# Social Network Deanonymization

Zhou Ben 515030910610

May 25, 2018

## 1 Abstract

In this project, we proposed two methods based on **neural networks** to address the **social network deanonymization problem**. The first method is based on supervised learning, in which we trained a network to learn the match between nodes. Its two variations **balanced supervised learning** and **extended supervised learning** have gotten an AUC of 0.84 and 0.79 respectively. The second method is based on unsupervised learning, which receives positive results on small networks.

## 2 Introduction

The social network deanonymization problem is first addressed by Narayanan[1] in 2009, who proved that attackers can reveal the users' private information from public datasets from social platforms with the help of an auxiliary network, even if the companies have already tried to remove or conceal personal information. One typical result of a successful deanonymization attack is that attackers can match the nodes in one network against another, which directly confers personal information leakage.

Since then, deanonymization and anonymization techniques has developed like playing a chasing game[2]. A stronger anonymization algorithm is developed, and then it is breached with certain deanonymization algorithm. It is not certain who will be the final winner, but it is sure that the development of deanonymization algorithms can spur the growth of the anonymization theory and private information protection policy.

## 3 Problem Formulation

In this section, we give our precise definition of the social network deanonymization problem.

Given a undirected graph $G = (V, E)$ representing a social network. $G$ represents the real (and possibly latent) social interpersonal relationship.

Let $G_k = (V_k, E_k), k = 1, \cdots, K$ denote several samples of G, where

$$V_k \subseteq V, E_k \subseteq E$$

$G_k$ represents the interpersonal relationship reflected in various social network data sources. Define $\phi_k : V \to V_k$ to describe the change happened in this sampling process (including shuffling and/or anonymization). Our target is to find the matching of nodes across different subgraphs.

Target:

$$\text{Find } \mathscr{M}, \mathscr{I}, s.t. \ \forall u \in V_{k_1}, v \in V_{k_2}$$
$$\mathscr{M}(\mathscr{I}(u), \mathscr{I}(v)) = \begin{cases} 1, \phi_{k_1}^{-1}(u) = \phi_{k_2}^{-1}(v), \\ 0, \text{otherwise} \end{cases}$$

Here, $\mathscr{M}$ is the signal function, and $\mathscr{I}(u)$ is the various information around a node used to match nodes. In most case, such ideal function $\mathscr{M}$ can not be found, and different methods give different approximation of it.

$\phi$ is not always invertible (in fact anonymization is intended to be non-invertible), $\phi^{-1}$ here is just to denote the original node.

# 4 Related Work

In **Table.1**, we summarize the different deanonymization methods according to the $\mathscr{M}$ and $\mathscr{I}$ they used.

It can be seen that neural networks haven't been used widely in the social netowrk deanonymization problem. So, this project makes the attempt.

# 5 Deanonymization Based on Supervised Learning

In this section, we introduce our supervised learning based method. A forward neural network is trained to learn the similarity between two nodes. We denote the output by $\mathscr{N}(\mathcal{F})$, where $\mathcal{F}$ is the input.

Generally speaking, neural networks can not handle matching problems, since the output space changes with different inputs. To circumvent this problem, we convert the matching problem to classification problem with two classes.

| | What does $\mathscr{I}$ include | How is $\mathscr{M}$ found |
|---|---|---|
| NS[1] | topology; common-neighbor counts | iteratively found from seeds to all nodes |
| Community-Enhanced NS[3] | topology; community designation; | first match communities, then within communities |
| HYDRA[4] | user attributes; user generated content; user behavior trajectory; user core social network features; | supervised learning based on real world information |
| Latent User Space[5] | user attributes | supervised learning based on real world information |
| Max A Posteriori[6] | topology; community designation; | the equation of analytic solution is given by MAP, then solved with approximation algorithm |
| Random Forest[2] | degrees of 1-hop neighbors | random forest |

Table 1: Summary of deanonymization methods

$$\text{Matching: } \mathscr{N} : V \to V$$
$$\text{Classfication: } \mathscr{N} : V \times V \to \{0, 1\}$$

This means the network gets a node pair $(u, v)$ as inputs. If the output should be 0, $u$ and $v$ are predicted to be non-identical, 1 as identical.

One problem of this transformation is **label imbalance**. For a network with $n$ nodes, there exist $n$ correct node matching but $n(n-1)$ wrong ones. This results in positive feedback getting overwhelmed by negative ones. In this project, two methods are tested to address this problem. In the first method, only $n$ negative samples are selected at random to balance the dataset. In the second method, the inputs and outputs are further expanded to $(u, v_1, v_2) \to \{0, 1\}$. In this case, an output of 1 implies $(u, v_1)$ is a better match than $(u, v_2)$. We name the first model as **balanced supervised learning**, the second as **extended supervised learning**.

## 5.1 Feature Vector

We use a feature vector $\mathcal{F}(u)$ to represent a node $u$. The node feature vector used here is based on the feature used by Sharad [2]. In that work, each node is represented with its neighbors' degrees.

$$\forall u \in V, \mathcal{F}(u) \in \mathbf{R}^{F_n}$$

$$\mathcal{F}(u)_i = |\{v : (u,v) \in E, iF_b \leq deg(v) < (i+1)F_b\}|, i = 1, \cdots, F_n$$

In our case, $F_n = 10$, $F_b = 10$.

### balanced supervised learning

Two node vectors are concatenated together to be input of the network.

$$\mathcal{F}_{\text{bal}}(u,v) = \{\mathcal{F}(u), \mathcal{F}(v)\}$$

### extended supervised learning

Three node vectors are concatenated in row.

$$\mathcal{F}_{\text{ext}}(u, v_i, v_j) = \{\mathcal{F}(u), \mathcal{F}(v_i), \mathcal{F}(v_j)\}$$

## 5.2 Network Structure

There is no need to resort to deep networks since the feature vector is simple. In our case, a 4-layer network is used, with 32, 32, 8, 2 hidden units in each layer respectively. Dropout with a rate of 0.5 is used after the first 2 layers. See Fig. 1.

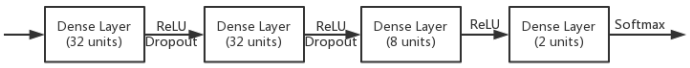


Figure 1: Network Structure for Supervised Learning

## 5.3 Training

### balanced supervised learning

A synthetic ER graph $G = (V, E)$ with $|V| = 100$ and a density of 0.3 is generated to represent the social network.

Two subgraphs $G_1, G_2$ are sampled by randomly discard edges at a probability of 0.8.

There is no need to shuffle the nodes order explicitly as [1] did since nodes pairs are feeded into the network without global information.

The training dataset $\mathcal{X}$ is composed of all of the 100 identical node vector pairs $\mathcal{X}_1 = \{\mathcal{F}_{\text{bal}}(u_i, v_i) : u_i \in V_1, v_i \in V_2, i = 1, \cdots, 100\}$ with label 1 and 100 non-identical node vector pairs $\mathcal{X}_2 = \{\mathcal{F}_{\text{bal}}(u_i, v_j) : u_i \in V_1, v_j \in V_2, i \neq j\}$ selected at random with label 0.

#### extended supervised learning

Subgraphs with identical parameters are generated.
The training dataset $\mathcal{X}$ is composed $\mathcal{X}_1 = \{\mathcal{F}_{\text{ext}}(u_i, v_i, v_j) : u_i \in V_1, v_i, v_j \in V_2, i = 1, \cdots, 100, j \neq i\}$ with label 1 and $\mathcal{X}_2 = \{\mathcal{F}_{\text{ext}}(u_i, v_j, v_i) : u_i \in V_1, v_i, v_j \in V_2, i = 1, \cdots, 100, j \neq i\}$ with label 0.

### 5.4 Evaluation

A new graph with identical parameters is generated for evaluation.

Training result: an ROC curve. See Fig. 2.

#### analysis

It can be seen from the ROC curve and the experiment that:

- Both variations give a positive result on prediction
- Extended supervised learning proves to be better than balanced supervised learning, the reason can be possibly attribute to the richer dataset the extended scheme possesses.
- Though extended supervised learning is slightly better, it takes much more time to construct the training dataset.

## 6 Deanonymization Based on Unsupervised Learning

Supervised learning requires a training dataset already known to the attacker, which is sometimes too high a requirement. In this section, a neural network based unsupervised learning method is proposed.

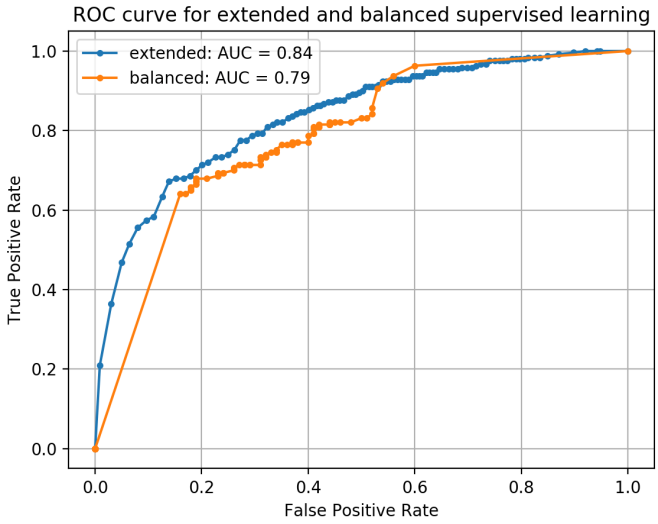


Figure 2: ROC curve for supervised learning

## 6.1 Estimation based on MAP

Define $\pi$ to a permutation matrix representing the node matching, and $\pi_0$ to be correct match. Onaran[7] has already form the deanonymization problem into a max a posteriori (MAP) optimization problem under the community model (which is not concerned in this project) and given the objective function:

$$\hat{\pi} = \arg\max_{\pi \in \Pi} P(\pi = \pi_0 | G_1, G_2, \theta, \mathcal{C})$$

$$\hat{\pi} = \arg\min_{\pi \in \Pi} \sum_{i \leq j} w_{\mathcal{C}(i)\mathcal{C}(j)} |1_{\{(i,j) \in E_1\}} - 1_{\{(\pi(i),\pi(j)) \in E_2\}}|$$

where

$$w_{ij} = \log(\frac{1 - p_{ij}s(2-s)}{p_{ij}(1-s)^2})$$

The symbol used here:

- $\mathcal{C}(i)$: the community of the i-th node.
- $\Pi$: all possible permutation matrices.
- $\pi(i)$: the image of the i-th node in subgraph $G_2$.
- $\theta$: $\{\{p_{ij}\}, s\}$, all introduced parameters

And Fu [6] has given its convexity approximation as

$$\hat{\pi} = \arg\min_{\pi \in \Pi} (\|\pi\tilde{\mathbf{A}} - \tilde{\mathbf{B}}\pi\|_F^2 + \mu\|\pi\mathbf{m} - \mathbf{m}\|_F^2)$$

where:

- $\tilde{\mathbf{A}}$: the weighted adjacency matrix of $G_1$.
- $\tilde{\mathbf{B}}$: the weighted adjacency matrix of $G_2$.
- $\mathbf{m}$: the community assignment vector.
- $\|\cdot\|_F$: the Frobenius norm.

## 6.2 Unsupervised learning formulation

Fu [6] has given various methods to optimize this objective function. In this project, a new method is proposed to as a new attempt. The idea is to use the objective function as the loss function of the network (instead of cross entropies which requires known labels). With back propagation, the weights of the whole network will be adjusted to minimize the loss function, which force the network to give the correct match. Also, since community structures are not concerned in this report, the regularization term($\mu\|\pi\mathbf{m}-\mathbf{m}\|_F^2$) can be dropped.

## back propagation equation

Define the loss function:

$$\mathcal{L} = \frac{1}{2}\|\pi\tilde{\mathbf{A}} - \tilde{\mathbf{B}}\pi\|_F^2$$

The back propagation can be calculated in this way:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \pi_{kl}} =& \frac{1}{2}\frac{\partial}{\partial \pi_{kl}}\|\pi\tilde{\mathbf{A}} - \tilde{\mathbf{B}}\pi\|_F^2 \\
=& \frac{1}{2}\frac{\partial}{\partial \pi_{kl}}\sum_i\sum_j (w_{ij}(\pi\mathbf{A} - \mathbf{B}\pi)_{ij})^2 \\
=& \frac{1}{2}\sum_i\sum_j \frac{\partial}{\partial \pi_{kl}}(w_{ij}(\pi\mathbf{A} - \mathbf{B}\pi)_{ij})^2 \\
=& \sum_i\sum_j w_{ij}(\pi\mathbf{A} - \mathbf{B}\pi)_{ij} \cdot w_{ij}\frac{\partial}{\partial \pi_{kl}}(\pi\mathbf{A} - \mathbf{B}\pi)_{ij} \\
=& \sum_i\sum_j w_{ij}^2(\pi\mathbf{A} - \mathbf{B}\pi)_{ij}(\frac{\partial(\pi\mathbf{A})_{ij}}{\partial \pi_{kl}} - \frac{\partial(\mathbf{B}\pi)_{ij}}{\partial \pi_{kl}}) \\
=& \sum_j w_{kj}^2(\pi\mathbf{A} - \mathbf{B}\pi)_{kj}\mathbf{A}_{lj} - \sum_i w_{il}^2(\pi\mathbf{A} - \mathbf{B}\pi)_{il}\mathbf{B}_{ik} \\
=& \sum_j (\pi\tilde{\mathbf{A}} - \tilde{\mathbf{B}}\pi)_{kj}\tilde{\mathbf{A}}_{lj} - \sum_i (\pi\tilde{\mathbf{A}} - \tilde{\mathbf{B}}\pi)_{il}\tilde{\mathbf{B}}_{ik}
\end{aligned}$$

With this expression, the loss can be propagated backwards onto the weights of the network.

## 6.3 Training

The neural network and training dataset is constructed in the same way as balanced supervised learning (section 5.3) does. The difference is that the node feature vectors are input as a whole.

$$\mathscr{N} : \mathcal{F} \times \mathcal{F} \to [0,1]^{n \times n}$$

The network will output a n-by-n matrix $\pi^* = \mathscr{N}(\mathcal{F})$ with values between 0 and 1, where $n$ is the number of nodes. $\pi^*$ will be used as the approximation of the permutation matrix $\pi$ (since a permutation matrix is required to be orthogonal but $\pi^*$ is not) and calculate the loss.

The exact matching can be calculated this way: the 0-1 matching can be generated from the network output sequentially by marking the highest probability in the output matrix as matched (setting the value to 1, and setting all values in the same column and row to 0), until we get a permutation matrix.

## 6.4 Evaluation

Synthetic random graph with 30, 60 and 100 nodes are generated as test data, whose ROC curve is depicted in **Fig.**3.

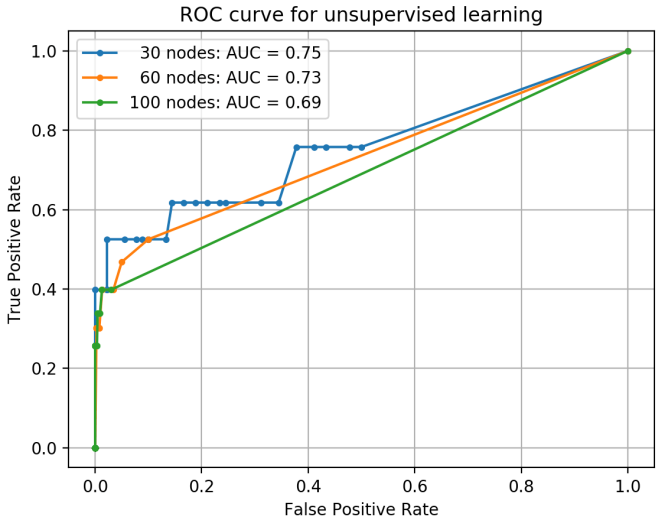


Figure 3: ROC curve for unsupervised learning

**analysis**

From the ROC curve and experiment, it can referred that:

- The performance of unsupervised learning is much worse then the supervised setting.
- As the scale of the social networks grows, the performance of unsupervised learning recedes rapidly. One possible reason can be the too much local optima the objective function contains, which goes beyond the ability of neural networks.

# 7 Summary

In this report, we proposed two neural network based methods to address the social network deanonymization problem under supervised and unsupervised settings.

In supervised setting, a network is trained to learn the match between nodes. Its two variation balanced and extended supervised learning prove to be effective for node matching, with a AUC of 0.84 and 0.79 respectively.

In unsupervised setting, a network is trained to lower the loss function which will finally lead to a correct matching. This proves to be less effective than supervised learning, and its performance recedes rapidly as the size of the network grows.

# References


[1] A. Narayanan and V. Shmatikov, "De-anonymizing social networks," in *Security and Privacy, 2009 IEEE Symposium on*, 2009, pp. 173–187.

[2] K. Sharad, "Learning to de-anonymize social networks," 2016.

[3] S. Nilizadeh, A. Kapadia, and Y. Y. Ahn, "Community-enhanced de-anonymization of online social networks," pp. 537–548, 2014.

[4] S. Liu, S. Wang, and F. Zhu, "Structured learning from heterogeneous behavior for social identity linkage," *IEEE Transactions on Knowledge & Data Engineering*, vol. 27, no. 7, pp. 2005–2019, 2015.

[5] X. Mu, F. Zhu, J. Wang, J. Wang, J. Wang, and Z. H. Zhou, "User identity linkage by latent user space modelling," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1775–1784.

[6] L. Fu, X. Wu, Z. Hu, X. Fu, and X. Wang, "De-anonymizing social networks with overlapping community structure," 2017.

[7] E. Onaran, S. Garg, and E. Erkip, "Optimal de-anonymization in random graphs with community structure," pp. 1–2, 2016.