# CSC 591-605 Graph Data Mining Capstone Project: Edge Weight and Sign Prediction in WSGs using Weighted Signed Graph Convolutional Network

Ameya Dhavalikar
Vivek Reddy Karri
Yashwanth Reddy Soodini

December 12, 2019

## 1 Introduction

The number of online communities in the internet has grown rampantly in the recent years. These communities can often be modelled as signed graphs. A signed graph is a graph whose weights can be labelled as being positive and negative. The meaning of the sign of an edge depends on the community being modeled. For example, if we are modeling a social networking community, then the sign of an edge between two users would correspond to whether the users like or dislike each other, with a positive sign indicating they like each other and a negative sign indicating they dislike each other. However, in most real world situations, people like or dislike each other with varying intensities. For instance, Linda may not like her sister Jennifer as much as her friend Carol. To model these varying intensities, we need a weighted signed graph. A weighted signed graph (WSG) is a graph whose edges have weights and can labelled as being positive and negative. So in the context of the social networking community discussed earlier, a with a high positive weight would indicate the users like each other a lot and a low positive weight would indicate the users like each other comparitively lower.

In this project, we explore the graph mining task of predicting the weights of edges in weighted signed graphs. The Bitcoin exchange networks (OTC and Alpha), the Slashdot social network, the Epinions review community, and the Wikipedia networks are a few examples of WSGs whose data is openly available on the internet [5]. Most of these graphs however are incomplete, i.e., they only give explicit relations/ratings. However, Prediciting implicit relations in highy useful. In a social network, me might want to suggest new friends. This can be done by predicting Edge Weight b/w friends of her friends and setting certain threshold on predicted weight. In a who-trusts-whom network like the Bitcoin Alpha and the Bitcoin OTC networks, we might wanna find out whether a user A implicitly trusts user B, even if user A has never explicitly rated user B. This can also be done by predicting the edge weight between the users. Edge weight prediction in WSGs can also be used in conjunction with other graph mining tasks like anomaly detection, community detection, and network analysis, among others.

Our Literature review on this topic goes from Spectral Graph Theory to Geometric Deep Learning: (1) Fairness-Goodness Model, (2) Signed Spectral Embedding, and (3) Signed Graph Convolutional Network (SGCN).(Order specified as per the chronological order of the publication). We have proposed a Weighted Signed Graph Convolutional Network (WSGCN), which is a generalization of SGN's for signed graphs that have weighted edges (WSGs). Finally, We have conducted out the experiments on the Bitcoin-Alpha Dataset and Bitcoin-OTC Dataset, obtained from the SNAP Data Collection.

## 2 Related Work

**Fairness-Goodness Model.** [3]The aim of this paper is to predict the weights of edges in weighted signed networks using metrics like Fairness and Goodness. Weighted signed networks are networks in each the edges are assigned weights that are either positive or negative. This helps capture relationships such as trust/distrust, like/dislike, etc. Fairness of a vertex(rater) in a graph is a measure of how fair or reliable the vertex is in assigning ratings(such as like/dislike, agree/disagree, trust/distrust and so on) to other vertices. From a 'fair'/'reliable' rater, a user is expected to receive the rating that is deserved, otherwise the rater is unfair and such a rating can be treated of low importance. Goodness of a vertex specifies how much other vertices like/dislike, trust/distrust or agree/disagree with that vertex and what is its true quality. A good rating is a rating that would be given by a totally fair vertex. Hence, both the metrics, goodness and fairness, depend on each other. The Fairness Goodness algorithm gives the process through which these two metrics are calculated iteratively. The algorithm converges in linear time to the size of the network. The predicted weight of an edge(u,v) is simply the product of f(u)*g(v). This score is hence called the F X G score of the edge.

**Signed Spectral Embedding.** [4]Signed Spectral Embedding is a spectral clustering algorithm for signed graphs. For a graph which has only positively weighted edges, laplacian matrix is a symmetric, positive-semidefinite matrix which captures the relationships between individual nodes of the graph and is useful for finding many of the graph's properties. For a signed graph however, the laplacian matrix, if calculated in the ordinary fashion will result in an indefinite matrix. Therefore, we use a modified version of the degree matrix $D$ for calculating the Laplacian matrix of a signed graph, which we call the signed laplacian matrix.

The signed Laplacian Matrix $\bar{L}$ of a graph $G$ is given by

$$\bar{L} = \bar{D} - A$$

where A is the adjacency matrix of the graph and $\bar{D} \in \mathbb{R}^{V \times V}$ is the signed degree matrix of the graph given by

$$\bar{D}_{ii} = \sum_{j \sim i} |A_{ij}|.$$

The matrix thus obtained is proved to be a positive semidefinite matrix in the original paper by Kunegis et al. It can therefore be used for constructing low dimensional embeddings of nodes in a signed graph. For an $m$-dimensional embedding in the euclidian space, we sort the eigenvectors in ascending order, leave the zero eigenvectors and use the next $m$ corresponding eigenvectors for embedding.

**Signed Graph Convolutional Network.**[2] A Graph Convolutional Networks is a generalization of the theories specified by CNN's onto Graph Structures which are non-euclidean. GCN's are used to generate Node Embeddings which can efficiently and easily used by the well defined techniques of Machine Learning. A SGCN is a GCN specified for Signed Graphs. SGCN utilizes balance theory to compute the Embeddings. Balance theory classifies cycles in a signed network as either balanced or unbalanced. Balance cycles are those which have an even number of negative links whereas unbalance ones are those which have odd number of negative links. The authors propose definitions for balanced and unbalanced paths in an analogous fashion i.e., balanced paths as having even number of negative links and unbalanced paths as having odd number of negative links. Users that can be reached from a user $u_i$ along a balanced path of length $l$ are placed in the set $B_i(l)$. Similar definition holds for $U_i(l)$. $\mathcal{N}_i^+$ and $\mathcal{N}_i^-$ are the sets of positive and negative neighbours of $u_i$ respectively.

The balanced and unbalanced sets can be calculated recursively. Using these sets, nodes are allowed to incorporate information from a multi-hop neighbourhood.

# 3 Proposed Framework (Weighted SGCN

**Algorithm 1:** Weighted Signed GCN Embedding Generation

**Input:** $\mathcal{G} = (\mathcal{U}, \mathcal{E}^+, \mathcal{E}^-)$; an initial seed node representation $\{\mathbf{x}, \forall u_i \in \mathcal{U}\}$;
number of aggregation layers L; weight matrices $\mathbf{W}^{B(l}$ and $\mathbf{W}^{U(l}, \forall lin1, ..., L$; non-linear function $\sigma$

**Output:** Low-dimensional representations $\mathbf{z}_i, \forall u_i \in \mathcal{U}$

$\mathbf{h}_i^{(0)} \leftarrow \mathbf{x}_i, \forall u_i \in \mathcal{U}$

**for** $u_i \in \mathcal{U}$ **do**

$\quad \mathbf{h}_i^{B(1)} \leftarrow \sigma\left(\mathbf{W}^{B(1)}\left[\sum_{j\in\mathcal{N}_i^+} \frac{|A_{ij}^+|\times h_j^{(0)}}{\sum_{j\in\mathcal{N}_i^+}|A_{ij}^+|}, h_i^0\right]\right)$

$\quad \mathbf{h}_i^{U(1)} \leftarrow \sigma\left(\mathbf{W}^{U(1)}\left[\sum_{k\in\mathcal{N}_i^-} \frac{|A_{ik}^-|\times h_k^{(0)}}{\sum_{k\in\mathcal{N}_i^-}|A_{ik}^-|}, h_i^0\right]\right)$

**end**

**if** $L > 1$ **then**

$\quad$ **for** $l = 2...L$ **do**

$\quad\quad$ **for** $u_i \in \mathcal{U}$ **do**

$$\mathbf{h}_i^{B(l)} = \sigma\left(\mathbf{W}^{B(l)}\left[\sum_{j\in\mathcal{N}_i^+} \frac{|A_{ij}^+|\times h_j^{B(l-1)}}{\sum_{j\in\mathcal{N}_i^+}|A_{ij}^+|}, \sum_{k\in\mathcal{N}_i^-} \frac{|A_{ik}^-|\times h_k^{U(l-1)}}{\sum_{k\in\mathcal{N}_i^-}|A_{ik}^-|}, \mathbf{h}_i^{B(l-1)}\right]\right)$$

$$\mathbf{h}_i^{U(l)} = \sigma\left(\mathbf{W}^{U(l)}\left[\sum_{j\in\mathcal{N}_i^+} \frac{|A_{ij}^+|\times h_j^{U(l-1)}}{\sum_{j\in\mathcal{N}_i^+}|A_{ij}^+|}, \sum_{k\in\mathcal{N}_i^-} \frac{|A_{ik}^-|\times h_k^{B(l-1)}}{\sum_{k\in\mathcal{N}_i^-}|A_{ik}^-|}, \mathbf{h}_i^{U(l-1)}\right]\right)$$

$\quad\quad$ **end**

$\quad$ **end**

**end**

$\mathbf{z}_i \leftarrow \left[\mathbf{h}_i^{B(l)}, \mathbf{h}_i^{U(l)}\right], \forall u_i \in \mathcal{U}$

We note that in SGCN, the edge-weights values are totally ignored in the training process and just the sign (+1 or -1) is used. Also, SGCN generates embeddings that can only be used for predicting the sign of the edges, but not their weights. For effective weight prediction or even Sign Prediction , incorporating the information regarding the weights into the framework is a must.

We now propose Weighted Signed Graph Convolutional Networks, a generalization of SGCN's for graphs that have signed as well as weighted edges. The definitions of balanced and unbalanced sets remained unchanged from SGCN. Similar to SGCN, while aggregating and propagating information in our WSGCN, we are going to maintain two separate representations at every layer, one for the corresponding balanced set of users and one for the unbalanced set. We use $h_i^{(0)} \in \mathbb{R}^{d^{in}}$ to represent the initial $d^{in}$ node features for user $u_i$. Note that instead of calculating the mean of the users in the set $B_i$, we are calculating a weighted mean. So in the final embedding, if two nodes have a strong positive link between them, their corresponding embeddings will be very close and should result better performance in downstream tasks.

Now that we have discussed the aggregation methods, the entire WSGCN framework is presented in the paper. We are using the same objective function used in the SGCN framework for learning the parameters.

# 4 Evaluation Methodology

0) Data Cleaning and Division of the Edge Set into train and test sets (85-15) Split Ratio. (Statified Splitting to handle label Imbalance)

1) Generation of Initial Embeddings using Modified SSE described in Section II on Train Set
2) Generated Lower Dimensional and Graph Structure information infused Embeddings using WSGCN.
3) Edge Sign Prediction (modelled as a Classfication Problem) Using an ANN trained on Embeddings generated by SGCN and validation on Test Dataset. (Used Balanced Binary-Cross Entropy i.e. Penalizing a wrong prediction of a negative sign over a wrog prediction of positive sign, Again to handle label imbalance)[1]
4) Edge Weight Prediction (modelled as a Regression Problem) using a comparitively Deeper ANN on the embeddings generated by SGCN and Validation on test set. [1]

Note 1: For the last two tasks, the prediction of nodes is done by concatenating the embeddings and then applying the appropriate algorithm

Note 2: There are only 20 unique and discrete Labels present as the edge weights in both the datasets. However any classification loss function would take +10 and +9 as the exact same weights as +10 and -10 which in reality are farther away. On the other hand, having just 20 unique values for training a regression model for a set of around 20k-30k training examples would lead in the values of R2 would be very low (order of negative hundreds). So we have introduced a random noise sampled from Gaussian distribution of mean 0 and Standard deviation of 0.5. This would result in a relative spread in the prediction space. This would work as we don't need fine grained predictions and any prediction in the range of 0.5 to 1.5 can be rounded to as being 1.

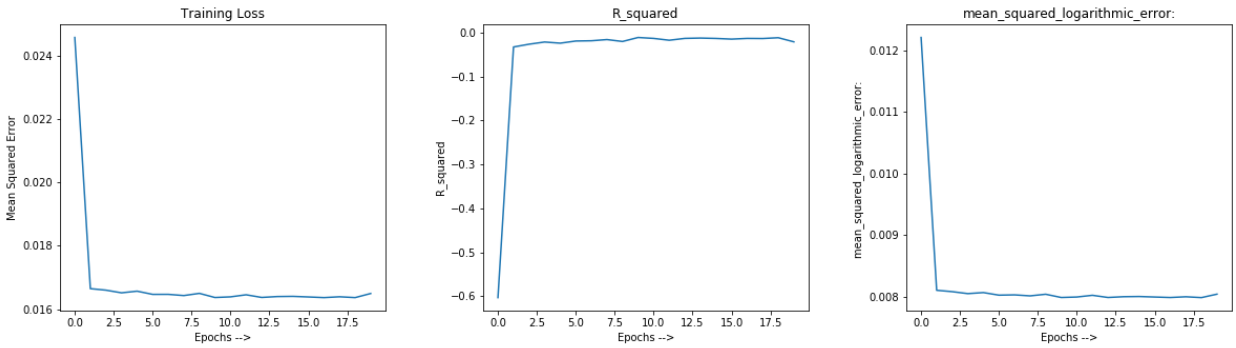
Note 3: Details of Hyper-Parameters used can be found in the exec.py file of the source code.

# 5 Results

Table 1: Classification Metrics for Edge Sign Prediction

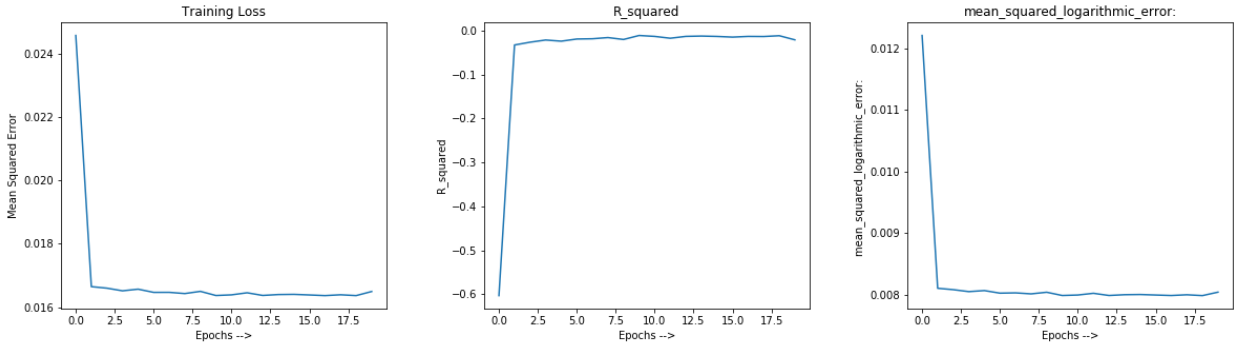
| Dataset | Split | Precision | Recall | Weighted Log loss |
|---|---|---|---|---|
| Alpha | Train | 1.0 | 0.9364724 | 0.142105 |
| Alpha | Test | 1.0 | 0.9366042 | 0.141941 |
| OTC | Train | 1.0 | 0.8998777 | 0.182629 |
| OTC | Test | 1.0 | 0.8999812 | 0.182534 |

Figure 1: Training Metrics for Edge Weight Prediction on Alpha




Discussion: 1) As it can be observed from Table 1, infusing Weight information did not affect the Edge Sign Classification and the results are promising

Figure 2: Training Metrics for Edge Weight Prediction on OTC




2) After noise addition into the labels for dependable regression model, R2 value has rose from negative hundreds to almost zero. Although, having a good R2 doesn't necessarily mean that the model is good and reverse is also not true. Also, Getting a good R2 is such higher dimensional spaces (256-dim) is relatively hard. Further investigation can be went onto using dim-reduction techniques on generated Embeddings to search for performance gains.
3) Geometric Deep Learning is a nascent field of research and many more ways of applying DNN models to Graphs can also be Generalized to Signed and Weight Graphs.

## References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] T. Derr, Y. Ma, and J. Tang. Signed graph convolutional networks. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 929–934, Nov 2018.

[3] Srijan Kumar, Francesca Spezzano, V.S. Subrahmanian, and Christos Faloutsos. Edge weight prediction in weighted signed networks. In *Data Mining (ICDM), 2016 IEEE International Conference on*, 2016.

[4] Jérôme Kunegis, Stephan Schmidt, Andreas Lommatzsch, Jürgen Lerner, Ernesto W. De Luca, and Sahin Albayrak. *Spectral Analysis of Signed Graphs for Clustering, Prediction and Visualization*, pages 559–570.

[5] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.