Learning Social Graph Topologies using Generative Adversarial Neural Networks

Sahar Tavakoli[1], Alireza Hajibagheri[1], and Gita Sukthankar[1]

[1]University of Central Florida, Orlando, Florida
sahar@knights.ucf.edu,alireza@eecs.ucf.edu,gitars@eecs.ucf.edu

Abstract. Although sources of social media data abound, companies are often reluctant to share data, even anonymized or aggregated, for fear of violating user privacy. This paper introduces an approach for learning the probability of link formation from data using generative adversarial neural networks. In our generative adversarial network (GAN) paradigm, one neural network is trained to generate the graph topology, and a second network attempts to discriminate between the synthesized graph and the original data. After the generative network is fully trained, the learned weights can be disseminated and used to "clone" the hidden dataset with minimal risk of privacy breaches. We believe that the learned neural network also has the potential to serve as a more general model of social network evolution.

Keywords: synthetic network generator; link formation; generative adversarial networks

1 Introduction

A key issue in agent-based modeling is how to create a synthetic population with realistic behaviors from a limited amount of survey and census data. To achieve maximum verisimilitude with minimal data collection effort, the modeler must focus on the subset of agent attributes most likely to affect the outcome of the simulation scenario. However, since humans are by nature social animals, accurately modeling social connections has cross-cutting impact for any type of simulation in which the humans are affected by their peers due to social influence, norms, collective behavior, and group dynamics. Hence for many scientific questions, accurately reconstructing the topology of the peer network can dramatically affect the simulation fidelity.

Synthetic network generators can be used to create network topologies for agent-based social simulations in cases where limited network data is available. Most generators assume that the social network fits one of the standard mathematical models of network formation and rewiring, When data is available, fitting techniques can be used to select the model parameters that best match the data. However, human networks often emerge from the confluence of many social processes and are not well modeled by a single model, limiting the applicability of parametric models of network generation. Moreover, these techniques are difficult to extend to networks with node features or multiple layers.

Rather than assuming a specific parametric model, we introduce a semi-supervised model of network generation in which the network topology is learned by generative adversarial neural networks. Generative adversarial networks (GANs) [1] can approximate hidden probability distributions using a competitive learning process in which one neural network generates a sample and a second neural network attempts to discriminate between synthetic samples and examples drawn from the real distribution. In the domains of computer generated artwork and computer vision, GANs have a proven track record of being able to synthetically generate images capable of fooling human observers. We train our GANs by treating the network adjacency matrix as an image; large datasets can be created by sampling the permutation distribution of a single example network. This paper describes our training procedure and compares the GAN output to a synthetic network generator that attempts to directly match network attributes using stochastic optimization (Attribute Synthetic Generator [2]). We demonstrate that our GAN can successfully learn the topology of networks with very different degree distributions, making it more versatile than standard mathematical models.

2 Method

We experimented with multiple neural network architectures before adopting the following procedure. To synthesize the graph topology for the four social networks, the discriminator was provided with flattened adjacency matrices from the original networks, along with 10,000 permutations of the adjacency matrix with randomly swapped node indexes. These matrices can be treated and visualized as greyscale images. Increasing the diversity of input samples makes it more difficult for the GAN to memorize the training set and has been shown to improve the quality of the synthesized data [3].

Our discriminator network includes one linear hidden layer, followed by a rectified linear unit (ReLU) activation layer, and an output layer with a sigmoid activation function (Figure 1, top). The generator input consists of 100 random samples drawn from a normal distribution ($N(0, 1)$). Our generator network consists of two linear hidden layers each with 1000 neurons, batch normalization and ReLU activation functions, and an output layer with n^2 sigmoid units where n is the number of nodes in the original network (Figure 1, bottom). If the original dataset was a sparse network, only one hidden layer with 500 nodes was used during training. In our experiments, the sparsity of the network was determined by dividing the number of edges by the number of nodes and treating it as sparse if the ratio is less than or equal to ten.

2.1 Training Procedure

Despite the simplicity of the GAN model, the training procedure can be tricky because the discriminator and generator must improve in a harmonious fashion such that the discriminator challenges the generator without decisively beating it

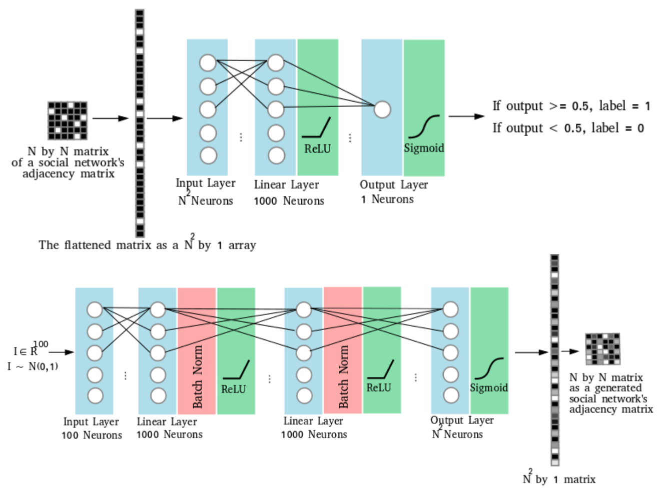

N by N matrix of a social network's adjacency matrix
The flattened matrix as a N^2 by 1 array
Input Layer N^2 Neurons
Linear Layer 1000 Neurons
ReLU
Output Layer 1 Neurons
Sigmoid
If output >= 0.5, label = 1
If output < 0.5, label = 0
$I \in R^{100}$
$I \sim N(0,1)$
Input Layer 100 Neurons
Linear Layer 1000 Neurons
Batch Norm
ReLU
Linear Layer 1000 Neurons
Batch Norm
ReLU
Output Layer N^2 Neurons
Sigmoid
N^2 by 1 matrix
N by N matrix as a generated social network's adjacency matrix

Fig. 1: Network architectures (top: discriminator (D) and bottom: generator (G))

at the min-max game. The training process starts by training the discriminator, as the generator needs the error output from the trained discriminator to progress to the next level. The key point is that D should not be trained completely, which leads to overfitting, and does not improve G in the next level.

Each network is trained using stochastic gradient descent (SGD), and this alternating training process was repeated 1000 times. 10,000 data samples (the original adjacency matrix plus permutations) were presented to the discriminator along with the same number of synthetic samples from the generator in batches of 100. Learning rates for the discriminator and the generator were set to 0.0001 and 0.01 respectively.

The final output of the generator is a matrix with values ranging from zero to one, visualized as a greyscale image. To use this output as a graph adjacency matrix, it needs to be thresholded using a fitting procedure to clone the original dataset. For each dataset, we selected the best of 25 threshold levels using two features to determine the best candidate: 1) edges and 2) degree assortativity.

Our implementation uses the Torch scientific framework [4], and the experiments were performed on a PC with an Intel Xeon(R) CPU W3565 @ 3.20GHz 4 and 23.5 GB memory.

3 Results

To evaluate the performance of our approach, we learned the graph topology for four real-world networks. Table 1 provides the network statistics for each of the real datasets, along with the version of each network cloned using the GAN. As a benchmark, we also provide network cloning results from the Attribute Synthetic Generator [2] which uses stochastic optimization. Although it is difficult to evaluate the generality of the learned link formation model, we can examine the performance of its fitting procedure for cloning networks. Compared to ASG, the GAN performs slightly better at recreating many aspects of the network topology across the four datasets. Note that the time required to generate a network using a GAN is significantly higher than ASG, particularly for the dense Enron network.

Figure 2 shows degree distributions of the real networks (left) as well as the synthetic version generated by the GAN (right). Although many networks possess power law degree distributions, this is not always true for networks generated from different social processes; our four datasets display considerable diversity in the degree distributions. The best model for each network (synthetic and real) was selected based on the value of R^2 which indicates the goodness of a fit (see figure inset for model and R^2 value). As shown in the figures, the GAN successfully recreates the degree distributions of the original networks; however unlike most synthetic network generators, the GANs do not assume that the degree distribution will follow a specific form.

4 Conclusion and Future Work

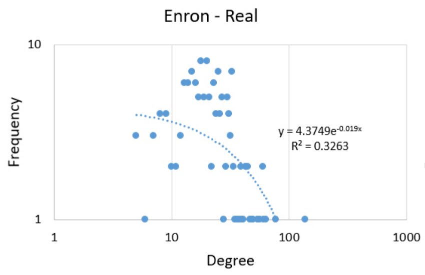
This paper introduces a novel semi-supervised approach for learning graph topologies of social networks using generative adversarial neural networks. There are several promising directions for future work. For instance, alternative network representations such as stochastic block models would be useful for scaling the learning to larger networks. Introducing convolutional layers into the network might facilitate the learning of local structures such as communities. Recurrent neural networks have been used to generate many types of sequential data and could be extended to the problem of learning network dynamics. In the future, we plan to create new architectures for synthetically generating more complex graph topologies including dynamic and multilayer networks.

References

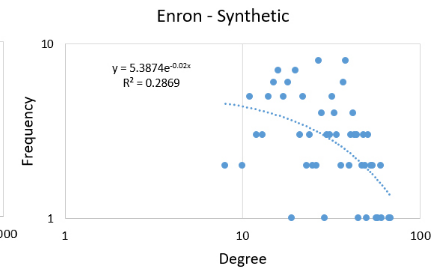
1. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems. (2014) 2672–2680
2. Ali, A.M., Alvari, H., Hajibagheri, A., Lakkaraju, K., Sukthankar, G.: Synthetic generators for cloning social network data. In: Proceedings of the ASE International Conference on Social Informatics. (2014) P41:1–9
3. Im, D.J., Kim, C.D., Jiang, H., Memisevic, R.: Generating images with recurrent adversarial networks. arXiv preprint arXiv:1602.05110 (2016)
4. Torch: `http://torch.ch/`.

Table 1: Statistics of real networks vs. generated networks

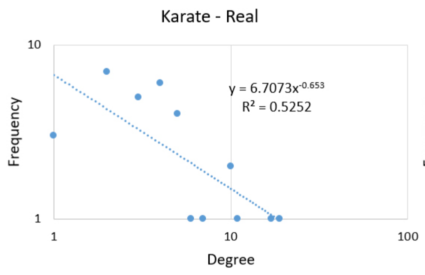
Feature	Method	Enron	Karate	Football	Dolphins
No. of Nodes	Real	154	34	115	62
	GAN	154	34	115	62
	ASG	154	34	115	62
No. of Edges	Real	1917	78	613	159
	GAN	2288	77	472	153
	ASG	497	103	360	194
Average degree	Real	24	4	10	5
	GAN	29	4	8	5
	ASG	6	6	6	6
Max degree	Real	138	19	21	19
	GAN	68	17	19	15
	ASG	50	18	42	28
Min degree	Real	5	1	1	1
	GAN	8	1	1	1
	ASG	1	1	1	2
Assortativity	Real	-0.068	-0.547	-0.144	-0.174
	GAN	-0.093	-0.250	-0.146	-0.127
	ASG	-0.140	-0.035	-0.080	-0.127
Network Diameter	Real	3	4	5	5
	GAN	3	5	5	5
	ASG	8	5	8	6
Avg. Path Length	Real	1.923	2.157	2.335	2.620
	GAN	1.837	2.456	2.460	2.558
	ASG	3.586	2.770	3.619	3.190
Runtime (hrs)	Real	-	-	-	-
	GAN	62.26	2.92	33.47	9.27
	ASG	< 1	< 1	< 1	< 1



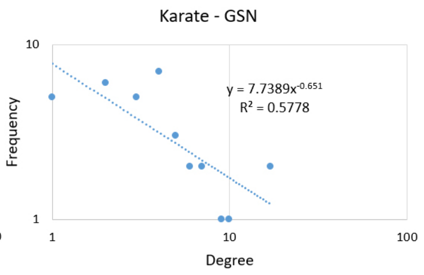
(a)



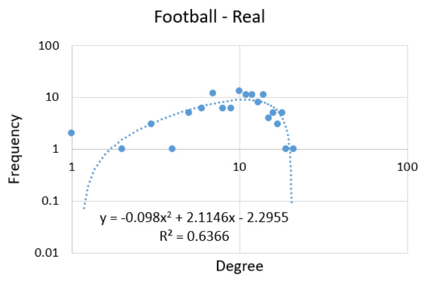
(b)



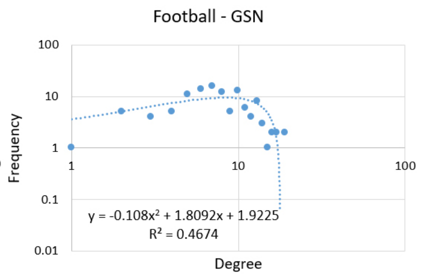
(c)



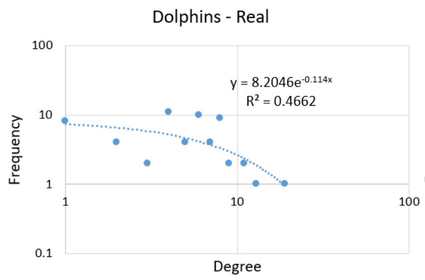
(d)



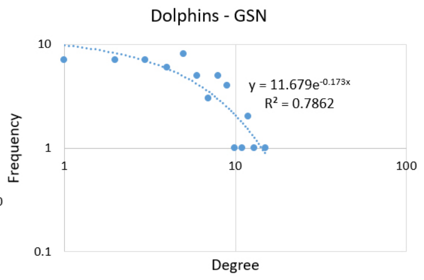
(e)



(f)



(g)



(h)

Fig. 2: Degree distributions of different real networks vs. networks generated by GAN. Note that the networks do not always follow a power law distribution. The best fitting distributions were selected using the R^2 fitness value which is displayed at the bottom of the figures, along with the model. The GAN does well at matching the distributions of the original networks without any *a priori* knowledge.