Review on Dynamic Graph Learning Representation

Yue Xiang

2021.3.19

Outlines

RNN based models

- Streaming Graph Neural Networks
  Ma, Yao, et al. "Streaming graph neural networks." Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 2020.
- JODIE
  Kumar, Srijan, Xikun Zhang, and Jure Leskovec. "Predicting dynamic embedding trajectory in temporal interaction networks." Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2019.

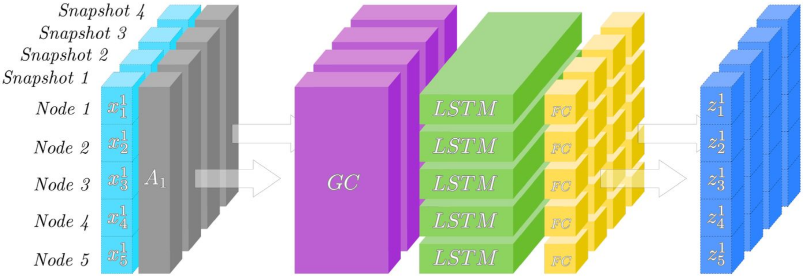
Temporal point process (TPP) based models

- DyRep
  Trivedi, Rakshit, et al. ”Dyrep: Learning representations over dynamic graphs.” International Conference on Learning Representations. 2019.
- Graph Hawkes Network
  Han, Zhen, et al. ”The graph hawkes network for reasoning on temporal knowledge graphs.” arXiv preprint arXiv:2003.13432. 2020.

Ovewview

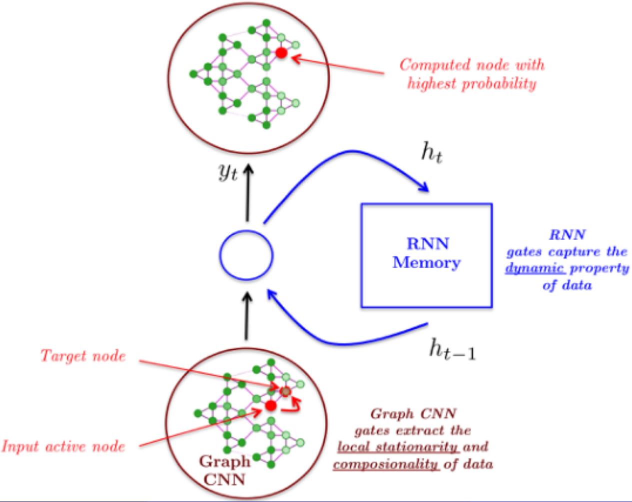
- Stacked DGNNs: GNN+RNN/LSTM/GRU/...
- Integrated DGNNs: combine GNNs and RNNs in one layer(combine modeling of the spatial and the temporal domain in one layer)
- Dynamic graph autoencoders and generative models

Stacked DGNNs

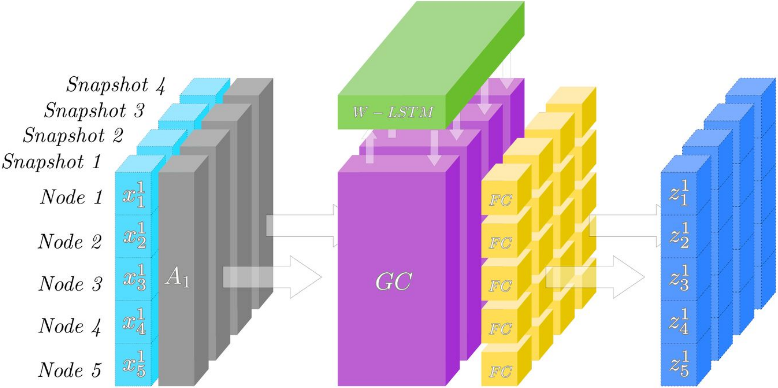


Stacked DGNNs: GCRN

Seo, Youngjoo, et al. "Structured sequence modeling with graph convolutional recurrent networks." International Conference on Neural Information Processing. Springer, Cham, 2018.



Integrated DGNNs



Integrated DGNNs: GCRN

Given a graph G with N nodes, whose topology is determined by the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, and a sequence of node attributes $\mathcal{X} = \left\{\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots, \mathbf{X}^{(T)}\right\}$, GCRN reads M-dimensional node attributes $\mathbf{X}^{(t)} \in \mathbb{R}^{N \times M}$ and updates its hidden state $\mathbf{h}_t \in \mathbb{R}^p$ at each time step t

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

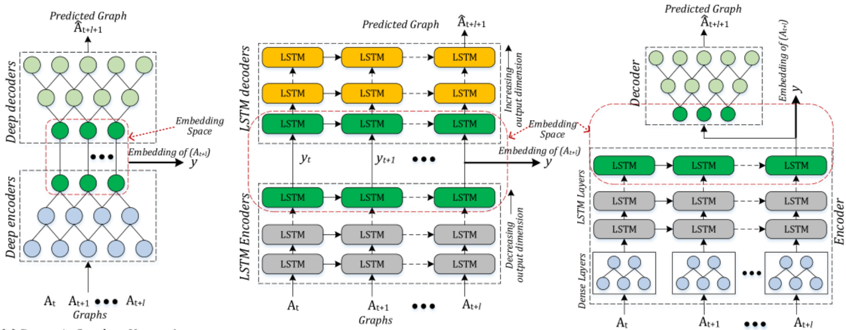
f: a non-probabilistic deep neural network(deep layers inside LSTM/...are replaced by graph convolutional layers)

$$p\left(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots, \mathbf{X}^{(T)} \mid \mathbf{A}\right) = \prod_{t=1}^{T} p\left(\mathbf{X}^{(t)} \mid \mathbf{X}^{(<t)}, \mathbf{A}\right); \quad p\left(\mathbf{X}^{(t)} \mid \mathbf{X}^{(<t)}, \mathbf{A}\right) = g\left(\mathbf{A}, \mathbf{h}_{t-1}\right)$$

Dynamic Graph Autoencoders and Generative Models

- DynamicGEM
- VGRNN
- Dyngraph2vec
- GCN-GAN
- Dyngraphgan

Goyal, Palash, Sujit Rokka Chhetri, and Arquimedes Canedo. "dyngraph2vec: Capturing network dynamics using dynamic graph representation learning." Knowledge-Based Systems 187 (2020): 104816.



(a) Dynamic Graph to Vector Auto Encoder (**dyngraph2vecAE**)

(b) Dynamic Graph to Vector Recurrent Neural Network (**dyngraph2vecRNN**)

(c) Dynamic Graph to Vector Autoenncoder Recurrent Neural Network (**dyngraph2vecAERNN**)

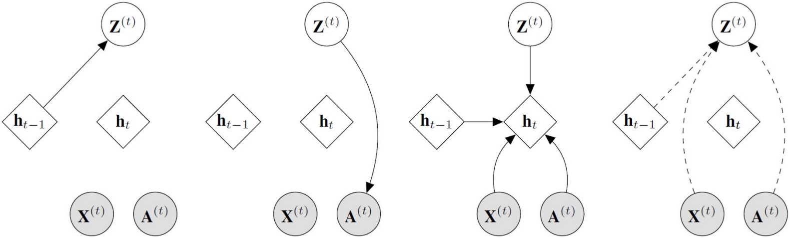
Figure 4: dyngraph2vec architecture variations for dynamic graph embedding.

VGRNN: VGAE+GRNN

Hasanzadeh, Arman, et al. "Variational Graph Recurrent Neural Networks." Advances in neural information processing systems 32 (2019).

- GCRN: aim to model dynamic node attributes defined over a static graph.
- GRNN: get different adjacency matrices at different time snapshots and reconstruct the graph at time t by adopting an inner-product decoder on the hidden state h_t.
- To further improve the expressive power of GRNN, we introduce stochastic latent variables by combining GRNN with variational graph autoencoder (VGAE)

VGRNN



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

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

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

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

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

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

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

VGRNN: Generation

In (b), φ^{dec} is inner-product decoder. Potenti extensions with other decoders can be integrated with VGRNN if necessary.

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

where $\mathbf{z}_i^{(t)}$ corresponds to the embedding representation of node $v_i^{(t)} \in \mathcal{V}^{(t)}$ at time step t.
The generating distribution can also be conditioned on $\mathbf{h}_{t-1}$ if we want to generate $\mathbf{X}^{(t)}$ in addition to the adjacency matrix for other applications. In such cases, φ^{dec} should be a highly flexible neural network instead of a simple inner-product function.

VGRNN: Learning

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

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

Semi-implicit VGRNN (SI-VGRNN)

SI-VGRNN imposes a mixing distributions on the variational distribution parameters to model the posterior of VGRNN with a semi-implicit hierarchical construction:

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

While the variational distribution $q\left(\mathbf{Z}^{(t)} \mid \boldsymbol{\psi}_t\right)$ is required to be explicit, the mixing distribution, $q_{\boldsymbol{\phi}}$, is not subject to such a constraint, leading to considerably flexible $\mathbb{E}_{\boldsymbol{\psi}_t \sim q_\phi(\boldsymbol{\psi}_t \mid \mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}_{t-1})}\left(q\left(\mathbf{z}_t \mid \dot{\boldsymbol{\psi}}_t\right)\right)$)

Deep Encoders for Dynamic Network Typology

Model type	Model name	Encoder	Link addition	Link deletion	Node addition	Node deletion	Network type
Discrete networks							
Stacked DGNN	GCRN-M1 [58]	Spectral GCN [59] & LSTM	Yes	Yes	No	No	Any
	WD-GCN [65]	Spectral GCN [57] & LSTM	Yes	Yes	No	No	Any
	CD-GCN [65]	Spectral GCN [57] & LSTM	Yes	Yes	No	No	Any
	RgCNN [61]	Spatial GCN [62] & LSTM	Yes	Yes	No	No	Any
	DyGGNN [63]	GGNN [64] & LSTM	Yes	Yes	No	No	Any
	DySAT [67]	GAT [68] & temporal attention from [69]	Yes	Yes	Yes	Yes	Any
Integrated DGNN	GCRN-M2 [58]	GCN [59] integrated in an LSTM	Yes	Yes	No	No	Any
	GC-LSTM [72]	GCN [59] integrated in an LSTM	Yes	Yes	No	No	Any
	EvolveGCN [71]	LSTM integrated in a GCN [57]	Yes	Yes	Yes	Yes	Any
	LRGCN [73]	R-GCN [75] integrated in an LSTM	Yes	Yes	No	No	Any
	RE-Net [74]	R-GCN [75] integrated in several RNNs	Yes	Yes	No	No	Knowledge network
Continuous networks							
RNN based							
	Streaming GNN [86]	Node embeddings maintained by architecture consisting of T-LSTM [88]	Yes	No	Yes	No	Directed strictly evolving
	JODIE [87]	Node embeddings maintained by an RNN based architecture	Yes	No	No	No	Bipartite and interaction
TTP based							
	Know-Evolve [89]	TPP parameterised by an RNN	Yes	No	No	No	Interaction, knowledge network
	DyREP [36]	TPP parameterised by an RNN aided by structural attention	Yes	No	Yes	No	Strictly evolving
	LDG [90]	TPP, RNN and self-attention	Yes	No	Yes	No	Strictly evolving
	GHN [92]	TPP parameterised by a continuous time LSTM [93]	Yes	No	No	No	Interaction, knowledge network