level latent variables to GRNN not only increases its expressiveness to better model the complex dynamics of graphs, but also generates interpretable random latent representation for nodes.
- SI-VGRNN is developed by combining VGRNN and semi-implicit variational inference to achieve flexible non-Gaussian latent representations.
- We tested our proposed methods on dynamic link prediction tasks and they outperform competing methods substantially, specially for very sparse graphs.