DYNGAN: GENERATIVE ADVERSARIAL NETWORKS FOR DYNAMIC NETWORK EMBEDDING

Ayush Maheshwari†, Ayush Goyal#, Manjesh Kumar Hanawal†, Ganesh Ramakrishnan†

Indian Institute of Technology Bombay, India†
Samsung Research, Bengaluru, India#

The Problem

- Embedding large graphs in a low-dimensional space has proven useful in various applications. However, there is a limited focus on real-world networks that are dynamic in nature and continuously evolving with time.
- In this work, we propose a novel adversarial algorithm to learn representation of dynamic networks. We leverage generative adversarial networks and recurrent networks to capture temporal and structural information. We conduct extensive experiments on the task of graph reconstruction, link prediction and graph prediction. Experimental results demonstrate consistent, stable, and better results against state-of-the-art methods in many cases.

GAN Model

Dynamic network. A series of undirected graphs $G_1, G_2 \ldots G_T$ where $G_t = (V_t, E_t)$ represents a graph at time t.

- Our goal is to learn low-dimensional stable representation of vertices v_i over time such that temporal and structural properties of the series of graph are effectively captured. In essence, consecutive embeddings should differ little if graph structure does not change much.
- **Generator** captures the data distribution and learns a parameter θ_g such that $G(v|v_c;\theta_g)$ can approximate the true distribution. v and v_c are the sampled vertices in the generator [3, 5].
- **Discriminator** estimates a probability to differentiate the samples arriving from the generator and true distribution. It learns a parameter θ_d such that $D(v_i, v_j; \theta_d)$ can discriminate between the presence or absence of an edge between v_i and v_j.
- The minimax game with objective function $V(G, D)$ can be formalised as $\min_{\theta_G} \max_{\theta_D} V(G, D)$ where $V(G, D)$ is given as $\sum_{c=1}^{V} \Big(\mathbb{E}_{v\sim p_{true}(\cdot|v_c)}\big[\log D(v, v_c; \theta_D, w_D)\big] + \mathbb{E}_{v\sim G(\cdot|v_c;\theta_G,w_G)}\big[\log\big(1 - D(v, v_c; \theta_D, w_D)\big)\big]\Big)$

Dynamic GAN

- GAN is initialized with random embedding U_{raw}. For each time step t, our model generates embedding U_t that is fed as an input to the next GAN component.
- Such an architecture is capable of handling evolving graphs due to preservation of weights from previous time step. Additionally, it ensures the stability of embeddings due to initialization of current time step embeddings with the previous time step output embeddings.
- However, DynGAN fails to capture temporal sequence in the network due to its limited capability to capture previous time-step information.

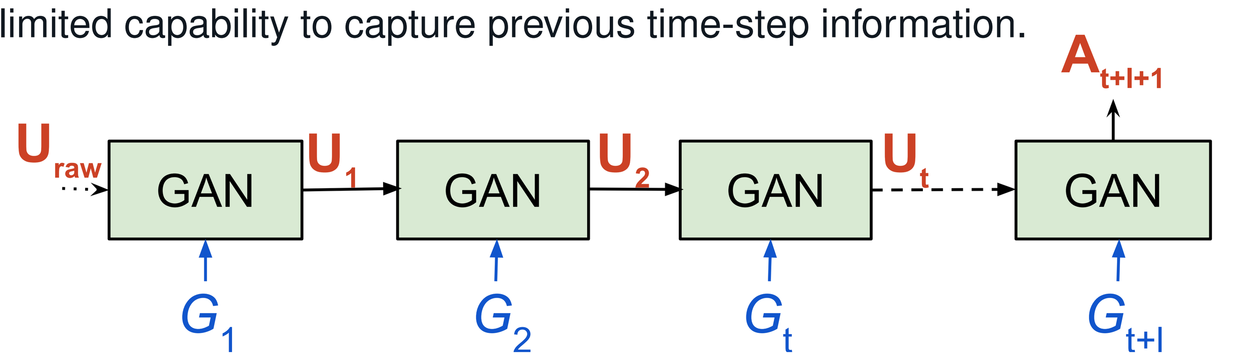

Fig. 1: Architecture of Dynamic GAN

DynGAN-LSTM

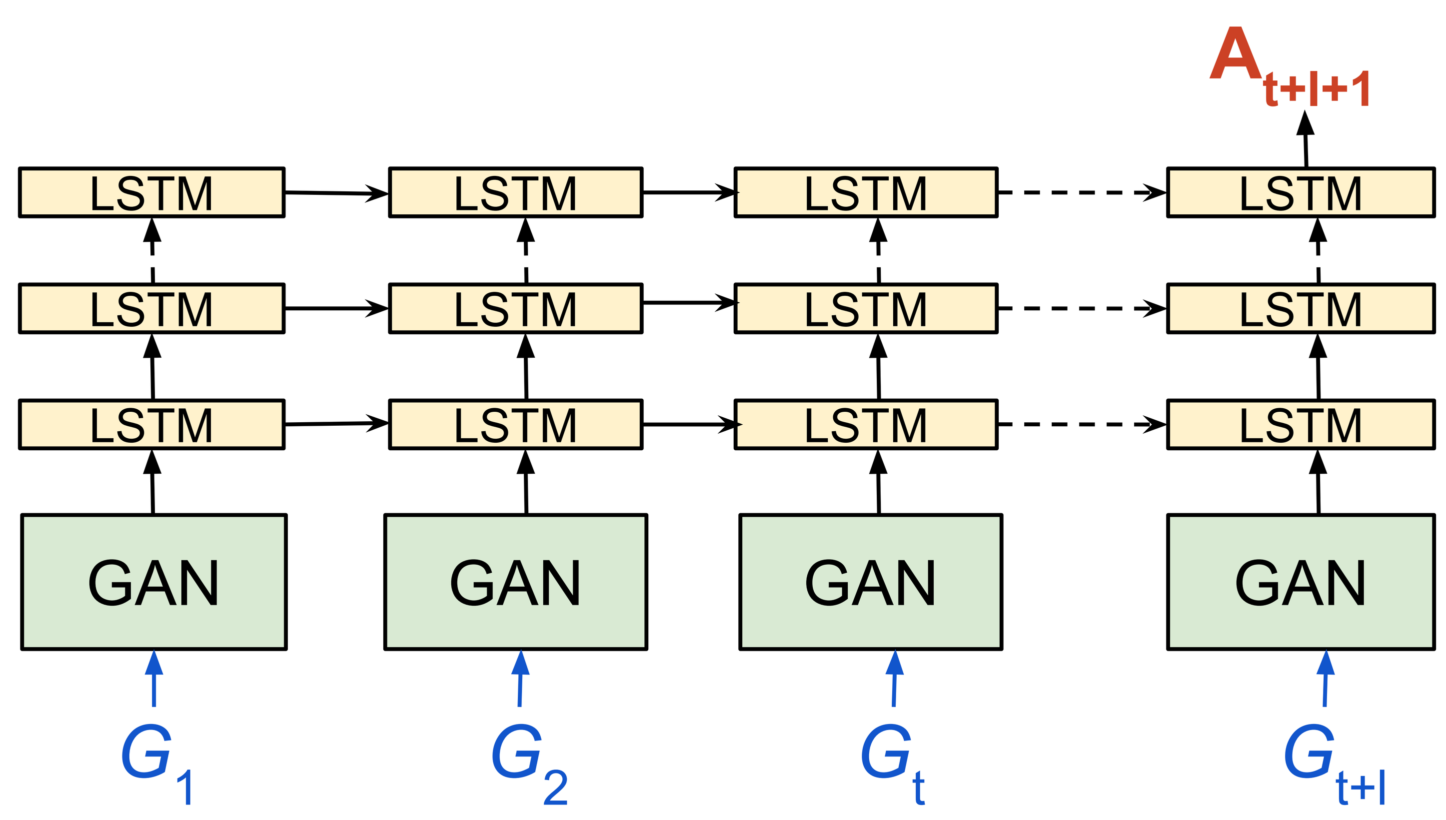

Fig. 2: Architecture of Dynamic GAN - LSTM

We use the DynGAN model to learn embeddings at each time step and pass them through a sequence of LSTM networks to capture sequential information.
Given an adjacency matrix A_t at time t our model optimizes the following loss function,

$$\begin{aligned} L_{t+1} &= ||\hat{A}_{t+l+1} - A_{t+1+1} \odot \beta||_F^2 \\ &= ||(f(y_t, ..., y_{t+l}) - A_{t+l+1})||||(f(A_t, ..., A_{t+1}) - A'_{t+l+1}) \odot \beta||_F^2 \end{aligned}$$

Here, β is a hyperparameter penalizing observed edges that gives higher weights to observed edges than unobserved edges([**laplacian**]), l is the temporal look back factor that controls the range of sequential dependency in our model and $\odot$ represents elementwise product.

Results

We run our experiments on **HEP-TH** that contains 136 time steps and the number of nodes range from 150 to 14446 and **Autonomous Systems(AS)** dataset contains 733 time steps with a fixed number of nodes but number of edges ranges from 487 to 26467.
Graph Reconstruction In this task, we attempt to accurately reconstruct the graph from the learned embeddings of nodes. We reconstruct the edges between pair of nodes using DynGAN model.

Task	Graph Reconstruction		Link Prediction	
Algorithm	AS	HEP-TH	AS	HEP-TH
SDNE [4]	0.214	0.51	0.09	0.1
dynGEM [2]	0.216	0.491	0.21	0.26
DynGAN	**0.465**	**0.65**	**0.464**	**0.636**

Fig. 3: Average MAP for the task of graph reconstruction and link prediction

Graph Prediction - In this task, we train the model with $\{G_1, G_2..., G_{t-1}\}$ snapshots of graphs to predict G_t. Instead of predicting over all time-steps, we consider last 50 snapshots of the datasets. We observe that change in the number of nodes and edges are more frequent in last snapshots of the graph (refer supplementary material). We remark that the efficiency of the models must be tested when a sudden change in the nodes and edge occurs. We train our model with a lookback factor of 2 and embedding dimension of 32.

Comparison

	MAP Estimate		Precision@alledges	
Algorithm	AS	HEP-TH	AS	HEP-TH
dynGEM	0.097	0.258	0.0613	0.073
dyngraph2vecAE [1]	0.182	0.395	0.018	0.003
dyngraph2vecRNN	0.235	0.545	0.438	**0.402**
dyngraph2vecAERNN	**0.275**	**0.595**	0.002	0.005
DynGAN	0.26	0.376	0.152	0.1811
DynGANLSTM	0.232	0.45	**0.637**	0.262

Fig. 4: Average MAP & Precision@all edges for last 50 snapshots of AS & HEP-TH.

In the last 50 snapshots, when nodes and edges are changing by a large number in consecutive snapshots, our model performs marginally lower than dyngraph2vec. However, our model performs better on AS dataset when precision metric is considered and predicts consistently across various scenarios, unlike other models.

Conclusion

- We introduced DynGAN and DynGAN-LSTM, a model for capturing temporal and structural information in the dynamic networks. It learns the evolution pattern in an adversarial manner and predicts node embeddings.
- We conduct extensive experiments on benchmark datasets containing large timesteps and high variations. Our model demonstrates superiority and consistency of results in graph reconstruction, link prediction, and graph prediction and outperforms state-of-the-art methods.

References

[1] Palash Goyal, Sujit Rokka Chhetri, and Arquimedes Canedo. "dyngraph2vec: Capturing network dynamics using dynamic graph representation learning". In: *Knowledge-Based Systems* (2019).

[2] Palash Goyal et al. "Dyngem: Deep embedding method for dynamic graphs". In: *arXiv preprint arXiv:1805.11273* (2018).

[3] Ayush Maheshwari et al. "Representation Learning on Graphs by Integrating Content and Structure Information". In: (2019), pp. 88–94.

[4] Daixin Wang, Peng Cui, and Wenwu Zhu. "Structural deep network embedding". In: (2016), pp. 1225–1234.

[5] Hongwei Wang et al. "GraphGAN: Graph representation learning with generative adversarial nets". In: *arXiv preprint arXiv:1711.08267* (2017).