Software Engineering Department
Braude College

Capstone Project Phase B – 61999

Temporal Link Prediction Model for Weighted Dynamic Networks

23-1-R-15

Students:

Tal-Chen Ben-Eliyahu 318173044 Talchenben1234@gmail.com
Lior Karish 318267242 Liorkarish@gmail.com

Supervisors:

Dr. Renata Avros

Prof. Zeev Volkovich

Table of content

Abstract - This paper presents an algorithm for dynamic link prediction. There are several issues with the reliability of articles, in particular, the reliability of reference links inside a network. A link prediction approach is intended to evaluate the readiness of the citations. The previously presented methods do not take into account possible edges weights. Therefore, to improve the prediction, the model contains GCN-GAN which is capable to deal with this problem and predict links between nodes in non-linear networks. The current project utilizes the GCN-GAN model to predict links. The GCN-GAN model utilizes the graph convolutional network (GCN) which analyzes the localized topological properties of individual snapshots, long short-term memory (LSTM) to capture the evolving features of the dynamic networks and utilizes the generative adversarial network (GAN) which creates the next snapshot of the graph. By fully leveraging the GCN-GAN model, it is possible to conserve the topological structure and dynamic prediction of the weight links. The algorithm proposed in this article predicts the links between the articles using a random weight. When the model can predict the correctness of links given articles and references in the article. To verify the model's effectiveness, we conduct extensive experiments on iCite dataset, for several iterations. Therefore, adjustments were required to serve the model. In addition, since we are utilizing GCN-GAN the input of the model is snapshots of the graph. Therefore, we created a snapshot with random weights. Finally, we found the number of arcs that are incorrectly predicted for a constant percentage error per number of iterations. The result obtained for the output error is about 3%. The experimental results demonstrate that our model achieves impressive results.

Keywork: Temporal Link Prediction, Weighted Dynamic Networks, Generative Adversarial Networks, Graph Convolutional Networks.

1. Introduction

A dynamic network is a network changing over time. Prediction in a dynamic network can be a difficult task, mainly because of the complexity of the networks themselves. Moreover, the behavior of the network can be complicated because it is affected by a wide variety of factors, such as external factors, and this makes it difficult for the network to predict the next time frame.

There are several studies that have addressed the problem of prediction in dynamic networks. For example, in [1], dynamic networks were used to predict bandwidth usage due to the demand for more bandwidth increases, which can both save energy and improve browsing performance. Today, there are studies that focus on solving this problem, but most of them do not work for general scenarios. However, in [2] they propose a new approach to learning the non-linear temporal link prediction of weighted dynamic networks.

The proposed model is a new model for predicting a relationship between articles based on the model described in [2] GCN-GAN. The contribution is to predict the link between articles, in particular, to check whether an article is really based on its link. When predicting links between two articles and check whether there is a link between them. For example, to predict if they have common topics or whether they share the same citations or sources. In the project, network analysis techniques are used to predict the link between articles. The proposed network is represented as a graph and each node represents an article. The edge between the two nodes means that there is a connection between the two articles, i.e., the information in the two articles is connected. By analyzing the network, it is possible to identify a pattern that indicates a possible connection between the two articles.

The remainder of this article is organized as follows. In section 2. Introducing the related work. In Section 3. Defining the problem of temporal link prediction of articles in a weighted dynamic network. The mathematical background is introduced in section 4. In Section 5. Presenting a diagram of the flow and the algorithm of the paper workflow. In section 6. Presenting the expected achievements. In section 7. Presenting the final product including diagrams GUI and tests. The evaluation plan is presented in section 8. In section 9. A summary of the paper is presenting.

2. Related Work

In recent years, the use of dynamic networks has significantly increased. There are several uses that have been developed over the years to deal with predictions in dynamic networks. In [1], the authors introduce Convolutional Neural Networks (CNNs) to predict short-term traffic in a data center network. Specifically, they focus on predicting the number of users that will use a network connection on a second-by-second basis. Moreover, the use of link prediction plays an important role in data mining. In [3], the authors propose a new method for predicting links in social networks. They use the topological information and attributes of the nodes to build a vector, and then predict the links between them. In GCN-GAN [2], they use GCN [4] and LSTM [5]. In [4], the authors present a model that scales linearly with the number of graphs edges and learns hidden representations that encode both the local graph structure and the features of the nodes. The nodes in [4] are documents where labels are only available for a small subset of the nodes. In [5], the author introduces Long Short Term Memory. The advantage presented in [5] is the use of forget gates, which allow an LSTM cell to learn to reset itself at appropriate times, thus releasing internal resources. Therefore, in this paper, GCN-GAN is utilized to predict the relationship between two articles. This paper also relies on the aforementioned article to build a new and improved link prediction model in a dynamic network.

3. Problem Definition

Today, there are several issues with the reliability of articles, in particular, the reliability of reference links inside the articles. For example, they may create fake articles or websites and try to link them to legitimate sites in an attempt to make them appear more credible. It is important for individuals and organizations to be aware of these potential risks and to take steps to protect themselves from fraud. When a person publishes an article, it is advisable to check its correctness by checking the articles that are in the references. If they are related to the topic of the published article, then the article is legit. Define a graph $G = \{G_1, G_2, \dots, G_\tau\}$ as dynamic network while $G_t = \{V, E_t, W_t\}$. G_t is a graph in time slice t when $t \in \{1,2, \dots, \tau)$ and τ represent the current time, V is the set of nodes, E_t is the set of edges and W_t is the weight between two nodes in G. When proposed network is represented as a graph so that each node represents an article. The edge between the two nodes means that there is a connection between the two articles, i.e., the information in the two articles is connected. By analyzing the network, it is possible to identify a pattern that indicates a possible connection between the two articles. In this study, the purpose is to predict a link between articles, in particular, to check whether an article is genuine based on its link. Described by a formula:

$$\tilde{A}_{\tau+1} = f(A_{\tau-l}, A_{\tau-l+1}, \dots, A_\tau) \tag{1}$$

Where f is the proposed model and A is an article from the data set and $\tilde{A}_{\tau+1}$ represents the prediction result of the proposed model.

4. Mathematical Background

4.1 Graph Convolutional Network (GCN)

GCN is a graph convolutional network. GCN has a "graph convolutional layer" that filters the graph data. In [4] they represent a model that can learn the structure of the graph and the properties of each node.

$$GCN(Z, A_t) = X_t \tag{2}$$

when:

- A_t - adjacency matrix when $A_t \in R^{N \times N}$
- Z- noise matrix when $Z \in R^{N \times N}$
- X_t- output of the GCN layer, when $X_t \in R^{N \times N}$

The input of GCN layer is a set of adjacency matrices, $A^{\tau}_{\tau-l}$, and a noise matrix Z. Where each adjacency matrix A_t ($t \in \{\tau - l, \cdots, \tau\}$) should be normalized into the rang $[0, 1]$ as the cells in the adjacency matrix represent the weight of the edge, and its value may vary greatly. The outputs of the GCN are a sequence $X^{\tau}_{\tau-l}$ which is enter into the LSTM layer. Furthermore, the activation function is the sigmoid function of all GCN units, and the noise matrix Z represents a uniform distribution in rang $[0, 1]$. Note: ($t \in \{1,2, \cdots, \tau\}$ when τ represents the current time.

$$X = GCN\,(Z, A) = f\left(\widehat{D}^{-\frac{1}{2}}\,\hat{A}\,\widehat{D}^{-\frac{1}{2}}\,Z\,W\right), \tag{3}$$

when:

- f- The sigmoid activation function.
- $\hat{A} = A + I_N$
- $\widehat{D}^{-\frac{1}{2}}\,\hat{A}\,\widehat{D}^{-\frac{1}{2}}$ - Approximate graph convolution.
- $\widehat{D}_{ii} = \sum_{j=1}^{N} \widehat{A}_{ij}$
- W- weight matrix.
- X- Output of the GCN layer, when $X_t \in R^{N \times N}$

<u>*Adjacency Matrix*</u>

The adjacency matrix represents a graph. The size of the adjacency matrix is determined by the number of nodes. Adjacency matrix with N nodes will be in size $N \times N$. If there is a connection between node i and node j then the positions (i, j) and (j, i) will be 1.

The definition of adjacency:

$$A_{i,j} = \begin{cases} 1, & if\ there\ is\ an\ edge\ from\ V_i\ to\ V_j \\ 0, & otherwise \end{cases}$$

<u>*Noise Matrix*</u>

The noise matrix represents the probability distribution - uniform distribution. The size of the noise matrix is the same as the adjacency matrix, size of $N \times N$.

4.2 Long Short-Term Memory (LSTM) Network

LSTM is Long Short-Term Memory [5]. LSTM is an improvement of the recurrent neural network (RNN), which is capable of learning and storing information over long periods of time. The stored information is controlled by gates (i.e., the input gate, the forget gate and the output gate). These gates allow the network to remember things over a long period of time without losing them due to the "vanishing gradient problem", i.e., when the elements of the network are too small. Usually, the use of LSTM to predict the next snapshot of time series tasks when the value of the network changes over time. When the power of LSTM is used to learn long-term dependencies in sequential data and capture evolving patterns in dynamic networks. For example, LSTM are particularly useful for tasks such as natural language processing, speech recognition, and video analysis, where the input data has a temporal dimension, and the order of the data points is important.

The LSTM layer receives $X_{\tau-l}^{\tau}$ as input which is the output of the GCN layer. Where each matrix X_t (t belonging to the set $\{\tau - l, \cdots, \tau\}$) is converted into a row-wise vector X_t. The outputs of LSTM CELL are a vector $h_{\tau-l}^{\tau} = \{\mathrm{h}_{\tau-l}, \cdots, \mathrm{h}_{\tau}\}$. Eventually, the last hidden state h_τ enter the dense- output layer to generate the graph snapshot $\tilde{A}_{\tau+1}$, in the range $[0,1]$, of the next time slice.

4.3 Generative Adversarial Network (GAN)

GAN consists of two main parts: a discriminator network and a generator network. The generator creates fake samples, and the discriminator classifies whether they are fake or real. The diagnostician is a classifier, and its purpose is to distinguish between real examples, the training set, and fake examples generated by the neural network of the generator. The generator learns the distribution of the data, and its purpose is to learn to produce new data that looks like it was taken from the same distribution. The purpose of the discriminator is to increase as much as possible the difference between the value given to the real examples and that given to the fake examples, (that is, to increase as much as possible the distance between the distribution of the real information and the distribution of the examples created by the generator). While the purpose of the generator is to reduce the difference between the real examples and the ones it generates, the generator is to try to 'fool' on the discriminator.

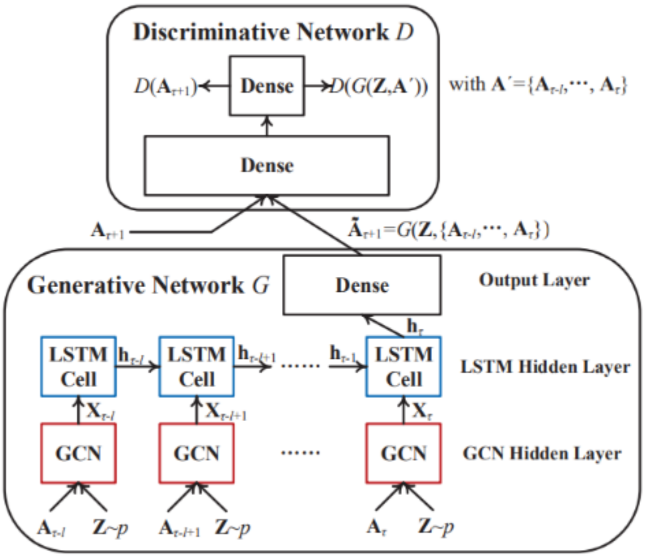


Fig. 1. The GCN-GAN temporal link prediction architecture model. This diagram is taken from: "K. Lei, M. Qin, B. Bai, G. Zhang and M. Yang, "GCN-GAN: A Non-linear Temporal Link Prediction Model for Weighted Dynamic Networks", IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, 2019, pp. 388-396, doi: 10.1109/INFOCOM.2019.8737631." from page 3.

4.3.1 Generative Network G

The generator model G consists different networks: GCN, LSTM (as discussed in the section above) and Dense. When GCN is catch the characteristics of the topological structure hidden in each single graph snapshot. Then, the output of the GCN is the input of LSTM network, that catch patterns of the weighted dynamic network with multiple continuous time slices. The output of the LSTM network is a row-wise vector that is fed into the dense network, which performs the reverse process of normalization. For more information about the dense layer, see section 4.3.3.

Activation Function

In the GCN-GAN network, the activation function is the sigmoid function. The sigmoid activation function takes any real value as input and outputs values in the range of [0,1]. The larger the input, the closer the output value is to 1.0, while the smaller the input, the closer the output value is to 0.0. The advantage of the sigmoid function is that the output values are always in the range [0,1]. Most models that use the sigmoid function predict the probability as the output (when the probability is only between [0,1]).

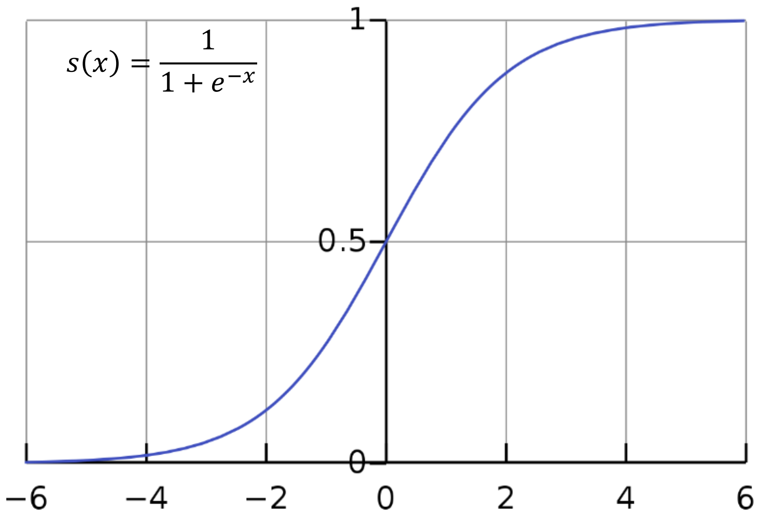


Fig. 2. The sigmoid function.

This figure is taken from: Wikipedia, https://en.wikipedia.org/wiki/Sigmoid_function

4.3.2 Discriminative Network D

The discriminative model D over a fully connected neural network with a hidden layer and an output layer. In the training process, D takes the output of the generator G, $\tilde{A}_{\tau+1}$, or the training data, $A_{\tau+1}$, as input. Since the input data in the hidden layer is usually represented as a vector, the dense layer in the discriminator D converts the matrix input into a corresponding row-wise long vector. Each matrix input $A_{\tau+1}$ should be normalized to the range $[0,1]$. Note that $\tilde{A}_{\tau+1}$ is already normalized in the generator network G. Finally, the output layer is set to be linear layer when the activation function is also linear since the model is trained in GCN-GAN with Wasserstein-GAN (WGAN).

4.3.3 Dense Layer

The dense layer is a simple layer of neurons. Here, each neuron in the layer receives input from the previous layer. The dense layer first applies a linear operation to the input and then uses a non-linear activation function to produce the output. Usually, there is more than one dense layer, and the output of the dense layer is fed into the next dense layer.

4.3.4 Root Mean Square Propagation (RMSProp) Algorithm

RMSProp is Root Mean Square Propagation [7] is an optimization algorithm used to train neural networks. It is an extension of the gradient descent algorithm that helps reduce the oscillations in the gradients and makes the training process more efficient. The key idea behind RMSProp is to divide the learning rate by an exponentially decaying average of the magnitudes of the recent gradients. In [2] they use RMSProp to optimize the model. They use RMSprop algorithm to improve the parameters of network G and D until coverage.

4.4 Wasserstein GAN (WGAN)

In [6], they use the Wasserstein GAN, in order to improve the general GAN.

4.4.1 Wasserstein distance

The Wasserstein distance for the real data distribution P_r and the generated data distribution P_g is mathematically defined as the greatest lower bound for each transportation plan (i.e., the cost of the cheapest plan):

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(\mathrm{P_r}, \mathrm{P_g})} \mathrm{E}_{(x,y)\sim\gamma}[||x - y||] \qquad (4)$$

When:

- $W(P_r, P_g)$ are the Wasserstein distance between the probability distribution of the real data (P_r) and the generated data (P_g).
- $(x, y)\sim\gamma$: x, y (sampled from the two distributions) are distributed based on the joint distribution that picked to satisfy the equation.
- $\mathrm{E}_{(x,y)\sim\gamma}[||x - y||]$ is the average of $[||x - y||]$.
- $\gamma \in \Pi(\mathrm{P_r}, \mathrm{P_g})$ is the join distribution.
- $\inf_{\gamma \in \Pi(\mathrm{P_r}, \mathrm{P_g})}$ - the minimum between the join distribution ($\gamma \in \Pi(\mathrm{P_r}, \mathrm{P_g})$).

But with this expression there is a problem that this distance is not resolvable. Therefore, a new formula has been developed which is simpler and uses 1-Lipshitz, and this equivalent expression is easier to approximate.

$$W(P_r, P_g) = \sup_{\|f\|_L \leq 1} \mathrm{E}_{x \sim P_r}[f(x)] - \mathrm{E}_{x \sim P_0}[f(x)] \tag{5}$$

When, the difference from the first expression is:

- $\|f\|_L \leq 1$: is 1-Lipschitz function.
- $\mathrm{E}_{x \sim P_r}[f(x)] - \mathrm{E}_{x \sim P_0}[f(x)]$ is the average of any function between real and generated data.
- $\sup_{\|f\|_L \leq 1}$ - the maximum between the average of the real and generated data.

The Wasserstein distance is the highest average difference of any function between the real input and the generated input when the function f is a 1-Lipschitz function. The 1-Lipschitz function is a function that for any two inputs and a distance function, the inputs are no less far apart than the outputs. Conveniently, it limits the function slope to one at most all along its trajectory. Now let's look at how the loss itself is defined for both the generator and discriminator:

Discriminator:

$$\nabla_w \frac{1}{m} \sum_{i=1}^{m} [f(x^i) - f\left(G(z^i)\right)] \tag{6}$$

Generator:

$$\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} f\left(G(z^i)\right) \tag{7}$$

When:

- $f\left(G(z^i)\right)$ is the data generated from the generator.
- $f(x^i)$ is the real data – the training set.

As with GAN for the discriminator, aim to take the difference between the scores of the real data and the fake data on average and update the weights to make that difference larger. There are two ways that discriminators can achieve this purposed, either by increasing the score of the real data or decreasing the score of the generated data. The purpose of the generator is the opposite. It produces fake data that leads to the highest possible score.

5. Overview of The Paper

5.1 Workflow

The aim of this research is to predict the connection between articles, where each article is represented by a node in a graph. The workflow in Fig. 3 shows the whole cycle of the learning process of the model.

First, the data set of articles passes through pre-processing, because if the parameters of G will not pass adjustments, then G can fool D but it may not be consistent with the next graph snapshot. Ideally, the prediction result should be as close as possible to the actual outcome. To address this issue, each adjacency matrix in the data set passes through the pre-processing using the following loss function:

$$\min_{\theta_G} h\left(\theta_G; Z, A_{\tau-l-1}^{\tau-1}, A_\tau\right) = \left\|A_\tau - G\left(Z, A_{\tau-l-1}^{\tau-1}\right)\right\|_F^2 + \frac{\lambda}{2}\|\theta_G\|_2^2 \tag{8}$$

In (8), G's parameters θ_G are used to try to restore the current graph snapshot A_τ, using the snapshot sequence $A_{\tau-l-1}^{\tau-1}$ and noise Z. This process allows G to fully incorporate the most recent temporal information of the dynamic network, which are the characteristics most like the true snapshot of $A_{\tau+1}$. The λ term is a parameter used to control the impact of the L2-regularization term.

Now, G has the initial capability to predict the next snapshot of the graph. After the pre-processing each adjacency matrix in the data set is normalized in order to be used by GAN. The model utilizes WGAN (as mentation in section 4.4) because it has more reliable performance than the standard GAN. Then, Generative network G takes the adjacency matrix and the noise matrix to the GCN model and LSTM. Finally, after the dense layer, G outputs the training set. The following loss function is used to update the parameters of D (θ_D) while keeping the parameters of G fixed:

$$\min_{\theta_D} h_D\left(\theta_D; Z, A_{\tau-l-1}^{\tau-1}, A_\tau\right) = E[D(A_\tau)] - E[D\left(G\left(Z, A_{\tau-l-1}^{\tau-1}\right)\right)] \tag{9}$$

The discriminative model D tries to distinguish between the real data set and the training set. To do this, it utilizes two layers of dense connections and updates the loss function accordingly. Then after updating the parameters of D, it clipped into a pre-defined range of $[-c, c]$. Then the following loss function is used to update the parameters of G (θ_G) while keeping the parameters of D fixed:

$$\min_{\theta_G} h_G\left(\theta_G; Z, A_{\tau-l-1}^{\tau-1}\right) = -E[D\left(G\left(Z, A_{\tau-l-1}^{\tau-1}\right)\right)] \qquad (10)$$

After this, RMSProp algorithm is adopted in order to update the parameters of θ_D and θ_G until coverage. Upon completing the training process, G can accurately predict the connections between the nodes.

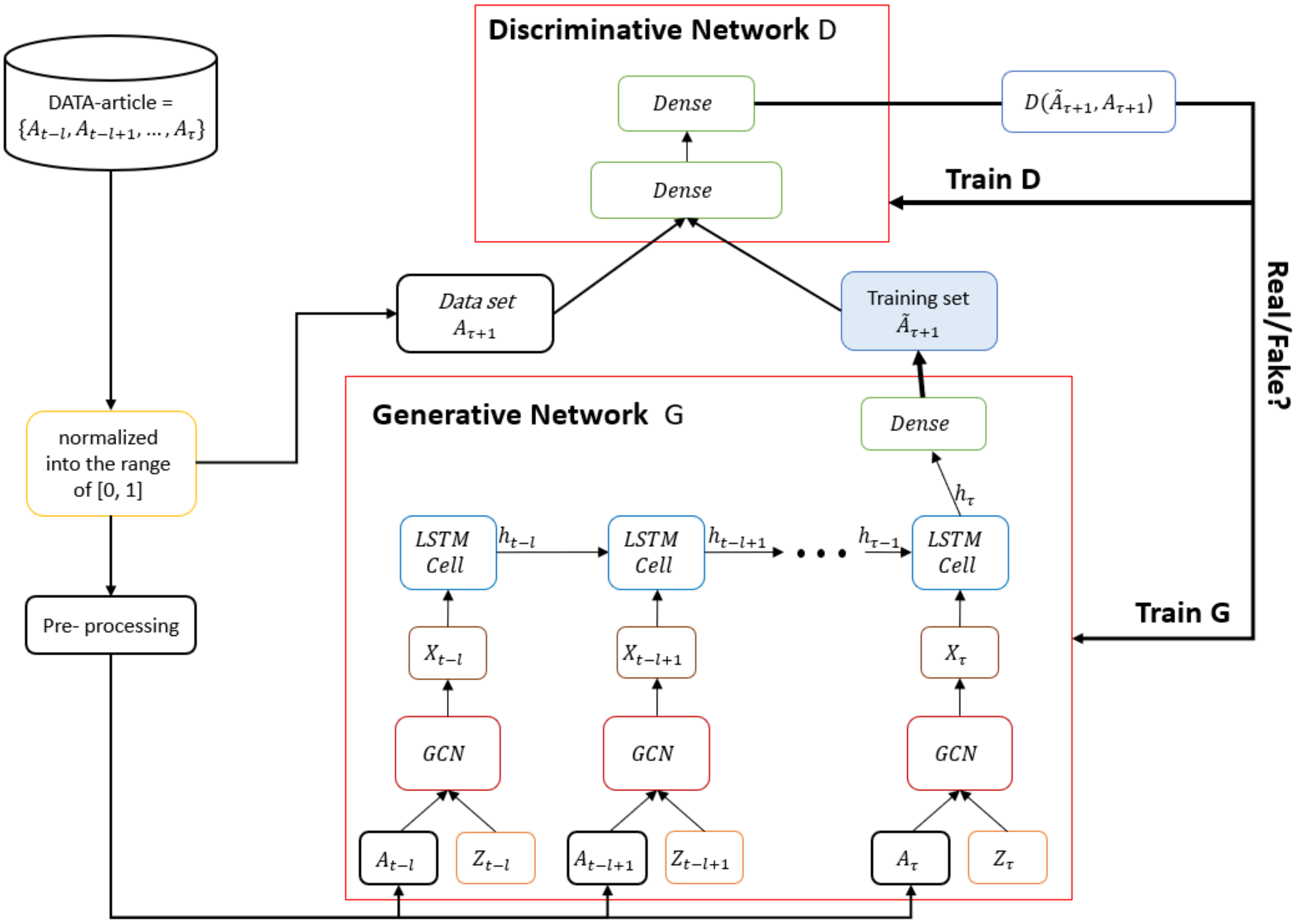


Fig 3. Workflow. The GCN-GAN temporal link prediction architecture model.

5.2 Pseudocode

Input: Dataset of articles - $\{A_{\tau-l}, A_{\tau-l+1}, \dots, A_\tau\}$

Output: Result prediction -$\tilde{A}_{\tau+1}$.

1. Normalize the values of $\{A_{\tau-l}, A_{\tau-l+1}, \dots, A_\tau\}$ into $[0, 1]$.
2. Pre-Processing:
   // The purpose of the pre-processing is to give G the ability to make the initial prediction
   2.1. for i from 1 to $\{number\ of\ iterations\}$

2.1.1. Generate the noise input $Z \sim U\,[0,1]$.
2.1.2. Calculate the loss function of G.
2.1.3. Update the parameters of G using RMSProp algorithm[7].

3. Train model:
   3.1. for i from 1 to $\{number\ of\ iterations\}$
      3.1.1. Generate the noise input $Z \sim U\,[0,1]$.
      3.1.2. Calculate the loss function of D.
      3.1.3. Update the parameters of G using RMSProp algorithm[7].
      3.1.4. Clipped D parameters into $[-c, c]$.
      3.1.5. Generate the noise input $Z \sim U\,[0,1]$.
      3.1.6. Calculate the loss function of G.
      3.1.7. Update the parameters of G using RMSProp algorithm[7].
4. Test model:
   4.1. Generate the prediction result $\tilde{A}_{\tau+1}$.
   4.2. Renormalize $\tilde{A}_{\tau+1}$ into the real value range.
   4.3. Refine the prediction result:
      4.3.1. Turn $\tilde{A}_{\tau+1}$ symmetric.
      4.3.2. Remove the edges with weight equal to zero.
      4.3.3. If the weight of the edge is less then ε, set the weight of the edge to be zero.

6. Expected Achievements

This paper presents the GCN-GAN model for non-linear temporal link prediction for weighted dynamic network. The model will be utilized to predict the link between articles. The aim of our project is to determine whether the links in each article are meaningful and relevant based on the database of articles. Specifically, the model will be able to predict the readiness of the links within a given article.

7. Research / Engineering process

7.1 The Process

Part A of the project includes few steps of research. The first step is to expand the understanding and learning about various data mining techniques. Then, studying the GCN-GAN [2] article and the algorithm it presents, thoroughly understanding and summarizing the article with comments, examples, and detailed proofs as an appendix. Additionally, examining the Wasserstein GAN model as a potential alternative to the standard GAN.

Prat B of the project includes implementation of CGN-GAN algorithm for predicting link connection between articles as discussed throughout this paper. The paper [2] also mentions the use of a certain type of GAN algorithm that can improve the efficiency of the algorithm. The graph is converted into an adjacency matrix and this matrix will be used as input for the GCN-GAN algorithm.

7.2 Product

The aim of this research is to accurately predict the validity of links present within a given article, represented by nodes in a graph. To achieve this, is required to embed each article in order to represent the dataset of articles as a graph and then convert it into an adjacency matrix. The dataset, in the form of an adjacency matrix, will undergo pre-processing and normalization to be used by the GAN. In the pre-processing stage, the G network aims to reconstruct the current graph snapshot and a noise matrix, providing G with the ability to make an initial prediction. The graph is then represented as a normalized adjacency matrix, which is passed on to the GCN model and LSTM by the generative network G. The output of this process is the training set, which is then fed to the discriminative model D, which tries to differentiate between the real dataset and the training set using two layers of dense connections, updating the loss function accordingly. Following the training process, G will be able to accurately predict the connections between nodes. For more information, see section 5 of the paper.

7.2.1 Diagram

Use Case

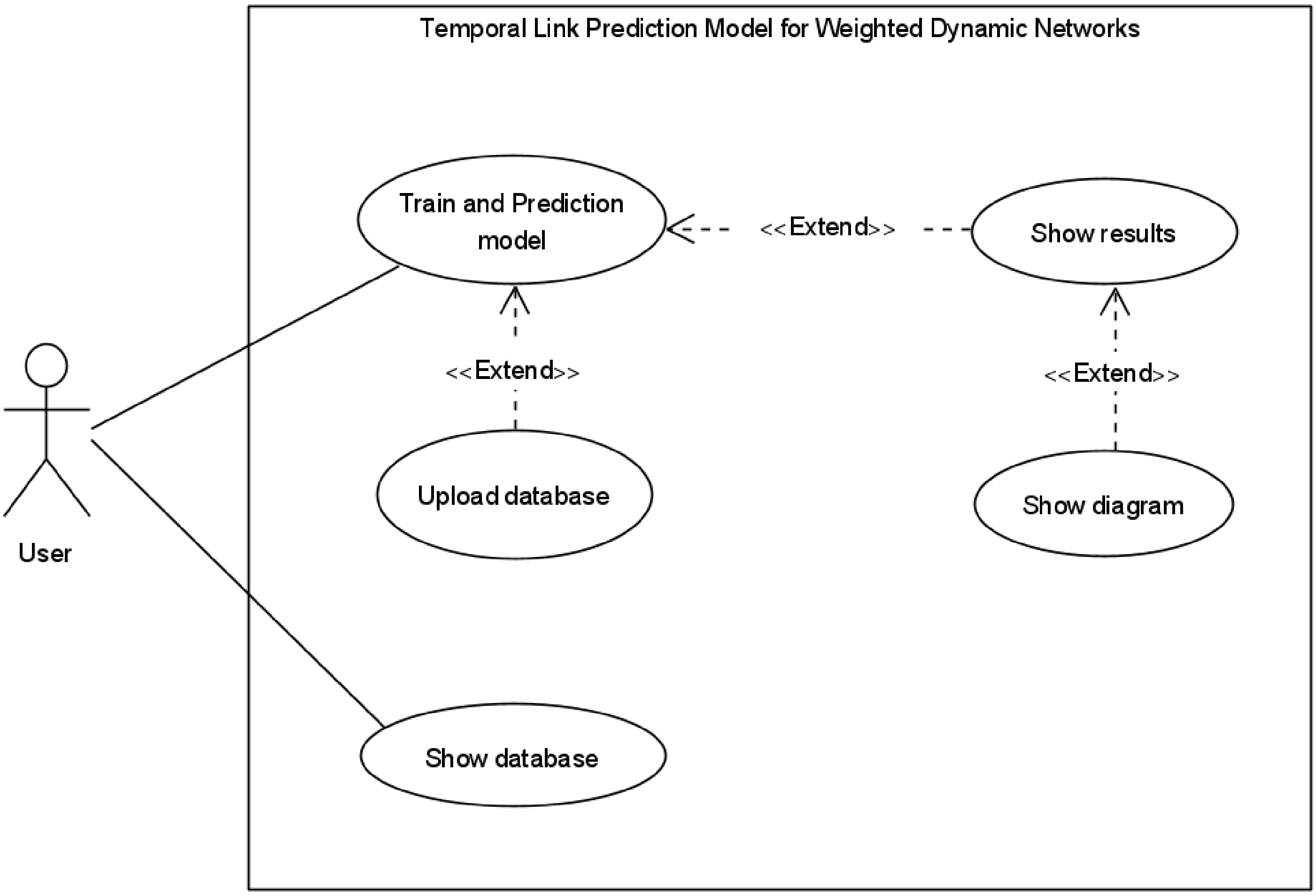


Fig. 4. Use-case diagram

Class Diagram

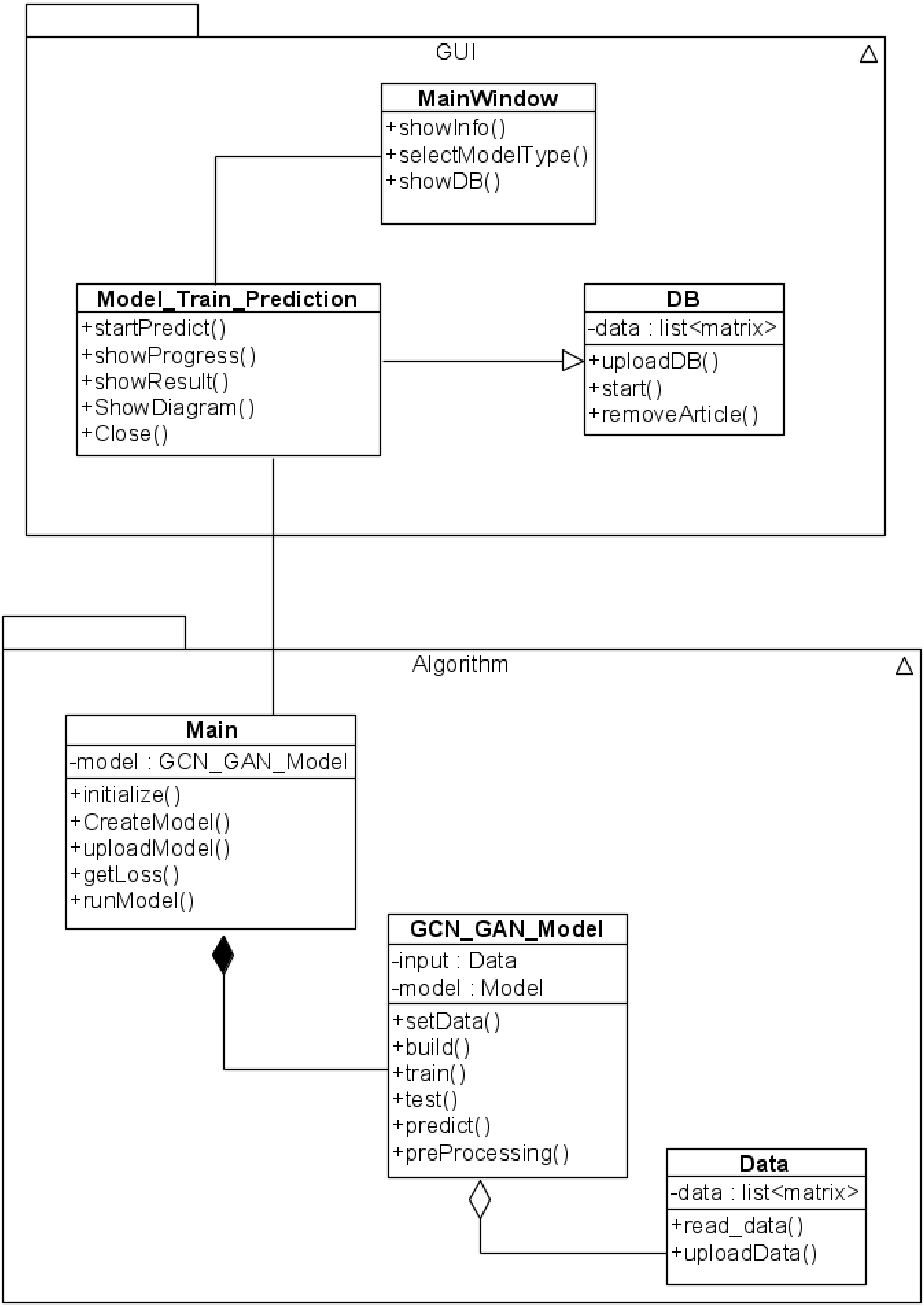


Fig. 5. Class diagram.

7.2.2 GUI

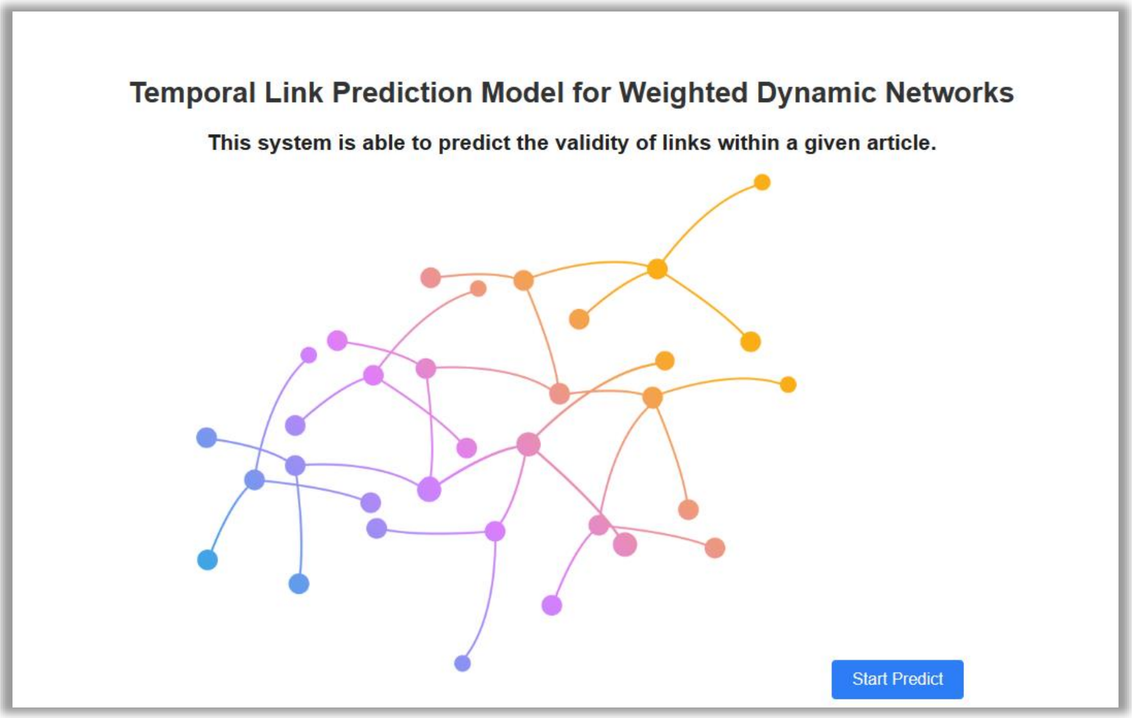


Fig. 6. GUI- home page.

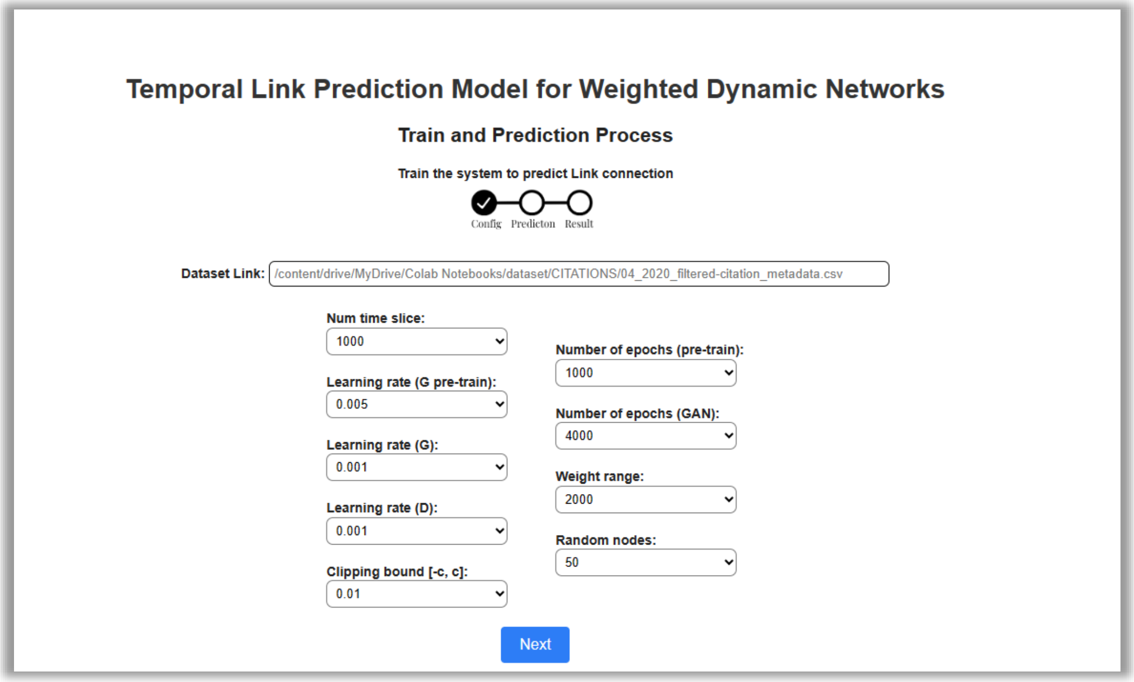


Fig. 7. GUI- Train and prediction process, the user can change the parameters of the model.

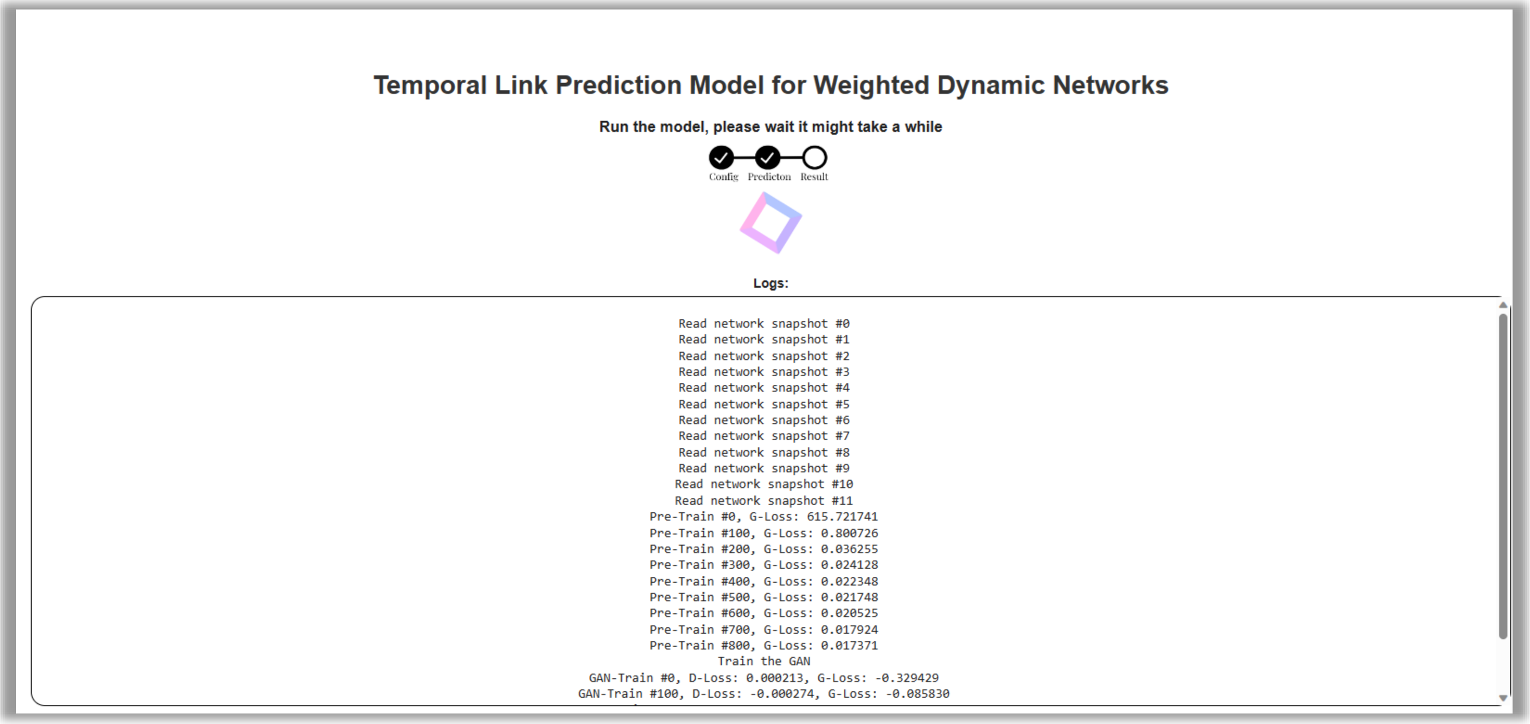


Fig. 8. GUI- Train and prediction process, the user can view the logs of the model.

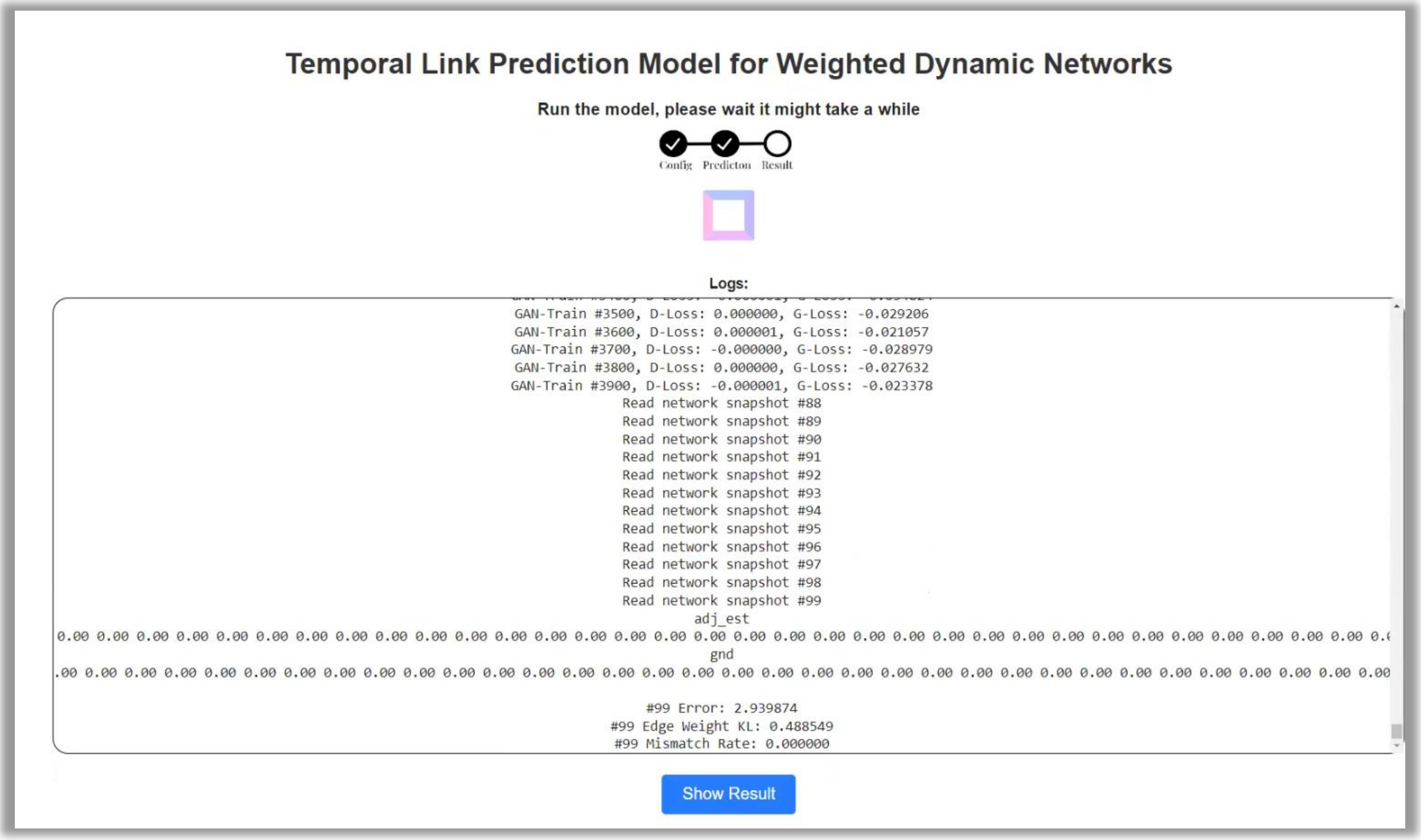


Fig. 9. GUI- Train and prediction process- continue: the user can view the full logs of the model. Additionally, the user has the option to view the results by clicking on the 'Show Result' button.

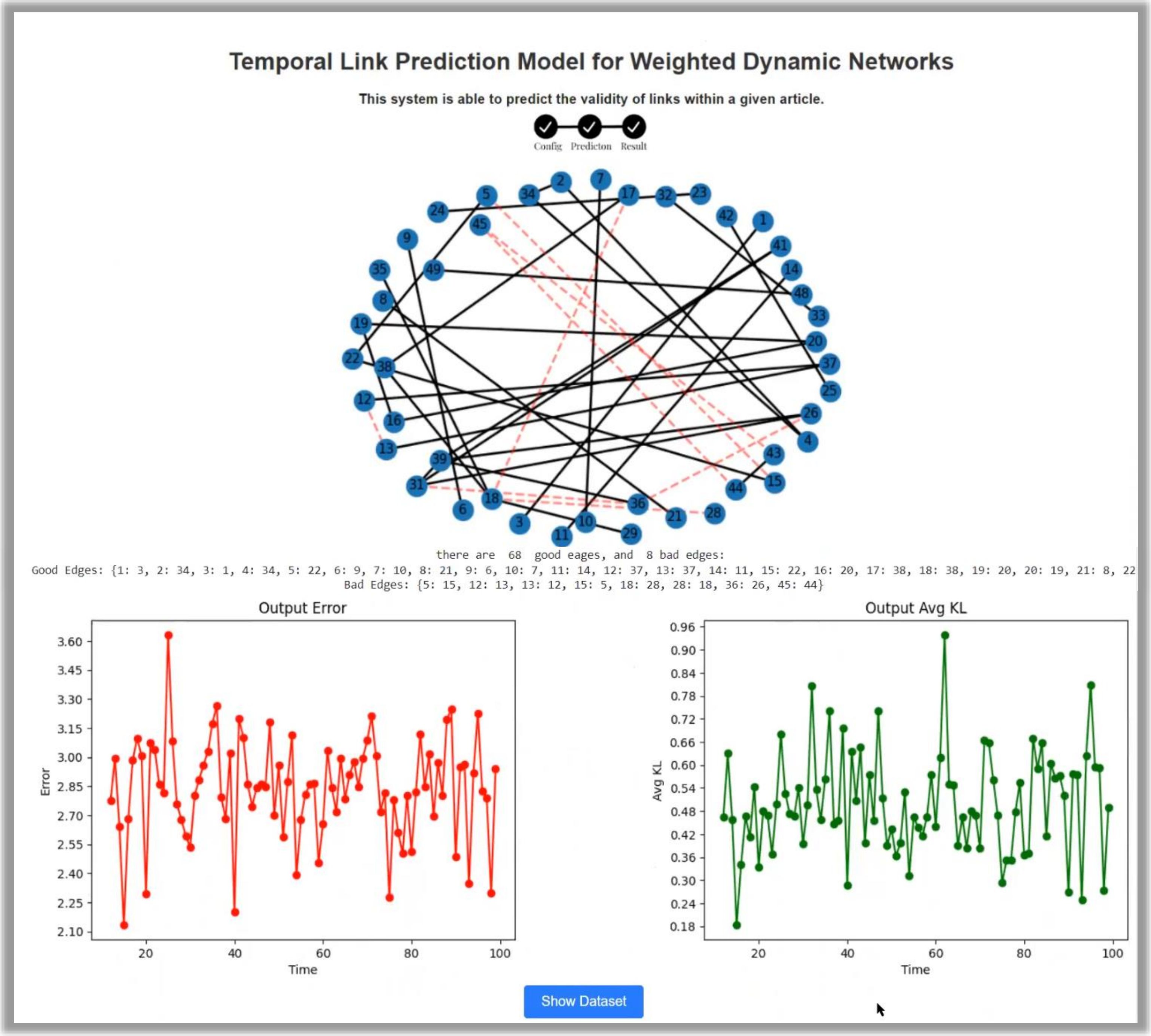
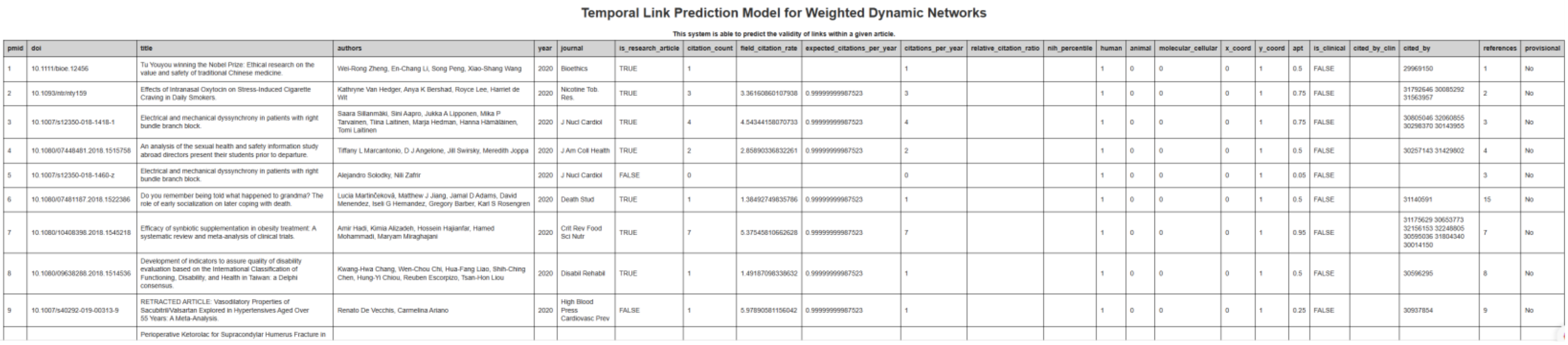


Fig. 10. GUI- Train and prediction process, the conclusion of the train and prediction model. When the result that sown is the predicted graph, average error and KL-Divergence. Additionally, they have the option to view the histogram diagram by clicking on the 'Show Diagram' button.



Temporal Link Prediction Model for Weighted Dynamic Networks

This system is able to predict the validity of links within a given article.

pmid	doi	title	authors	year	journal	is_research_article	citation_count	field_citation_rate	expected_citations_per_year	citations_per_year
1	10.1111/bioe.12456	Tu Youyou winning the Nobel Prize: Ethical research on the value and safety of traditional Chinese medicine.	Wei-Rong Zheng, En-Chang Li, Song Peng, Xiao-Shang Wang	2020	Bioethics	TRUE	1			1
2	10.1093/ntr/nty159	Effects of Intranasal Oxytocin on Stress-Induced Cigarette Craving in Daily Smokers.	Kathryne Van Hedger, Anya K Bershad, Royce Lee, Harriet de Wit	2020	Nicotine Tob. Res.	TRUE	3	3.36160860107938	0.99999999987523	3
3	10.1007/s12350-018-1418-1	Electrical and mechanical dyssynchrony in patients with right bundle branch block.	Saara Sillanmäki, Sini Aapro, Jukka A Lipponen, Mika P Tarvainen, Tiina Laitinen, Marja Hedman, Hanna Hämäläinen, Tomi Laitinen	2020	J Nucl Cardiol	TRUE	4	4.54344158070733	0.99999999987523	4
4	10.1080/07448481.2018.1515758	An analysis of the sexual health and safety information study abroad directors present their students prior to departure.	Tiffany L Marcantonio, D J Angelone, Jill Swirsky, Meredith Joppa	2020	J Am Coll Health	TRUE	2	2.85890336832261	0.99999999987523	2
5	10.1007/s12350-018-1460-z	Electrical and mechanical dyssynchrony in patients with right bundle branch block.	Alejandro Solodky, Nili Zafrir	2020	J Nucl Cardiol	FALSE	0			0
6	10.1080/07481187.2018.1522386	Do you remember being told what happened to grandma? The role of early socialization on later coping with death.	Lucia Martinčeková, Matthew J Jiang, Jamal D Adams, David Menendez, Iseli G Hernandez, Gregory Barber, Karl S Rosengren	2020	Death Stud	TRUE	1	1.38492749835786	0.99999999987523	1
7	10.1080/10408398.2018.1545218	Efficacy of synbiotic supplementation in obesity treatment: A systematic review and meta-analysis of clinical trials.	Amir Hadi, Kimia Alizadeh, Hossein Hajianfar, Hamed Mohammadi, Maryam Miraghajani	2020	Crit Rev Food Sci Nutr	TRUE	7	5.37545810662628	0.99999999987523	7
8	10.1080/09638288.2018.1514536	Development of indicators to assure quality of disability evaluation based on the International Classification of Functioning, Disability, and Health in Taiwan: a Delphi consensus.	Kwang-Hwa Chang, Wen-Chou Chi, Hua-Fang Liao, Shih-Ching Chen, Hung-Yi Chiou, Reuben Escorpizo, Tsan-Hon Liou	2020	Disabil Rehabil	TRUE	1	1.49187098338632	0.99999999987523	1
9	10.1007/s40292-019-00313-9	RETRACTED ARTICLE: Vasodilatory Properties of Sacubitril/Valsartan Explored in Hypertensives Aged Over 55 Years: A Meta-Analysis.	Renato De Vecchis, Carmelina Ariano	2020	High Blood Press Cardiovasc Prev	FALSE	1	5.97890581156042	0.99999999987523	1
		Perioperative Ketorolac for Supracondylar Humerus Fracture in								

pmid	relative_citation_ratio	nih_percentile	human	animal	molecular_cellular	x_coord	y_coord	apt	is_clinical	cited_by_clin	cited_by	references	provisional
1			1	0	0	0	1	0.5	FALSE		29969150	1	No
2			1	0	0	0	1	0.75	FALSE		31792646 30085292 31563957	2	No
3			1	0	0	0	1	0.75	FALSE		30805046 32060855 30298370 30143955	3	No
4			1	0	0	0	1	0.5	FALSE		30257143 31429802	4	No
5			1	0	0	0	1	0.05	FALSE			3	No
6			1	0	0	0	1	0.5	FALSE		31140591	15	No
7			1	0	0	0	1	0.95	FALSE		31175629 30653773 32156153 32248805 30595036 31804340 30014150	7	No
8			1	0	0	0	1	0.5	FALSE		30596295	8	No
9			1	0	0	0	1	0.25	FALSE		30937854	9	No

Fig. 11. GUI- Show database, in this window the user can see the list of articles of the entire database.

7.2.3 Evaluation\Verification plan

Index	Current window	Test Description	Expected results
1	Home page window	Click "Start Prediction" button	Open the Train and Prediction window
2	Train and Prediction window	Click on the input and write the path of your dataset.	The path to the dataset is configured with the chosen value. If it's empty, then it goes to the default path: "/content/drive/MyDrive/Colab Notebooks/dataset/CITATIONS/04_2020_filtered-citation_metadata.csv"
3	Train and Prediction window	Click on the dropdown and choose value form the list.	Parameter is configured with the chosen value in the network
4	Train and Prediction window	Click "Next" button	Open the Train and Prediction logs window
5	Train and Prediction logs window	Building graph with the dataset and preforming prediction process	Building the network successfully
6	Train and Prediction logs window	Click "Show results" button	Open the results window
7	Results window	Click "Go to dataset" button	Open the dataset window
8	Results window	Analyze the generated graph	The system will display the predicted graph along with a list of the good and bad predicted edges.
9	Results window	Analyze the output error	The system will display the right result.
10	Results window	Analyze the Avg KL	The system will display the right result.
11	Dataset window	Rolled down	Show the dataset in a long table and go to the next rows.
12	Dataset window	Rolled up	Show the dataset in a long table and go to the previous rows.
13	Dataset window	Click "Close" button	The GUI will close.

8. Prediction Process

8.1 Dataset

Firstly, the model is applied to the UCSB dataset mentioned in the article. The acquired outcomes validate the accuracy of the article and illustrate the model's proficiency in generating weighted links of superior quality. Consequently, we can proceed with our research, focusing on link prediction between articles. The prediction process is performed by the iCite dataset [8]. The iCite dataset contains 41 snapshots, while each snapshot is a time stamp of a specific month and year. Each snapshot of the iCite is provided as a single zipped CSV file, or compressed, tarred JSON file while the metadata file provides information about the article. Definition for individual data fields:

- **pmid**: PubMed Identifier (article ID in PubMed).
- **doi**: Digital Object Identifier (if available).
- **year**: Publication year.
- **title**: Article title.
- **authors**: List of authors.
- **journal**: Journal name (ISO abbreviation).
- **is_research_article**: Flag indicating if it is a primary research article.
- **relative_citation_ratio**: Metric of scientific influence (RCR) adjusted for field and time.
- **provisional**: Provisional RCRs for recent articles with at least 5 citations.
- **citation_count**: Number of unique articles citing this one.
- **citations_per_year**: Average yearly citations since publication.
- **field_citation_rate**: Measure of intrinsic citation rate in the article's field.
- **expected_citations_per_year**: Average yearly citations for NIH-funded articles in the same field and year.
- **nih_percentile**: Percentile rank of RCR compared to all NIH publications.
- **human**: Fraction of MeSH terms in the Human category.
- **animal**: Fraction of MeSH terms in the Animal category.
- **molecular_cellular**: Fraction of MeSH terms in the Molecular/Cellular Biology category.
- **x_coord**: X coordinate on the Triangle of Biomedicine.
- **y_coord**: Y coordinate on the Triangle of Biomedicine.
- **is_clinical**: Flag indicating a clinical article.
- **cited_by_clin**: PMIDs of clinical articles citing this one.
- **apt**: Approximate Potential to Translate, an estimate of future clinical trial/guideline citations.
- **cited_by**: PMIDs of articles citing this one.
- **references**: PMIDs of articles in the reference list.

8.2 Pre-processing

The iCite dataset [8] contains many articles and their references, as mentioned above. In order to run the algorithm with the iCite metadata dataset, reducing was needed. Because the size of the original dataset is huge, and we couldn't run it in google collab pro or Kaggle. Thus, we reduce the dataset by few conditions: We considered articles published in the year 2020, the article's subject is only human and has references to another article and is referenced by another article. Subsequently, we generated a metadata file that encompassed the source and target nodes, representing the articles and their corresponding references. This file captured the overall topology of the nodes.

8.3 The Algorithm

The input of the model is a graph snapshot of various time slices. Therefore, we ran the model with several snapshots taken from the iCite dataset, after they perform the pre-processing mentioned in section 8.2. But the number of snapshots wasn't enough for prediction. Because of that, in the prediction process, we randomly assigned weights for each of the edges in the graph on specific snapshot. To create multiple snapshots for the model, we employed the same metadata file described earlier, incorporating randomly selected weights to ensure variability in the prediction process. Ultimately, the primary concern of the prediction process revolves around identifying unstable edges. An edge that has not been predicted frequently is considered unstable and unreliable. Consequently, such edges indicate a lack of connection between the two articles. In order to address the issue of unstable edges, a code was developed with the capability to predict them. The main challenge involved determining the appropriate error percentage for identifying unstable edges. During the training process, the number of incorrect predictions was accumulated for each edge. Specifically, in each training iteration, the difference between the predicted edges and the ground truth was calculated, considering a threshold of 0.01. Subsequently, in the evaluation phase, the average of incorrect predictions was computed for each edge. The final step involved checking if the percentage of erroneous predictions was below the specified 'error_percent'. Notably, determining the appropriate 'error_percent' proved to be a challenging task, as it was dependent on the number of iterations. The predicted edges were then printed and presented, with reliable edges identified as zeroes. Following the prediction process, the edges are printed and presented. Among these edges, the reliable ones are indicated as zeroes.

9. Results

In the following plots, we can see the results after running the model. Initially, we utilized the UCSB dataset mentioned in the article and obtained the same results as depicted. This indicates that the model validates the accuracy of the article and illustrates the model's proficiency in generating weighted links of superior quality. The results of running the model on the UCSB dataset:

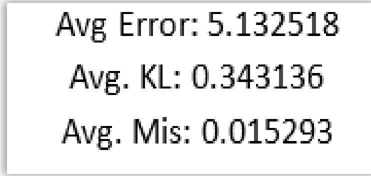

Avg Error: 5.132518
Avg. KL: 0.343136
Avg. Mis: 0.015293

Fig. 12. UCSB results

Then, we executed our model using the iCite metadata dataset. We conducted the algorithm multiple times with varying numbers of iterations, denoted by the variable "time_num", and we defined the threshold for distinguishing between good edges and bad edges as the percentage of error denoted by the variable "error_percent".

- The outcome achieved after performing 100 iterations (time_num=100) and considering a 0.005 percent error rate (error_percent = 0.005) is as follows:

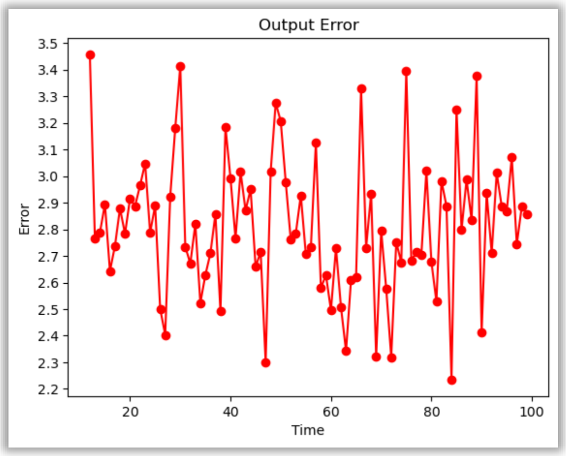


Fig. 13. Results Output Error for time_num=100, error_percent = 0.005

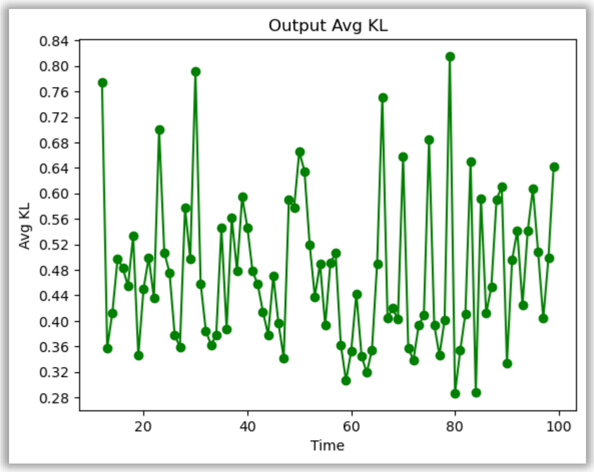


Fig. 14. Results Avg KL for time_num=100, error_percent = 0.005

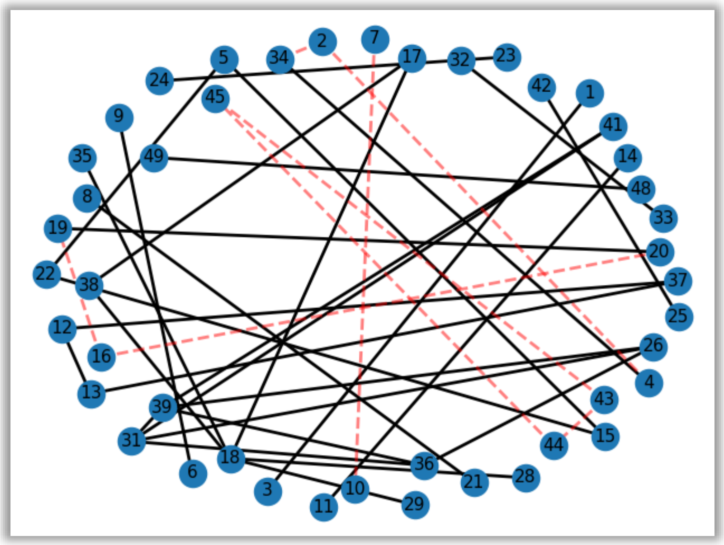


Fig. 15. Results Graph for time_num=100, error_percent = 0.005

<u>Running results:</u>

- The Avg. Error is 2.814330.
- The Avg. KL is 0.475736.
- The number of bad edges is 7, and the number of the good edges is 69.

- The outcome achieved after performing 400 iterations (time_num=400) and considering a 0.005 percent error rate (error_percent = 0.005) is as follows:

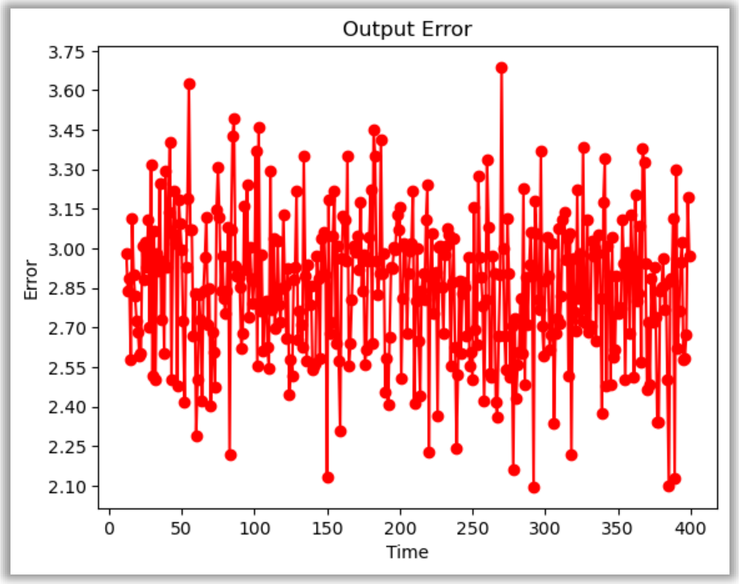


Fig. 16. Results Output Error for time_num=400, error_percent = 0.005

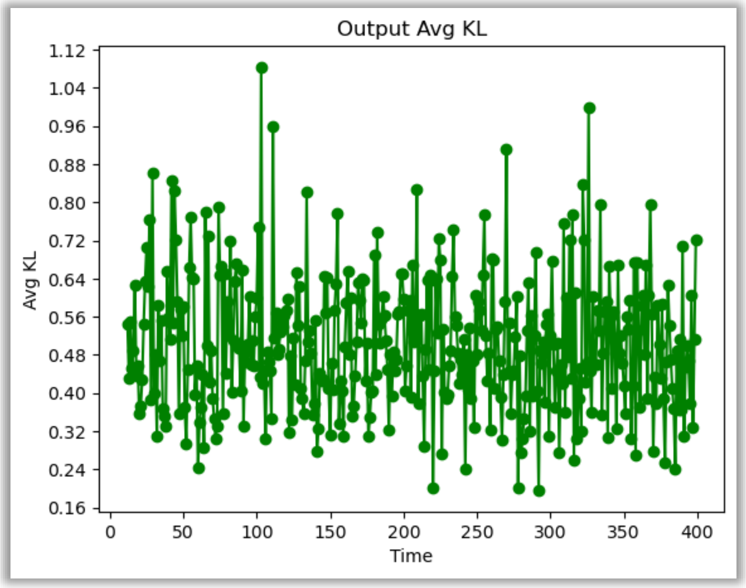


Fig. 17. Results Avg KL for time_num=400, error_percent = 0.005

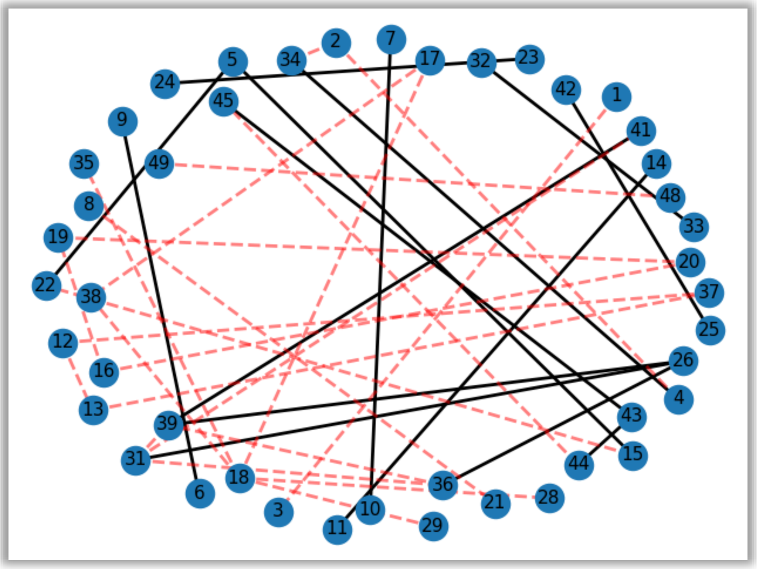


Fig. 18. Results Graph for time_num=400, error_percent = 0.005

Running results:

- The Avg. Error is 2.845889.
- The Avg. KL is 0.503244.
- The number of bad edges is 32, and the number of the good edges is 44.

To summarize the results and draw a conclusion, we provide an explanation for each output obtained:

The Graph reveals a significant disparity between the number of well-predicted edges and those that were inaccurately predicted as much as the value of time_num variable increases. The red edges represent edges that were not predicted accurately. As the prediction timeframe increases, the model presents a more significant number of bad (red) edges.

The Average KL represents the edgewise KL-divergence, which considers the disparity in link weights. It quantifies the Kullback-Leibler divergence for each iteration, reflecting the similarity between the predicted network and the ground-truth. At time_num = 100, the average KL value obtained was 0.475736, while at time_num = 400, the average KL value was higher and obtained 0.503244. In the context of average KL divergence, a smaller value indicates a better similarity between the predicted network and the ground-truth network. A larger value suggests a greater dissimilarity or divergence between the two networks.

The Output Error is representing the error present between the predicted network is to the ground-truth network. The charts demonstrate that the average Output Error is approximately 2.814 when time_num = 100 and around 2.845 when time_num = 400. In contrast, the UCSB dataset exhibits an output error of 5.13. These results are influenced by the specific dataset provided for analysis.

In summary, the Avg KL and the Avg Error are lower with 100 iterations and higher with 400 iterations. In addition, the model predicts less well as the number of iterations increases. We can explain this by that during the initial small number of iterations, the model may be learning quickly and adapting to the training data, which leads to a higher number of predicted edges. The lower output error and KL values suggest that the model is fitting the training data well. However, as the number of iterations increases, the model starts to become more complex and may begin to overfit the training data. Overfitting occurs when the model becomes too specialized in predicting the training data, and its performance on unseen or test data declines. Using random weights for snapshots in the model could potentially contribute to the overfitting behavior. Random weights can introduce initial randomness and variability to the model's predictions. During the initial iterations, when the model is trained with random weights, it may quickly adjust to the training data and make predictions that deviate less from the target values. This could result in lower output error and KL values, as the model begins to learn patterns in the data. However, as the number of iterations increases, the random weights might lead to the model fitting the noise or idiosyncrasies present in the training data. This can result in higher output error and KL values, as the model starts to overfit the training data.

10. User Instructions

To execute the model, the user needs to create a new blank notebook in Google Collab Pro and subsequently import/upload the project's notebook.

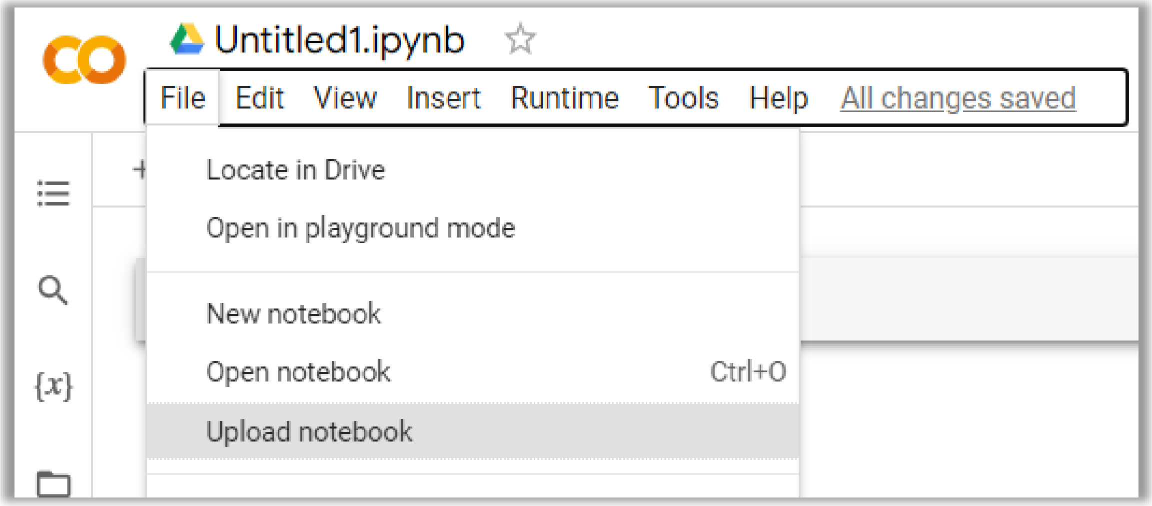


Fig. 19. Importing project's notebook.

The Python version is 3.10.12, and the TensorFlow version is 2.12.0, within the notebook. The user is required to navigate to the "Run the GUI" section and click on the play button. By doing so, the subsequent screen, known as the home page, will be displayed.

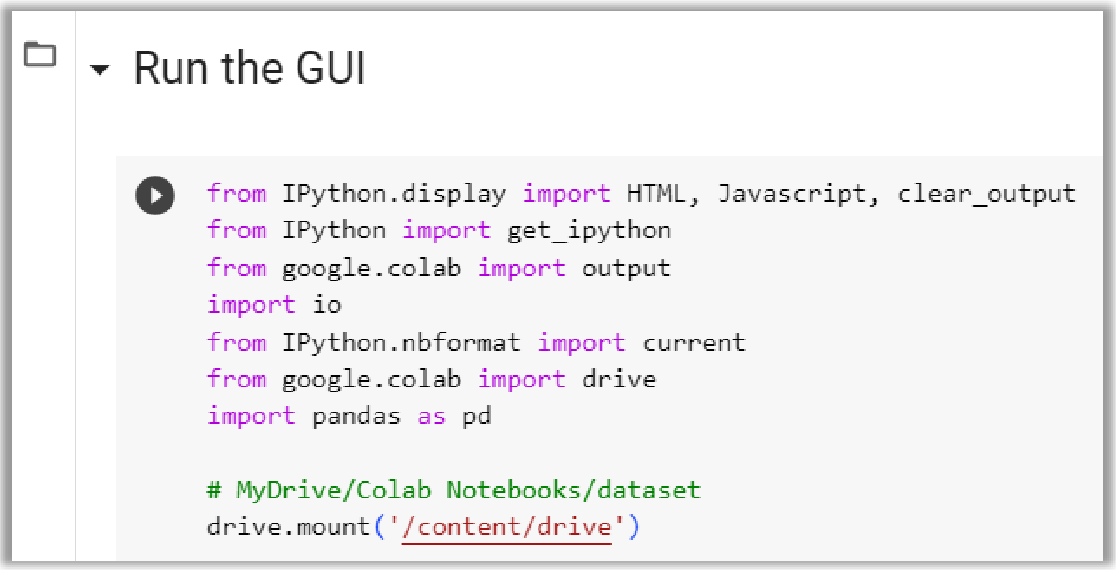


Fig. 20. Running the GUI.

Generally, the system relies on deep learning and is created to forecast connections between articles. User can click "start predict" in order to begin the prediction process with the default dataset named iCite metadata dataset.

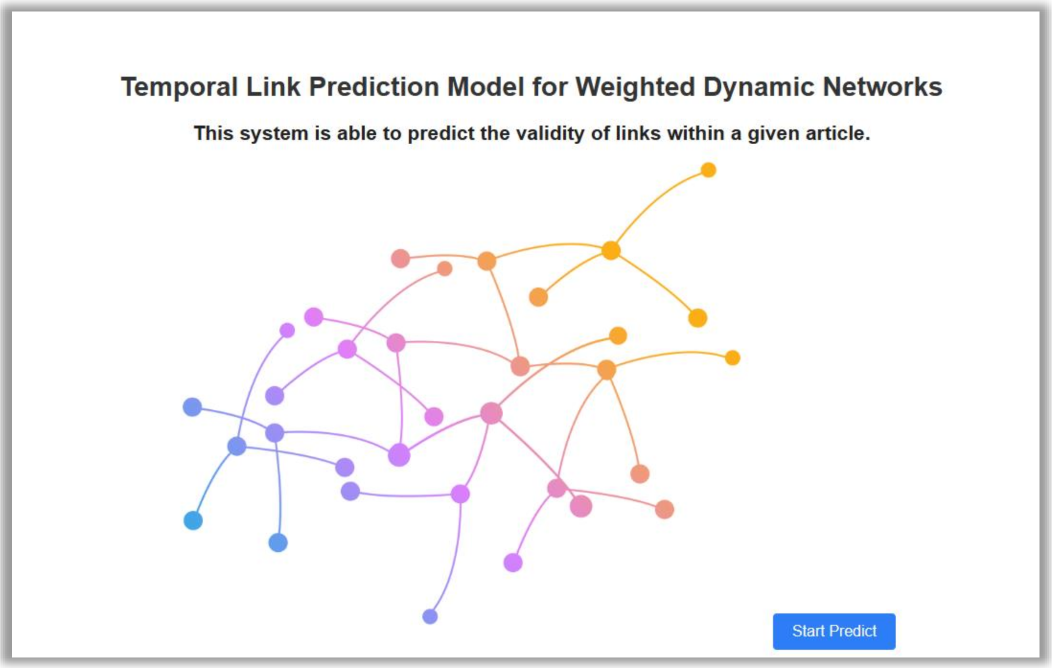


Fig. 21. User's home page.

The user is required to insert a link to the dataset -or- to place the dataset in the path that appears in the placeholder of the `Dataset Link`, in addition, can select different values for each of the model's parameters. Once the configuration is complete, clicking on the "Next" button initiates an immediate start to the prediction process.

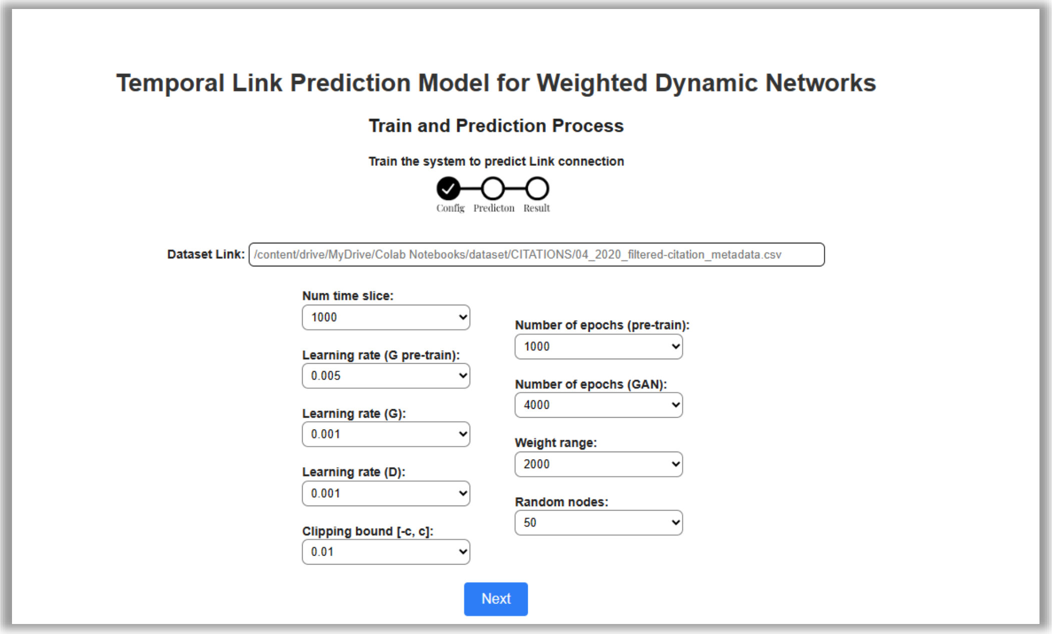


Fig. 22. User can Configure parameters.

Within this window, the user can view the logs of the ongoing prediction process.

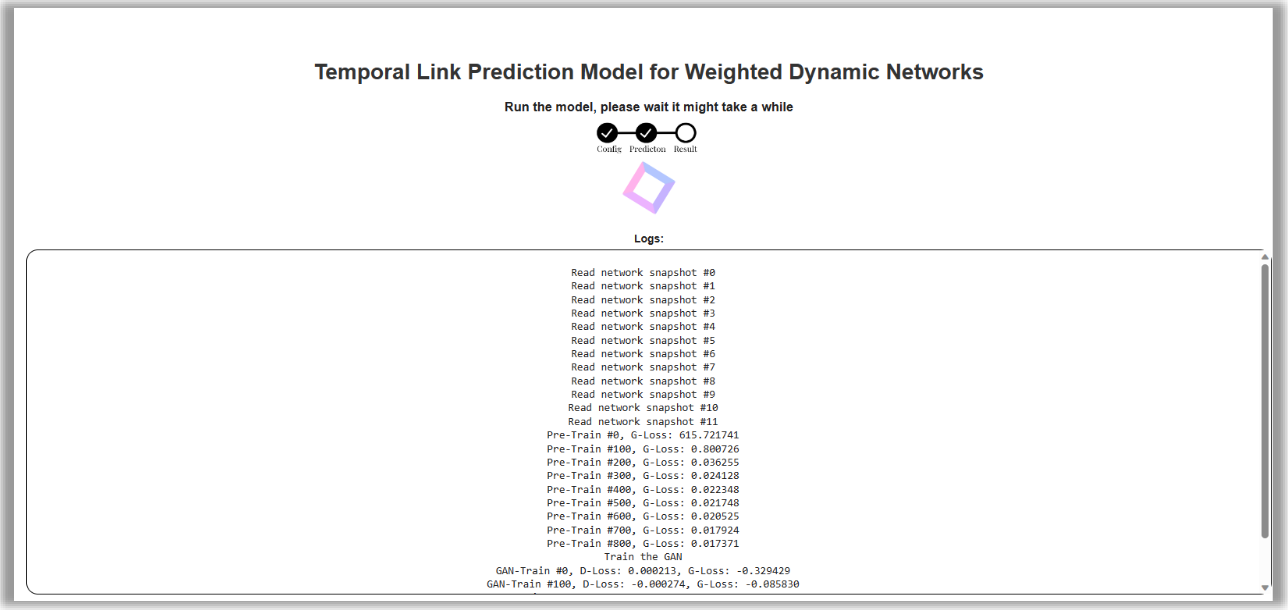


Fig. 23. User can see the logs.

Toward the end of the logs, the results of the last iteration will be displayed, including the error, edge weight KL, and mismatch rate. In addition, the user can view the result by scrolling inside the logs window. Once the process is complete, a new button is paper and the users have the option to click the "show results" button, which will open the charts depicting the prediction results.

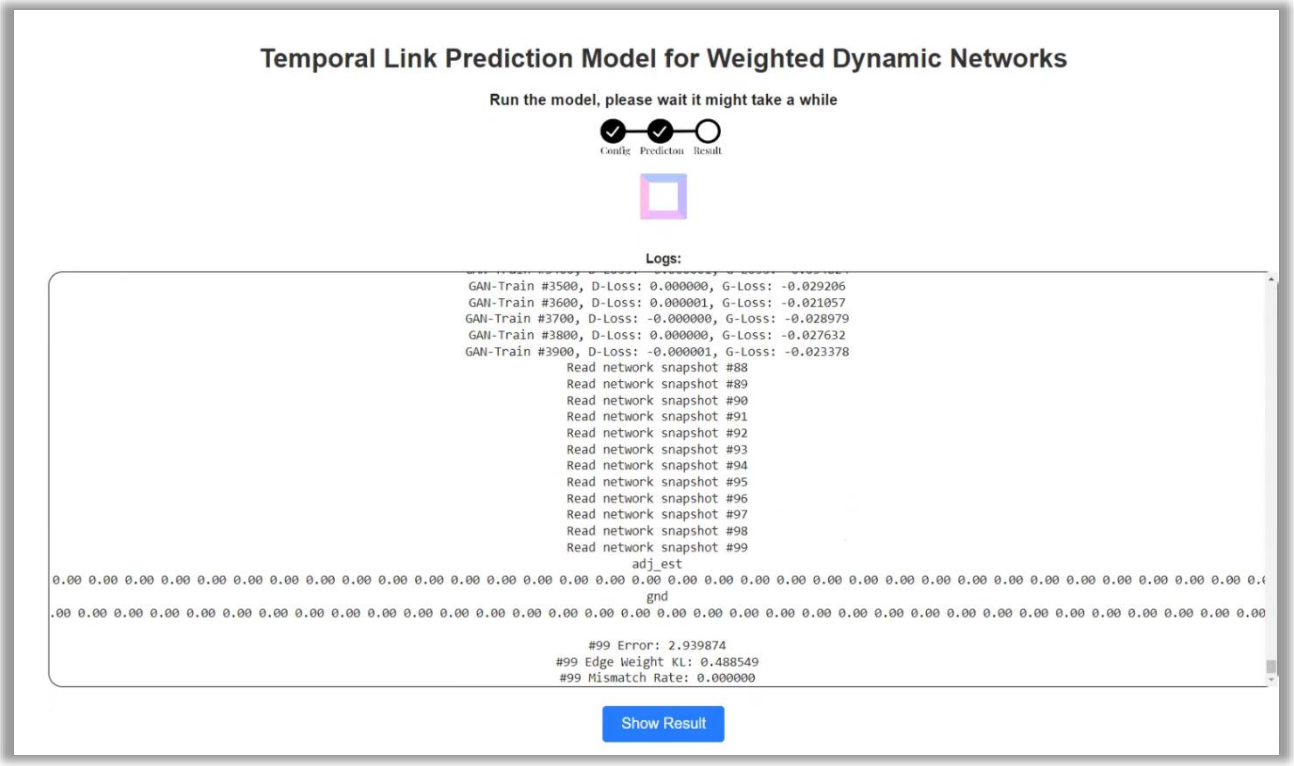


Fig. 24. User can choose show results when logs are done.

Within this screen, users have the ability to visualize all the results through charts.

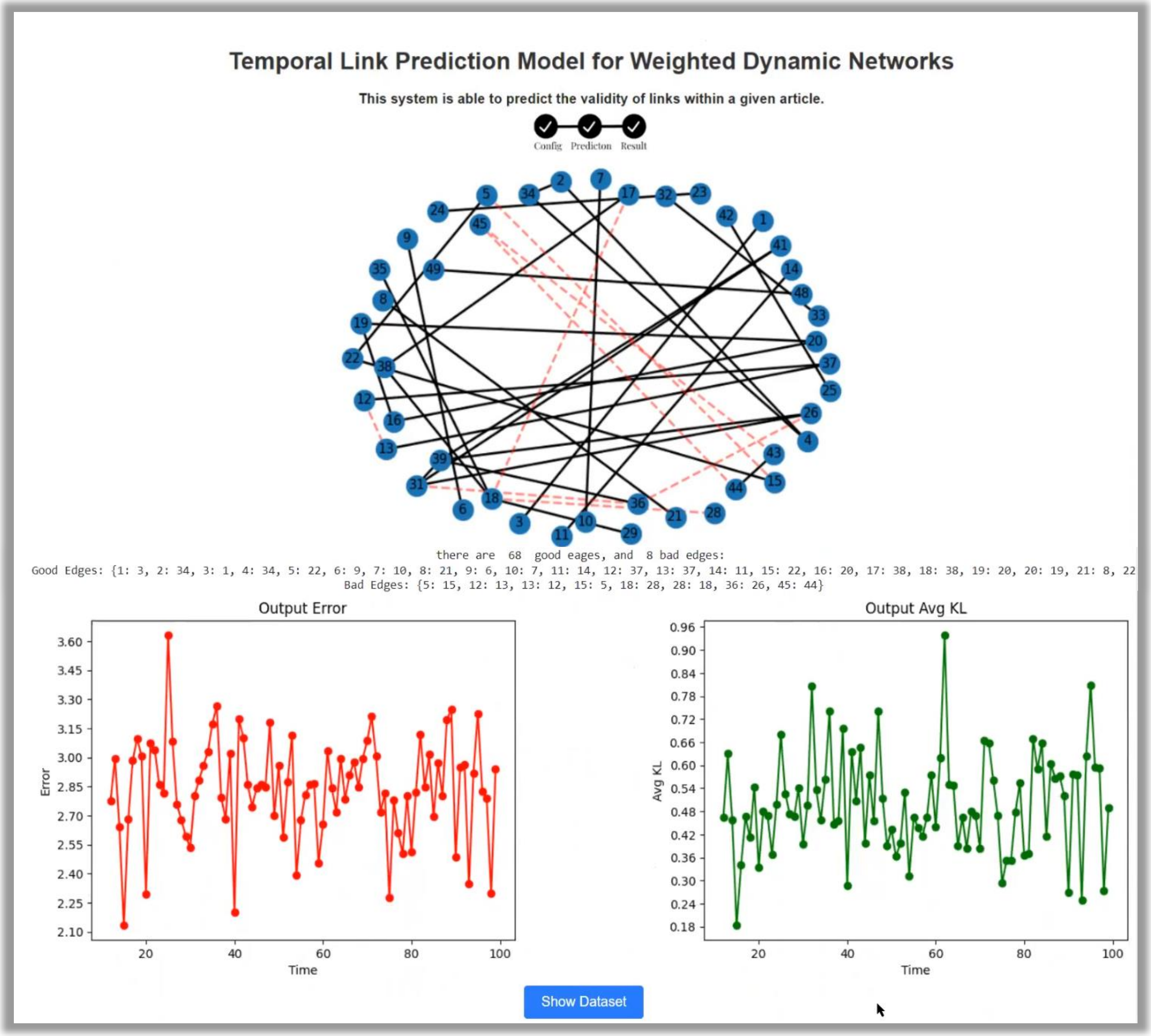
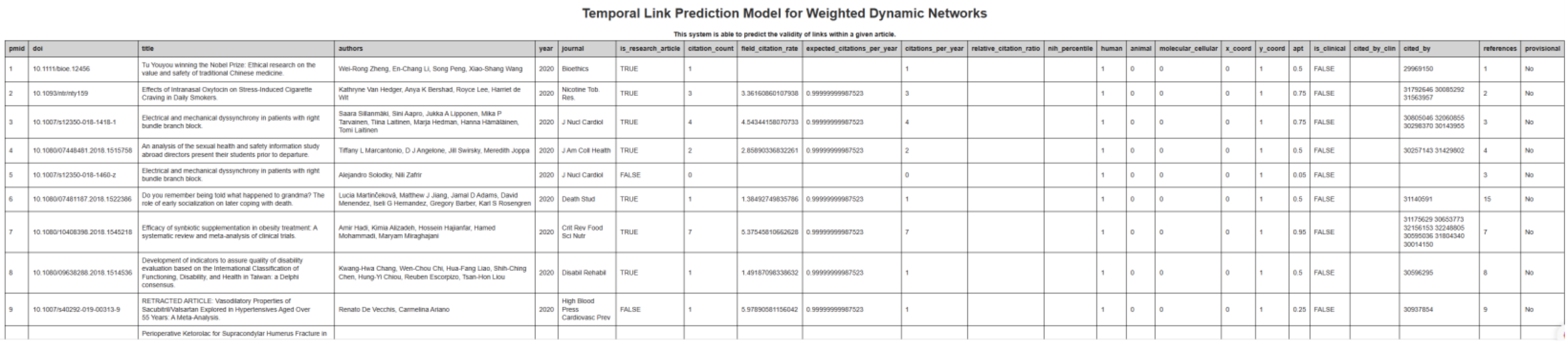


Fig. 25. User can see the results-, Graph., Output Error, Avg KL.

Towards the conclusion of this screen, users have the option to click on the "Show dataset" button to access the entire dataset, displayed in a table format, before any reduction or modification.



Temporal Link Prediction Model for Weighted Dynamic Networks

This system is able to predict the validity of links within a given article.

pmid	doi	title	authors	year	journal	is_research_article	citation_count
1	10.1111/bioe.12456	Tu Youyou winning the Nobel Prize: Ethical research on the value and safety of traditional Chinese medicine.	Wei-Rong Zheng, En-Chang Li, Song Peng, Xiao-Shang Wang	2020	Bioethics	TRUE	1
2	10.1093/ntr/nty159	Effects of Intranasal Oxytocin on Stress-Induced Cigarette Craving in Daily Smokers.	Kathryne Van Hedger, Anya K Bershad, Royce Lee, Harriet de Wit	2020	Nicotine Tob. Res.	TRUE	3
3	10.1007/s12350-018-1418-1	Electrical and mechanical dyssynchrony in patients with right bundle branch block.	Saara Sillanmäki, Sini Aapro, Jukka A Lipponen, Mika P Tarvainen, Tiina Laitinen, Marja Hedman, Hanna Hämäläinen, Tomi Laitinen	2020	J Nucl Cardiol	TRUE	4
4	10.1080/07448481.2018.1515758	An analysis of the sexual health and safety information study abroad directors present their students prior to departure.	Tiffany L Marcantonio, D J Angelone, Jill Swirsky, Meredith Joppa	2020	J Am Coll Health	TRUE	2
5	10.1007/s12350-018-1460-z	Electrical and mechanical dyssynchrony in patients with right bundle branch block.	Alejandro Solodky, Nili Zafrir	2020	J Nucl Cardiol	FALSE	0
6	10.1080/07481187.2018.1522386	Do you remember being told what happened to grandma? The role of early socialization on later coping with death.	Lucia Martinčeková, Matthew J Jiang, Jamal D Adams, David Menendez, Iseli G Hernandez, Gregory Barber, Karl S Rosengren	2020	Death Stud	TRUE	1
7	10.1080/10408398.2018.1545218	Efficacy of synbiotic supplementation in obesity treatment: A systematic review and meta-analysis of clinical trials.	Amir Hadi, Kimia Alizadeh, Hossein Hajianfar, Hamed Mohammadi, Maryam Miraghajani	2020	Crit Rev Food Sci Nutr	TRUE	7
8	10.1080/09638288.2018.1514536	Development of indicators to assure quality of disability evaluation based on the International Classification of Functioning, Disability, and Health in Taiwan: a Delphi consensus.	Kwang-Hwa Chang, Wen-Chou Chi, Hua-Fang Liao, Shih-Ching Chen, Hung-Yi Chiou, Reuben Escorpizo, Tsan-Hon Liou	2020	Disabil Rehabil	TRUE	1
9	10.1007/s40292-019-00313-9	RETRACTED ARTICLE: Vasodilatory Properties of Sacubitril/Valsartan Explored in Hypertensives Aged Over 55 Years: A Meta-Analysis.	Renato De Vecchis, Carmelina Ariano	2020	High Blood Press Cardiovasc Prev	FALSE	1
		Perioperative Ketorolac for Supracondylar Humerus Fracture in					

pmid	field_citation_rate	expected_citations_per_year	citations_per_year	relative_citation_ratio	nih_percentile	human	animal	molecular_cellular	x_coord	y_coord	apt	is_clinical	cited_by_clin	cited_by	references	provisional
1			1			1	0	0	0	1	0.5	FALSE		29969150	1	No
2	3.36160860107938	0.99999999987523	3			1	0	0	0	1	0.75	FALSE		31792646 30085292 31563957	2	No
3	4.54344158070733	0.99999999987523	4			1	0	0	0	1	0.75	FALSE		30805046 32060855 30298370 30143955	3	No
4	2.85890336832261	0.99999999987523	2			1	0	0	0	1	0.5	FALSE		30257143 31429802	4	No
5			0			1	0	0	0	1	0.05	FALSE			3	No
6	1.38492749835786	0.99999999987523	1			1	0	0	0	1	0.5	FALSE		31140591	15	No
7	5.37545810662628	0.99999999987523	7			1	0	0	0	1	0.95	FALSE		31175629 30653773 32156153 32248805 30595036 31804340 30014150	7	No
8	1.49187098338632	0.99999999987523	1			1	0	0	0	1	0.5	FALSE		30596295	8	No
9	5.97890581156042	0.99999999987523	1			1	0	0	0	1	0.25	FALSE		30937854	9	No

Fig. 26. User can see the dataset.

11. Conclusions

In this research, we propose a GCN-GAN model for non-linear temporal link prediction for weighted dynamic networks. The model will be utilized to predict the link between articles. Our project aims to determine whether the links in each article are meaningful and relevant based on the database of articles. The results obtained from applying the model to the UCSB dataset demonstrate its ability to generate weighted links of high quality, indicating the reliability of the article. Similarly, when the model was tested on the iCite metadata dataset, it produced favorable outcomes in terms of accurately predicting links between articles. To improve the results, acquiring additional data can be beneficial. Increasing the dataset size helps the model generalize more effectively and decreases the chances of overfitting. It's important to acknowledge that handling larger datasets may require substantial system resources, specifically in terms of RAM capacity. Moreover, monitor the model's performance during training and stop the training process when the validation error starts to increase. This prevents the model from overfitting by stopping it at the point where it performs best on unseen data.

12. References

[1] A. Mozo, B. Ordozgoiti, and S. Gomez, “Forecasting short-term data center network traffic load with convolutional neural networks”, Plos One, vol. 13, no. 2, p. e0191939, 2018.

[2] K. Lei, M. Qin, B. Bai, G. Zhang and M. Yang, "GCN-GAN: A Non-linear Temporal Link Prediction Model for Weighted Dynamic Networks", IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, 2019, pp. 388-396, doi: 10.1109/INFOCOM.2019.8737631.

[3] Y. Liang, L. Huang and Z. Wang, "Link prediction in social network based on local information and attributes of nodes", IOP Conf. Series: Journal of Physics: Conf. Series 887 (2017) 012043

[4] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks”, 2016.

[5] F. A. Gers, J. A. Schmidhuber, and F. A. Cummins, “Learning to forget: Continual prediction with lstm”, Neural Computation, vol. 12, no. 10, pp. 2451–2471, 2014.

[6] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein gan”, 2017.

[7] F. Zou, L. Shen, Z. Jie, W. Zhang, W. Liu, "A Sufficient Condition for Convergences of Adam and RMSProp", Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 11127-11135

[8] iCite dataset: figshare "https://nih.figshare.com/search?q=:keyword:%20%22Citation%20data%22"

13. Git

Link to our git repository:

https://github.com/TalChenBE/Temporal-Link-Prediction-Model-for-Weighted-Dynamic-Networks

In the repository, you will find the Python code file for the model, along with a notebook specifically designed for Google Collab usage. This notebook can be directly accessed and utilized within the Google Collab environment. Furthermore, we have included additional notebooks that provide a comprehensive review of our project's results, which have been presented in their entirety. Moreover, we have created a "Datasets" folder within the repository, containing the two datasets we utilized: USCB and the reduced iCite snapshot. Additionally, we have included a copy of the accompanying book and the project's presentation within the repository for your reference. To facilitate the usage of the code and materials, we have included a readme file that provides instructions on how to effectively utilize the resources, as explained in Section 10: User Instructions.