Adversary for Social Good: Protecting Familial Privacy through Joint Adversarial Attacks*

Chetan Kumar, Riazat Ryan, Ming Shao
Department of Computer & Information Science
University of Massachusetts Dartmouth, Dartmouth, MA, USA
{ckumar, rryan2, mshao}@umassd.edu

Abstract

Social media has been widely used among billions of people with dramatical participation of new users every day. Among them, social networks maintain the basic social characters and host huge amount of personal data. While protecting user sensitive data is obvious and demanding, information leakage due to adversarial attacks is somehow unavoidable, yet hard to detect. For example, implicit social relation such as family information may be simply exposed by network structure and hosted face images through off-the-shelf graph neural networks (GNN), which will be empirically proved in this paper. To address this issue, in this paper, we propose a novel adversarial attack algorithm for social good. First, we start from conventional visual family understanding problem, and demonstrate that familial information can easily be exposed to attackers by connecting sneak shots to social networks. Second, to protect family privacy on social networks, we propose a novel adversarial attack algorithm that produces both adversarial features and graph under a given budget. Specifically, both features on the node and edges between nodes will be perturbed gradually such that the probe images and its family information can not be identified correctly through conventional GNN. Extensive experiments on a popular visual social dataset have demonstrated that our defense strategy can significantly mitigate the impacts of family information leakage.

Introduction

We are living in the era of modern technology and now our lives are surrounded by social media networks. These networks allow us to connect and share our everyday activities with friends, family and others, but notably some of them are user sensitive data. It is our attention to protect those “explicit” sensitive data like credentials by all means, nonetheless other information gets leaked through an “implicit” manner. And this social information is critical to certain group of people. For example, it can be imagined how furious the celebrity will be when their family members photos (especially children) are exposed without their permission. On the other hand, with the rapid development in AI, and especially the new wave caused by deep learning and newly developed tools ready for various social data, we usually have limited knowledge about what additional information will be inferred from the social networks. For example, people’s browsing and reading interests (Google Chrome), shopping habits (Amazon), and movie preferences (Netflix) have been analyzed by advanced algorithms, and it is our feeling that these algorithms know us better than ourselves. In fact, all these services and convenience are supposed to serve for social good under the permission of the users, but sometimes unintentional leakage could occur. This becomes more likely for the user groups of limited knowledge or time for inspection, and puts their privacy at risk.

*This work is supported in part by the UMass Dartmouth College of Engineering faculty start-up fund and NVIDIA Corporation GPU donation.

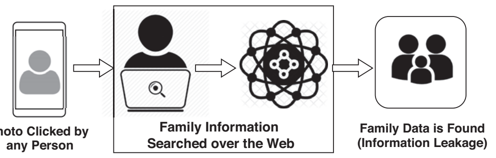


Photo Clicked by any Person
Family Information Searched over the Web
Family Data is Found (Information Leakage)

Figure 1: A sneak shot by someone can be used to find out the family information of the target through social network.

Here we conceptually demonstrate with an unintentional information leakage scenario on the social network. The photo of a targeted person is first captured by the attacker without permission. Then by connecting this photo to existing social networks (e.g., Facebook), the attacker may not only find the identity of the target, but also his/her private family information through dedicated algorithms, e.g., graph neural networks (GNN) (Zhang, Cui, and Zhu 2018), which will be experimentally demonstrated later. This idea is visualized in Figure 1. Generally, people have no willing to disclose personal data in such a manner, but it has already been out of our control, as long as people remain connected by the society and the Internet.

In fact, such risk is NOT rare on the Internet, and could be quickly transmitted and propagated through AI algorithms in an automatic manner. To parse the visual information of images on social media, off-the-shelf tools rooted in convolutional neural network (CNN) has proved their values on a variety of visual recognition tasks (Krizhevsky, Sutskever, and Hinton 2012; He et al. 2016). Most importantly, recent research claims that family and kin relation can be identified through facial images and social context in the photo (Guo, Dibeklioglu, and Van der Maaten 2014; Shao, Xia, and Fu 2014). On the other hand, graph neural networks have been widely used for networked data for better representation and downstream tasks such as node classification, link prediction, community discovery and anomaly detection. It does not only model the network topology, but also the attributes themselves on the nodes, which may work effectively with other multi-modal deep features.

The above family information leakage is rather *a social network problem*, as both visual features of face and their social context in and beyond the photo are critical. Recent attempts mainly focus on the social context in the photos (Guo, Dibeklioglu, and Van der Maaten 2014; Shao, Xia, and Fu 2014). In fact, images are not alone but usually attached and hosted by social networks, and connections among them can be made up by identity, family connections, friendship, or working relation. If we follow a similar way but only target on family information based on visual features as well as graph topology, it turns out to a valuable semi-supervised learning problem on the graph, which, to the best of our knowledge, has not yet been explored. In this research work, we hypothesize that this will cause an even severe information leakage, and demonstrate by the experiments in later sections. Thus, our research goal is to effectively protect social privacy in this regard through a new adversary learning model.

Adversarial samples (Goodfellow, Shlens, and Szegedy 2014) have attracted substantial attention in the field of security, which however, originated from a few intriguing observations on deep visual features. It has shown that by adding tiny noises, the pre-trained deep learning models can be easily fooled. To that end, more and more works have been proposed for better "attacks" and "defense" strategies. Notably, most of existing adversarial sample studies focus on visual or high-dimensional data, a necessary condition for attacks in this regimen. It comes to our attention recently that graph may also be subject to adversarial attacks by flipping the edges therein. This naturally leads to the following question: "can we use both adversarial samples and graph for social good?" To achieve this goal, we first demonstrate by the facts of family information leakage on social networks through graph neural network modeling. Second, we develop a novel joint adversarial attack modeling based on both node features and graph to effectively perturb the social network under the given budget and thus avoid the leakage of family information. In brief, the contributions of this paper can be summarized as:

- Demonstrate the family information is at risk on social network through plain graph neural networks.
- Propose a joint adversarial attack modeling on both features and graph structure for family privacy protection.
- Quantitatively and qualitatively shows the effectiveness of our framework on networked visual family datasets.

Related Work

Visual Family Understanding

Early works on visual family understanding focus on two well-defined face image based problems: (1) kinship verification, (2) family recognition. Kinship verification refers to identifying kin relation between two given samples, based on the evidence from psychology and cognitive sciences (Daly and Wilson 1982; Alvergne, Faurie, and Raymond 2007) that human faces convey similar characteristics among family members. Such approaches may be applied to family album organization, social media mining and can be used in issues such as human trafficking and finding missing children, in refugee crisis, etc. (Yan and Hu 2018; Robinson et al. 2018). Family recognition on the other hand attempts to identify the family from a pool of family data for the probe sample, which becomes even challenging with the increasing number of families in the wild. To validate this research, a few visual family and kinship datasets have been released recently, including Families in the Wild (FIW) (Robinson et al. 2018), KinFaceW (Lu et al. 2013), Family-101 (Fang et al. 2013), TSKinFace (Qin, Tan, and Chen 2015), etc. In this paper, we focus on family recognition but from a social network perspective. Different from existing works, both visual information and graph topology will be considered, and we cast the original family recognition problem to a semi-supervised learning on the graph.

Graph and Network Embedding

As a fundamental work for networked data, embedding (or network data dimensionality reduction) has been widely explored for decades. Early work focuses on graph embedding for better feature extraction for data distributed on manifold including (Belkin and Niyogi 2002; Roweis and Saul 2000; Yan et al. 2007). In the era of social networks, networked data become another research focus, and various network embeddings have been proposed including (Perozzi, Al-Rfou, and Skiena 2014; Grover and Leskovec 2016). It is not until recently that deep learning philosophy has been successfully integrated with network embedding, i.e., Graph Neural Networks (GNN) (Zhang, Cui, and Zhu 2018) and promotes a series of methods including Graph Convolutional Neural Network (GCN) (Kipf and Welling 2016), ChebyNet (Defferrard, Bresson, and Vandergheynst 2016), and GraphSAGE (Hamilton, Ying, and Leskovec 2017). While these approaches are originally proposed for networked data such as citation network, protein network, etc., they could also be the foundation of this research which may be directly applied to our social network based family recognition.

In this research, we will explore two aspects of GNN. Namely, (1) how to apply GNN to challenging family recognition problem to simulate the scenario of information leakage; (2) adversarial graph to protect the family privacy. To our knowledge, both of them are not explored in the previous research.

Adversarial Samples and Graph

Adversarial sample was first introduced by by Szegedy et al. in (Szegedy et al. 2013) and further interpreted in (Goodfel-

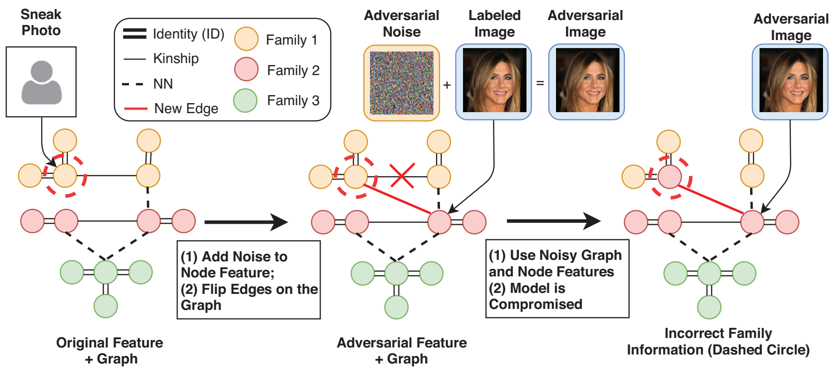


Figure 2: Framework Overview.

low, Shlens, and Szegedy 2014). The following works have significantly promoted this thought and spawn considerable intriguing "attack" and "defense" strategies. See two recent survey works (Akhtar and Mian 2018; Yuan et al. 2019) for more details. Leading works along this line usually attempt to perturbed features of the target by deviating them from the original gradient directions including FGSM (Goodfellow, Shlens, and Szegedy 2014), I-FGSM (Xie et al. 2018), MI-FGSM (Dong et al. 2018), Papernot's method (Papernot et al. 2016) and C&W method (Carlini and Wagner 2017; Ozdag 2018). The generated adversarial examples usually target at high-dimensional data. The added small perturbations to the original images will then misguide the classifier. Attacks are also designed based on human visual sensitivity (Shen et al. 2019). Different from existing works and their motivations, we carry this idea in another direction where we take advantage of adversarial samples to safeguard the privacy of family on social networks.

A recent trend is adversarial graph that accounts for attacks launched towards graph topology (Sun et al. 2018), which has not been well discussed in adversarial samples regime. Under the network environment, attackers are allowed to inject false data such as creating fake followers on social networks. In brief, approaches in this line can be divided into two groups, namely, (1) graph poisoning (Bojchevski and Günnemann 2019); (2) graph evasion (Zügner and Günnemann 2019). While the first one targets at the original graph such that the perturbed graph may affect the model training, the second one gets involved during the testing time. In this work, we take advantage of the recent graph poisoning work (Bojchevski and Günnemann 2019) to perturb the graph and flip the edges. In this way, we are more likely to discover the key connections that will potentially avoid the family information leakage. The contaminated graph will then be fed to the learning model (GNN) during the running time. This will work together with the adversarial samples to better protect the privacy.

Social Kinship Modeling

Social Family Recognition (SFR)

In this section, we will demonstrate that family recognition can be well addressed under the network environment by casting it to a semi-supervised learning problem on the social networks. Therefore, we will be able to prove our hypothesis that family understanding is rather a social network problem, which we formally name as social family recognition (SFR). Conventional visual family recognition (VFR) is to train a multi-class classifier first, and then assign family labels to each probe image in the running time. Therefore, the familial visual features are the key to success. Even with the most recent deep features designed for visual kinship, e.g., SphereNet (Liu et al. 2017), the accuracy is far from acceptable. Next, we simulate the social network environment and construct a graph to host labeled family images, unlabeled images and sneak shot by attackers.

Attack simulation on graph The overall framework illustration can be found in Figure 2. Assume on existing social networks, there is accessible public information for each *subject* such as name (identity), family information (family tag), as well as large amount of unlabeled data that attackers can only view their profile photos. There is also pairwise information encoding the relations between two subjects such as *same identity* and *have kin relation* on the networks. In addition, based on the visual similarity of two faces, attackers can link the sneak shot to the most similar faces via off-the-shelf deep features. Afterwards, the sneak shot is latched to the networks and attacks would be launched. This is illustrated in the first step of Figure 2.

Family Recognition on the Graph

First, let us define $\mathcal{G} = (V, E)$ be an attributed and undirected graph where V denotes the set of nodes and E denotes the set of edges. Further let $A \in \{0,1\}^{N\times N}$ be an adjacency matrix and $X \in \mathbb{R}^{N\times D}$ represents the node features where N is the number of samples, and D is the dimension of the input feature. Next, we formally model the social family recognition as a semi-supervised learning problem on the graph. Assume the labeled and unlabeled image feature matrices are $X_L \in \mathbb{R}^{D\times N_L}$ and $X_U \in \mathbb{R}^{D\times N_U}$, respectively, and the corresponding label vector is $y_L \in \mathbb{R}^{N_L}$. Our goal (i.e., attacker's aim) is to find out the mapping $f_{\mathcal{G}} : ([X_L, X_U]) \mapsto ([y_L, y_U])$ as well as the label vector $y_U \in \mathbb{R}^{N_U}$ for the unlabeled samples.

Second, we will elaborate the graph construction process.

In our graph, each node represents visual features generated by the state-of-the-art kinship descriptors, while edges encode the relation between two nodes. Without loss of generality and as explained before, we consider three different edges: (1) identity (ID); (2) kinship (KIN); (3) k nearest neighbor (KNN). ID edges link nodes of the same identity, and similarly KIN edges link two nodes of the same family label. To avoid isolated nodes, we apply KNN graphs to our problem. By varying the number of images with family labels or of same identity, and the number of neighbors k, we are allowed to generate different simulated social networks.

Next, we will take advantage of the recent GNN model to solve the proposed SFR on graph (Kipf and Welling 2016). Basically, we will take both labeled nodes X_L, unlabeled nodes X_U, and graph $\mathcal{G}$ in the training stage. Once the GNN parameters $\mathcal{W}_{\mathcal{G}}$ are learned, labels of unlabeled nodes on the graph will be assigned. The feed-forward process in GNN is defined as: $H^l = g_l(H^{l-1}, A)$ where g is a non-linear mapping function, H is the hidden layer representation, and l indexes the layer. Essentially, node features in each layer are aggregated via graph and then passed to next layer, namely,

$$H^l = \sigma\big(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l-1)}W^{(l-1)}\big), \tag{1}$$

where $\tilde{A} = A + I_N$ is the adjacency matrix with self-loops by adding identity matrix I_N. $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$, $W^{(l)}$ is the l-th layer trainable weight matrix, and $\sigma(.)$ represents activation function (usually ReLU). In our work, node features $H^{(0)}$ are extracted by SphereNet pre-trained on FIW dataset. Then these features are fed to a two-layer GNN. The output of our deep learning model can be formulated as:

$$Z = \text{softmax}\big(\hat{A}\text{ReLU}\big(\hat{A}XW^{(0)}\big)W^{(1)}\big), \tag{2}$$

where Z is the network output, and $\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$. Softmax function here normalizes the output feature in a row-wise manner. By using the entropy loss summed over labeled dataset, the model parameter $\mathcal{W}_{\mathcal{G}}$ can be trained end-to-end. The relevant results can be found in the first subsection of Experiment Results.

Adversary for Familial Privacy Protection

It can be seen that family information is at risk on the social network via the proposed GNN model. Next, we will briefly describe how to use adversarial samples methodology to protect family privacy. Afterwards, we will discuss two separated ways of protecting family privacy followed by a joint framework that embraces the merits of both.

Privacy protection and defense philosophy Since public information on social networks is available to everyone, we are allowed to add imperceivable noise to labeled images (i.e., family labels) in the same way as adversarial samples. This is doable for social websites while attackers are blind to this. In the meanwhile, social websites are able to hide certain links based on public information with respect to identity and family information. When adding adversarial noise to data, the pairwise similarity changes, which will affect the connections based on this, e.g., k nearest neighbor links. All of these allow us to construct the following adversarial attacks for privacy protection and social good, as shown in Figure 2, the second and third steps.

Visual Familial Feature Perturbation

Facial images are typically used in visual family recognition (VFR), and thus we will explore ways of compromising visual features, namely, SphereNet features. Note that other CNN models can also be applied here by following white box attack. Representations from fully connected layers are used as input which can be compromised through off-the-shelf tools including Fast Gradient Sign Method (FGSM) (Goodfellow, Shlens, and Szegedy 2014). Other recent adversarial sample models in the same vein can also be applied here, but explorations in that vein is beyond the scope of this paper. Note that only data with family labels are perturbed.

By borrowing the gradient information from SphereNet (also applicable to other CNN models), we are allowed to generate adversarial samples of the node features. Essentially, FGSM generates perturbation by updating features in the direction of the sign of the gradient at each pixel, which can be formally defined as:

$$x' \leftarrow x + \eta\,, \eta = \epsilon sign(\nabla_x J(\mathcal{W}_{\text{SphNet}}, x, y)), \tag{3}$$

where $\mathcal{W}_F$ is the model parameters of pre-trained SphereNet, ∇_x is the gradient of the loss function $J(\cdot)$ with respect to input x with true label y. ϵ controls the magnitude of the perturbation to be applied on the input. Once the SphereNet is learned, all these parameters are known and easy to calculate. The perturbed sample x' can be generated by simply adding the noise to the clean data.

A key issue in this attack is maintaining a small value for ϵ such that this perturbation is not noticeable for both regular users, and attackers. In other words, a heavily perturbed sample will not be recognized, but this change can be perceived by everyone. In our adversarial sample modeling, we always start from small ϵ, and may stop increasing its value if perturbation is observable. Another key difference with the conventional adversarial sample methods is we are perturbing node features before GNN training, similar to the graph poisoning philosophy. Once the model is trained, sneak shots from attackers will not be recognized correctly. While in regular adversarial attack, the data will not be perturbed until the running time.

Social Graph Perturbation

As our research targets are most likely hosted by social websites, and our preliminary research has shown that family recognition can be hacked easily on graph, we are well motivated to perturb the topology of the social networks, in addition to the node features for family privacy protection. Furthermore, we are using adversarial graphs for "defense" and require the graph perturbation to be blind to attackers. Thus, graph poisoning approach is more appropriate in our case. Namely, the graph will be perturbed before the model learning to maximize the loss of attacker's model. Inspired by recent graph perturbation work, we optimize the adversarial graph under the network embedding framework.

Assume the affinity graph after perturbation is A' and the embedding of the graph to be optimized is $V \in \mathbb{R}^{N\times d}$, then

A' can be optimized through the following model:

$$\max_{A'} \mathcal{L}_{\mathcal{E}}(V^*, A'), \text{s.t., } \|A - A'\|_0 = e, A' = A'^T, \tag{4}$$

where e is the budget on edge flips, $\|\cdot\|_0$ indicates matrix ℓ_0 norm, and $\mathcal{L}_{\mathcal{E}}$ denotes a generic loss function for Graph Embedding. It should be noted that the embedding V^* is not jointly optimized in Eq. (4) in an explicit manner, and we may consider its solution being part of the solution to A', if the embedding is modeled through DeepWalk algorithm (Perozzi, Al-Rfou, and Skiena 2014). In fact, the loss of DeepWalk algorithm that optimizes Z^* and A' that contributes to edge flip are both determined by the same spectrum, while the latter is slightly perturbed by a single flip on the graph. Therefore, by targeting at a fixed dimension r (i.e., rank-r SVD approximation) for V and maximizing the loss caused by a single flip in A', we are allowed to find the adversarial flips on the graph in an greedy manner. Empirically, such strategy is not only working well on DeepWalk, but also for GNN used in our SFR problem. To that end, an incremental evaluation has been shown in Figure 3(a), 3(b) and Figure 5 to demonstrate that with more edges flipped, attackers have less chance (lower accuracy) to obtain the familial privacy.

Joint Feature and Graph Adversarial Samples

It can be seen that both feature and graph perturbations contribute to our research goal, and their integration may improve the effectiveness while keeping the budget of each of them low. We may formulate this joint adversarial attack model as the following:

$$\begin{aligned} \max_{\{X', A'\}} \mathcal{L}_{\mathcal{AD}}(X', A') &\triangleq \max_{\{X', A'\}} \ln Z^*_{\text{pert}} - \ln Z^*_{\text{clean}}, \\ \text{s.t., } \lambda\|A - A'\|_0 &+ (1-\lambda)\|X - X'\|_F \le \theta, \end{aligned} \tag{5}$$

where $\mathcal{L}_{\mathcal{AD}}$ indicates the loss function of the joint attack, θ is the total budget, $\|\cdot\|_F$ indicates the matrix Frobenius norm, and λ is a balancing parameter. Z^*_{pert} is the softmax output of labeled data based on perturbed feature and graph, and Z^*_{clean} is that based on clean data and graph. Compared with Eq. (4), there are three changes: (1) feature perturbation and corresponding loss are added; (2) balance between features and graph perturbation are considered, controlled by λ; (3) the loss is determined by the softmax output of GNN. Intuitively, once θ and λ are fixed, we may search over the domain defined by A' and X', and obtain the maximum loss of $\mathcal{L}_{\mathcal{AD}}$. However, this bi-level problem has no closed-form solution. In this paper, we propose to approach this problem by decomposing into two sub-problems, and solve one at a time in an iterative manner:

$$\begin{cases} X'_{(t+1)} = \arg\max\limits_{X'_{(t)}} \mathcal{L}_{\mathcal{G}}(X'_{(t)}; A_{(t)}, \mathcal{W}_{\mathcal{G}}, \epsilon) \\ A'_{(t+1)} = \arg\max\limits_{A'_{(t)}} \mathcal{L}_{\mathcal{G}}(A'_{(t)}; X_{(t)}, \mathcal{W}_{\mathcal{G}}, e) \end{cases} \tag{6}$$

The first sub-problem can be solved by FGSM using the gradient of GCN loss with respect to features X' at step t, with affinity A' and $\mathcal{W}_{\mathcal{G}}$ parameters fixed. In fact, we can also perturb the original image through SphereNet, similar to Eq. (3). The benefit are two folds. (1) FGSM works especially well on high-dimension input, namely, the original image; (2) It depends only on image rather than image plus graph under the GCN framework. Further discussions can be found in Figure 6(b) in experiments. The second sub-problem is essentially a graph poisoning problem discussed in (Bojchevski and Günnemann 2019). As edge flipping is non-derivable due to the discrete nature of the problem, we also refer to the approximate solution to problem in Eq. (4) to solve the second sub-problem.

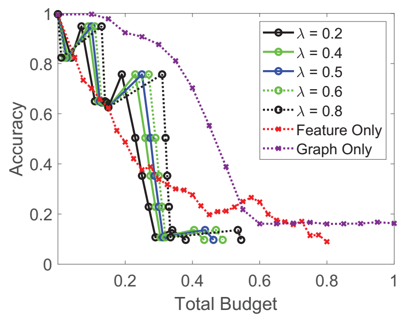
```
Input: $X, A, e, \epsilon, \lambda, \theta$.
Output: $\mathcal{W}_{GCN}$ from the last training, $X'_{(t)}, A'_{(t)}$.
Set $A'_{(0)} \leftarrow A$, $X'_{(0)} \leftarrow X$, $t = 0$, $\epsilon = \Delta\epsilon = 0.00025$,
  $e = \Delta e = 0.05|E|$ ;
while $\lambda \times \frac{e}{|E|} + (1-\lambda) \times 100\epsilon \le \theta$ do
    Compute $X'_{(t+1)}$ with budget $\epsilon$ based on Eq. (3) ;
    Compute $A'_{(t+1)}$ with budget $e$ based on Eq. (4);
    if $\mathcal{L}_{\mathcal{G}}(X'_{(t+1)}, A'_{(t)}) > \mathcal{L}_{\mathcal{G}}(X'_{(t)}, A'_{(t+1)})$ then
        Update $\mathcal{W}_G$ by $X'_{(t+1)}$ and $A'_{(t)}$;
        $A'_{(t+1)} = A'_{(t)}$ ;
        $\epsilon \leftarrow \epsilon + \Delta\epsilon$;
    end
    else
        Update $\mathcal{W}_{\mathcal{G}}$ by $X'_{(t)}$ and $A'_{(t+1)}$ ;
        $X'_{(t+1)} = X'_{(t)}$ ;
        $e \leftarrow e + \Delta e$;
    end
    $t \leftarrow t + 1$;
end
```

Algorithm 1: Procedure of Joint Adversarial Attack.

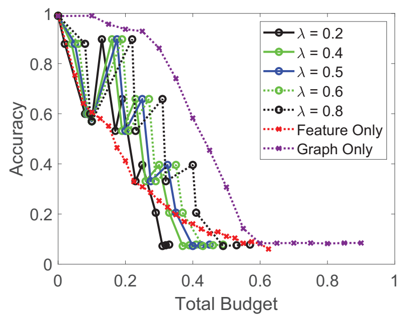
So far, we have been discussing loss minimization while ignoring the budgets allowed to perturb the affinity graph and feature matrix, namely, the total budget θ in problem Eq. (5). In fact, in our iterative solution, the incremental budget changes are not easy to quantify, and how to budget between X' and A' is unclear. To make the entire problem tractable, we pursue a local solution through greedy search. Each time, for a pre-defined incremental budget: $\Delta\epsilon$ for feature and Δe for edges, we compare the loss $\mathcal{L}_G$ caused by perturbed features and perturbed graph, respectively. If perturbed features lead to a larger loss, then we will budget feature perturbation in this iteration; otherwise, we will budget graph perturbation. The process will repeat until the total budget θ is reached, and we are allowed to further balance budgets between features and graph through λ. The details of joint attack algorithm can be found in Algorithm 1.

Experiments

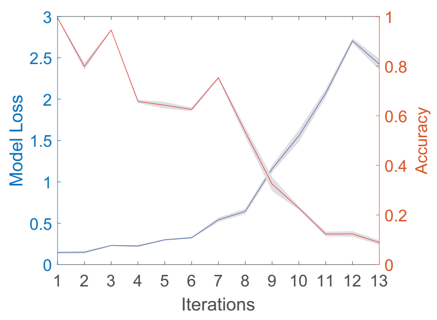
Database and settings In this study we have used Families In the Wild (FIW) dataset (Robinson et al. 2018). FIW is the largest visual kinship dataset available which comprises of 11 types of relationships which range from same generation, e.g., Brother to Brother (B-B) to first, e.g., Fa-



(a) Budget Vs. Acc on Family-100.



(b) Budget Vs. Acc on Family-300.



(c) Loss and accuracy on Family-100.

Figure 3: Budget, accuracy and model loss in different iterations.

ther to Daughter (F-D), and second generation, e.g., Grand Mother to Grand Daughter (GM-GD). It contains 11,932 family photos of 1000 families where on average 12 images are from each family and 656,954 image pairs are divided between 11 relationships. Pairs are labeled with *true* kin relation or *false* kin relation.

Pre-processing and configuration First, we have pre-processed the FIW dataset by extracting the features of the images by using pre-trained SphereNet model, and the dimension of node features is thus reduced to 512. Second, we construct the social graph to fulfill social family recognition by considering three factors: (1) number of labeled family members; (2) number of images per person; (3) number of neighbors in KNN graph. Their meaning and illustration also can be found in Figure 2. More details and impacts of these parameters will be discussed in the following sections.

We have created two social networks: small one with 100 families termed 'Family-100' and a large one with 300 families termed "Family-300". Family-100 dataset contains 502 subjects and 2758 facial images where on average each family has 5 kin members and each member has 5 face images. Among 2758 nodes, we have used 502 nodes for training with graph, while the rest for validation and testing. Family-300 dataset contains 1712 members and 10255 face images. We have used 1712 nodes for training with graph, while rest of them are used for validation and testing.

In all experiments, we use PyTorch library, SphereNet and GCN open implementation by (Kipf and Welling 2016; Liu et al. 2017). All the codes are implemented on Ubuntu16.04 system with i7-8700 (3.2 GHz), 16 GB memory and a nvidia GTX 1070 GPU card.

Experimental Results

Family recognition on the graph Family recognition has been discussed in (Robinson et al. 2018), and here we compare the results based on: (1) visual feature; (2) visual feature + graph. Under this setting, we achieve **17.00%** accuracy via visual feature only while **98.89%** accuracy via both visual features and graph on Family-300 dataset. This is a huge improvement, thanks to the graph that models the links between different nodes and GNN models.

In addition, we explore the impacts of different number of neighbors k in KNN graph (KNN), number of images with

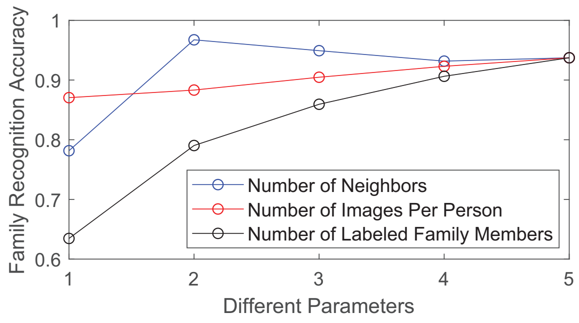


Figure 4: Illustration of impacts of graph parameters on Family-100. Note when evaluating one parameter, we fixed the rest by setting them = 5. For example, when discussing neighbor of neighbors, we vary it from 1 to 5, while fix number of image per person and labeled family members to 5.

Table 1: Impacts of different edge types on Family-100.

Edge Type			Edge Flipped	Accuracy(%)
KNN	ID	KIN		
✗	✗	✗	0	99.68
✓	✗	✗	10000	98.57
✗	✓	✗	10000	98.49
✗	✗	✓	10000	95.23
✓	✓	✓	10000	93.73

family label (KIN), number of images from the same identity (ID). For labeled images within one family, or images of the same person, they will be fully connected, which simulates the social network connections in real-world. It can be seen from Figure 4 that $k = 2$ is good choice when building the social networks to facilitate this problem. An increasing number of either images per person or labeled family members will benefit the accuracy.

Joint feature and graph adversarial samples In this section, we will demonstrate how joint adversarial attack model protects the family privacy. First, we will present the results from separate attacks: (1) feature perturbation and (2) graph perturbation, as can be seen in Figure 3(a) and 3(b), represented by red and purple dot lines, respectively. Note to

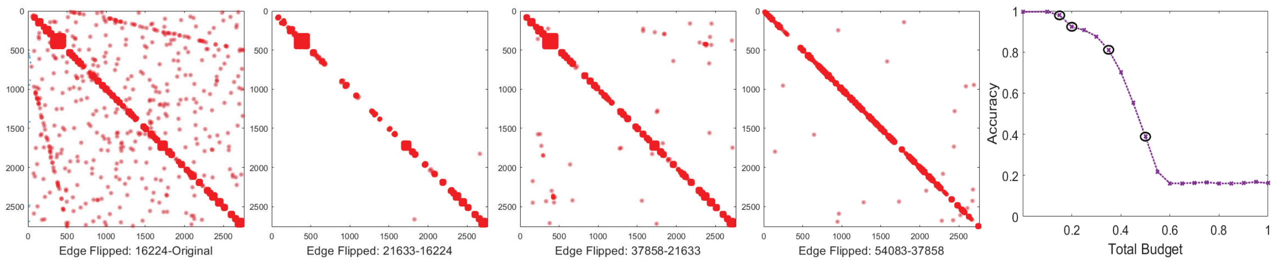


Figure 5: Illustration of changes on the graph when flipping different numbers of edges on Family-100. “16224-Original” means the flipped edges given the edge budget of 16224, and “21633-16224” means additional flipped edges given a larger budget 21633 compared to the previous budget 16224.

make a fair comparison between feature and edge perturbation budgets, we create *basic measurement* for each of them. For feature perturbation, we use $\epsilon \times 100$ as the basic measurement; for graph perturbation, we use percentage of the flipped edges as the basic measurement. To compute the total budget from both, we introduced a balancing parameter λ. Therefore, the total budget will be:

$$\text{Total-Budget} = \lambda * \text{Edge-Flipping-Ratio} + (1-\lambda) * 100 * \epsilon,$$

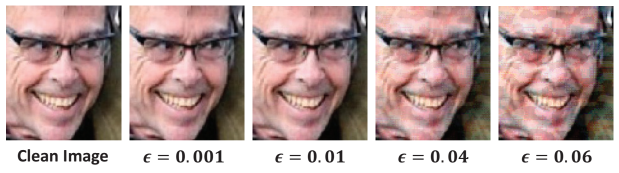
which is usually in the rage of [0,1]. As expected, with larger ϵ and number of flipped edges, the family recognition accuracy continues decreasing, and in general, feature perturbation works better in this case. The downside of each is excessive use of one particular attack, which compromises the attack effectiveness.

Second, we conduct the joint attack whose results can be found in Figure 3(a) and 3(b), represented by black, green and blue curves. We show five results with different λ values. It can be learned that most of joint attacks work better than single attack, and on the two datasets, $\lambda = 0.2$ works best. It also shows that our joint attack works well in a large range with respect to budget weight λ.

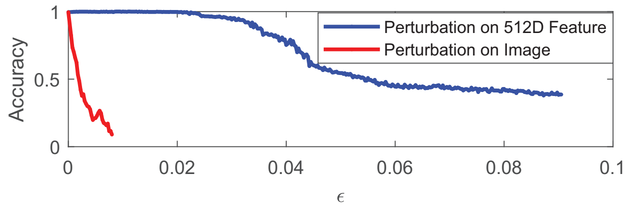
Third, as our solution is based on greedy search leading to local solution, it is necessary to check the convergence of the model. In our case, we check the model training loss and family recognition accuracy after each re-training. In Figure 3(c), 13 iterations from 5 trials are shown. As expected (in terms of adversarial attack), the loss becomes larger while the accuracy becomes lower with more iterations.

Discussions

First, we test the impacts of different edge types and results can be found in Table 1. It can be seen that flipping edges of different types may have different impacts, and KIN edge is very effective among three types. In addition, using a mixture of three based on their contributions (graph perturbation) shows the best performance. We also demonstrate the roles of different edges through the graph changes given different edge budgets in Figure 5. We consider four budgets: 16224, 21633, 37858, 54083, and visualize the flipped edges between two given budgets on the first four figures, and the accuracy change on the fifth figure, where each circle presents an edge budget. Therefore, we may know what edges are flipped when given larger edge budgets. It can be



(a) ϵ impacts on appearance.



(b) Image vs. 512D SphereNet feature.

Figure 6: Impacts of ϵ on visual and node features.

seen that from the beginning, a mixture of KIN, ID, and KNN edges are considered, but the effectiveness of the perturbation is not significant as the major structure of the graph does not change. With more edges flipped, there is an acceleration on the perturbation effects, and more KIN and ID edges are flipped, which echoes the results from Table 1.

Last, we also show the impacts of ϵ under different conditions. When perturbing the face images directly which lie in high dimensional space, a smaller ϵ would work well. In Figure 6(b), $\epsilon < 0.02$ (red curve) can achieve very good results, and this perturbation can be barely perceived in Fig 6(a). When using SphereNet feature (512D) as the attack target (blue curve), even larger $\epsilon > 0.8$ can not provide acceptable result, and at this magnitude, there is already significant noise found on the face image. While the noise is on the feature space in this case, it reflects the facts that the feature perturbation is perceivable. In brief, even feature perturbation approaches like FGSM weigh more and lead to better budget on Family-100 and Family-300 datasets, it has limitation when feature dimension is relatively low. Therefore, it might be appropriate to weigh more on graph perturbation.

Conclusions

In this paper, we proposed a novel joint adversarial attack model to prevent family information leakage through social network. First, we demonstrated the family information will be hacked through social networks and plain Graph Neural Networks and sneak shot on the street. Second, a defense mechanism based on the joint adversarial attack has been developed. Specifically, both perturbation on the node features and graph were considered and optimized given a pre-defined budget. Experiments on popular visual kinship dataset have shown our defense strategy was effective when limited changes upon budgets were made towards data.

References

Akhtar, N., and Mian, A. 2018. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access* 6:14410–14430.

Alvergne, A.; Faurie, C.; and Raymond, M. 2007. Differential facial resemblance of young children to their parents: Who do children look like more? *Evolution and Human Behavior* 28(2):135–144.

Belkin, M., and Niyogi, P. 2002. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems*, 585–591.

Bojchevski, A., and Günnemann, S. 2019. Adversarial attacks on node embeddings via graph poisoning. In *International Conference on Machine Learning*, 695–704.

Carlini, N., and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, 39–57. IEEE.

Daly, M., and Wilson, M. I. 1982. Whom are newborn babies said to resemble? *Ethology and Sociobiology* 3(2):69–78.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, 3844–3852.

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting adversarial attacks with momentum. In *IEEE Conference on Computer Vision and Pattern Recognition*, 9185–9193.

Fang, R.; Gallagher, A. C.; Chen, T.; and Loui, A. 2013. Kinship classification by modeling facial feature heredity. In *2013 IEEE International Conference on Image Processing*, 2983–2987. IEEE.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Grover, A., and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 855–864. ACM.

Guo, Y.; Dibeklioglu, H.; and Van der Maaten, L. 2014. Graph-based kinship recognition. In *2014 22nd International Conference on Pattern Recognition*, 4287–4292. IEEE.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 1024–1034.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Kipf, T. N., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 1097–1105.

Liu, W.; Wen, Y.; Yu, Z.; Li, M.; Raj, B.; and Song, L. 2017. Sphereface: Deep hypersphere embedding for face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 212–220.

Lu, J.; Zhou, X.; Tan, Y.-P.; Shang, Y.; and Zhou, J. 2013. Neighborhood repulsed metric learning for kinship verification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36(2):331–345.

Ozdag, M. 2018. Adversarial attacks and defenses against deep neural networks: A survey. *Procedia Computer Science* 140:152–161.

Papernot, N.; McDaniel, P.; Wu, X.; Jha, S.; and Swami, A. 2016. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, 582–597. IEEE.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 701–710. ACM.

Qin, X.; Tan, X.; and Chen, S. 2015. Tri-subject kinship verification: Understanding the core of a family. *IEEE Transactions on Multimedia* 17(10):1855–1867.

Robinson, J. P.; Shao, M.; Wu, Y.; Liu, H.; Gillis, T.; and Fu, Y. 2018. Visual kinship recognition of families in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40(11):2624–2637.

Roweis, S. T., and Saul, L. K. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290(5500):2323–2326.

Shao, M.; Xia, S.; and Fu, Y. 2014. Identity and kinship relations in group pictures. In *Human-Centered Social Media Analytics*. Springer. 175–190.

Shen, Z.; Fan, S.; Wong, Y.; Ng, T.-T.; and Kankanhalli, M. 2019. Human-imperceptible privacy protection against machines. In *27th ACM International Conference on Multimedia*, 1119–1128. ACM.

Sun, L.; Wang, J.; Yu, P. S.; and Li, B. 2018. Adversarial attack and defense on graph data: A survey. *arXiv preprint arXiv:1812.10528*.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Xie, C.; Zhang, Z.; Wang, J.; Zhou, Y.; Ren, Z.; and Yuille, A. 2018. Improving transferability of adversarial examples with input diversity. *arXiv preprint arXiv:1803.06978*.

Yan, H., and Hu, J. 2018. Video-based kinship verification using distance metric learning. *Pattern Recognition* 75:15–24.

Yan, S.; Xu, D.; Zhang, B.; Zhang, H.-J.; Yang, Q.; and Lin, S. 2007. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(1):40–51.

Yuan, X.; He, P.; Zhu, Q.; and Li, X. 2019. Adversarial examples: Attacks and defenses for deep learning. *IEEE Transactions on Neural Networks and Learning Systems*.

Zhang, Z.; Cui, P.; and Zhu, W. 2018. Deep learning on graphs: A survey. *arXiv preprint arXiv:1812.04202*.

Zügner, D., and Günnemann, S. 2019. Adversarial attacks on graph neural networks via meta learning. *arXiv preprint arXiv:1902.08412*.