社交网络去匿名化算法的研究

小组成员
PB20111647 鲍润晖
PB20000103 王炳勋
PB20111704 张宇昂
PB20111651 何泽昊
PB19071508 唐思渝

1. Abstract

许多互联网公司都提供社交网络服务，为了满足数据分析的需要，社交网络公司会向研究人员和广告商提供匿名化的网络信息。但是有攻击者可以通过比对现实社交网络与匿名化网络的相似性，将那些匿名化了的用户对应到真实用户，即社交网络的去匿名化攻击。
我们小组在本学期中阅读并总结了一些关于去匿名化社交网络的论文；构造了一系列数据集；复现了一些去匿名化算法，尝试并比对了它们在数据集上的性能；我们还构造了一系列的匿名化算法，来观察它们对抗去匿名化算法的效果。实验结果证明了去匿名算法的优秀性能，以及我们应该如何选择匿名化算法以对抗去匿名化攻击。

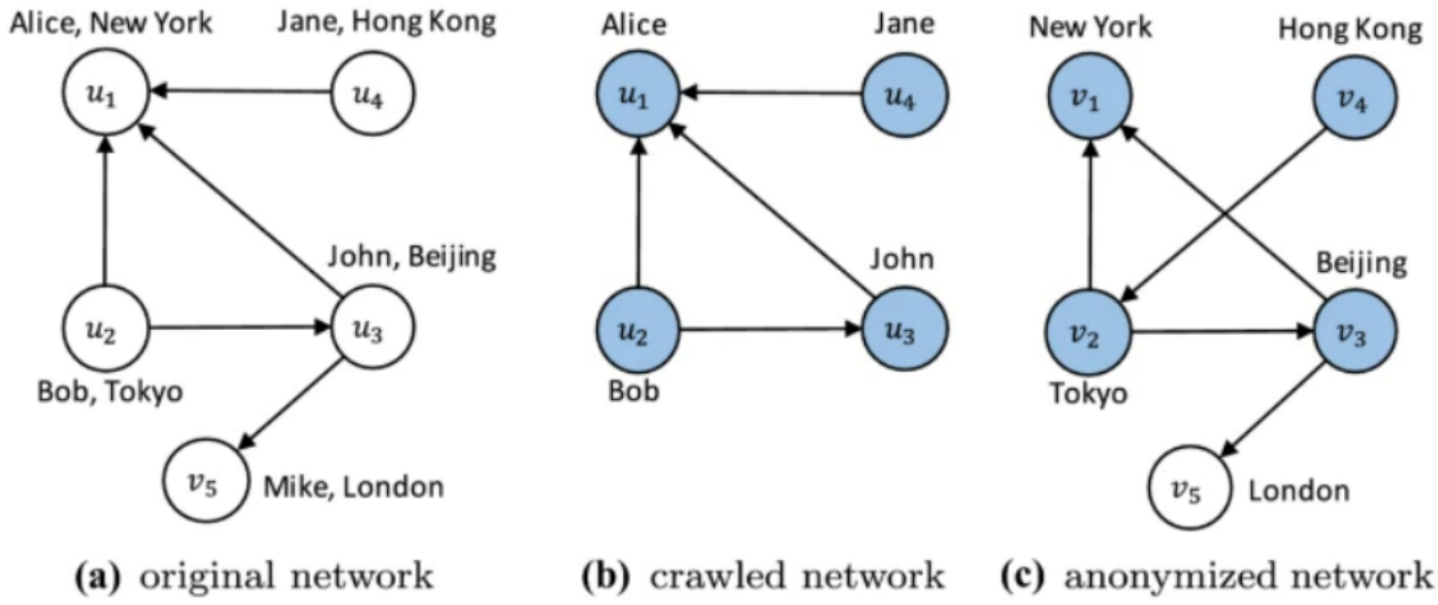
2. Introduction

在线社交网络一直是一个热门的话题，许多互联网公司都提供社交网络服务。社交网络可以建模为一个图：此类网络中用户是表示为具有多个属性的节点，包括姓名、性别、兴趣、位置等信息；用户之间的交互被抽象为用户节点之间的单向或双向边，比如关注、点赞、发送消息等交互。
社交网络也引起了研究人员和广告商的广泛关注。为了满足数据分析的需要，社交网络公司提供共享网络信息的服务。为了保护用户的隐私，社交网络公司在发布网络时会去除用户节点中姓名、位置等敏感信息，同时对网络进行匿名化处理 [1]（4.1节），比如随机地修改网络的结构。但是即便如此，这个匿名化网络仍有可能泄露用户隐私。
一个流行的问题是身份泄露，即攻击者从发布的匿名化网络中找到对应原始社交网络中的真实用户节点，也就是找到匿名节点的身份信息，这类攻击被称作社交网络的去匿名化攻击。去匿名化攻击大多考虑两个网络中节点的相似性，并尝试将相似的节点认为是对应同一节点。按去匿名化攻击前，攻击者是不是事先需要得知一些种子节点的对应关系（称为seed），可以把去匿名化攻击分为需要的seeded方法和不需要的seedless方法。

- seeded方法：是扩散性质的，需要一些已知身份信息的匹配点对，作为种子对，从它们出发不断扩展到其他的结点.由于已经有了扩散的中心，在初始中心正确的情况下,比较容易得到一些高质量的匹配对。但是这类方法的缺陷在于：往往对种子对的可靠性非常依赖,而该可靠性在实际操作中往往并不能保证，因此造成了相当的匹配错误率。
- seedless方法：将各种结点对放在一个相对平等的初始地位上，然后通过对图结构的分析匹配并获得身份结果。这类方法主要利用的是结点的结构特征信息（例如度数、子图、结点相似度、描述性信息等），这些特征信息如果利用得好，可以得到高质量的匹配对。而由于信息多而杂，表示方法多，不方便进行统一度量，度量之后也难以很好地加以利用，这类方法主要的困难在于对特征的提取以及利用。

在本学期的课程实验中，我们小组阅读并总结了一些关于去匿名化社交网络的论文，包括社交网络去匿名化方向的第一篇论文 *"De-anonymizing social networks"* [2]，考虑了“社区”结构的 *"Community-Enhanced De-anonymization of Online Social Networks"* [3]，使用局部敏感向量映射用户特征向量的 *"An efficient reconciliation algorithm for social networks"* [4]，以及seedless的 *"Fast De-anonymization of Social Networks with Structural Information"* [5]

为了复现算法，需要构造合适的数据集。我们在 Enron 数据集的基础上构建了生成数据集脚本，可以模拟问题的流程：首先在 Enron 数据集中挑选一个子图 G_s 作为问题的考虑范围，然后生成 G_s 的子图 G'_a 作为社交网络公司要发布的社交网络的部分，再在 G'_a 中进行匿名化处理得到最终发布的匿名化网络 G_a ，而攻击者通过在真实的社交网络上爬虫得到了 G_s 的子图 G_c . G_a 和 G_c 之间有 overlap 比例的重合，而攻击者的目标就是在这重合部分中找到节点的对应关系



我们还复现了 *"De-anonymizing social networks"* 中的种子查找算法 seed identification （4.2节），复现了 *"Fast De-anonymization of Social Networks with Structural Information"* 中的去匿名化算法 RoleMatch 和 α-RoleMatch（4.3节），并测试了它们在我们所构建的数据集上的性能

最后我们还尝试了不同的匿名化方法和匿名化程度，来观察它们对抗去匿名化算法的效果

3. Background & Related Works

3.1 De-anonymizing social networks[2]

作为社交网络反匿名化相关领域的开山之作，本文提出了一种通用的匿名社交网络重识别算法。该算法使用seed匹配信息和网络结构，最终实现匿名社交网络的反匿名化

算法为以下几步：

- 数据预处理
  了解社交网络中少量用户的详细信息，通过这些信息，用爬虫方法得到网络中大量的用户和关系。
- 社交网络建模
  本方法在社交网络图之外需要构建攻击者辅助图，攻击者辅助图是一个包含外部辅助信息的无向图，可以帮助攻击者推断出匿名目标图中用户的身份。它可以是任何类型的外部信息，例如用户的年龄、性别、地理位置等。攻击者可以收集这些信息并将它们与匿名目标图相结合，从而更好地推断出用户的身份。
- 节点识别
  - 识别出少量同时存在于匿名目标图和攻击者辅助图中的种子节点，并将它们相互映射。假设攻击者辅助图由k个节点组成，这些节点同时存在于辅助图和目标图中。那么只要知道这些节点中每个节点的度和每对节点的共同邻居的数量就足够了。
  - 运用种子查找算法（4.2节）将匿名目标图中的节点映射到攻击者辅助图相应的节点。
- 社交网络重构
  利用上一步得到的映射关系，不断找出新的映射关系，并加入到原有的关系中。

3.2 An efficient reconciliation algorithm for social networks[4]

这篇论文提出了一种高效的算法，用于在多个在线社交网络中识别同一个用户，它基于一个简单的原理：同一个用户在不同社交网络中的账户会有相似的特征。
该算法可以使用seed匹配信息，也可以不使用

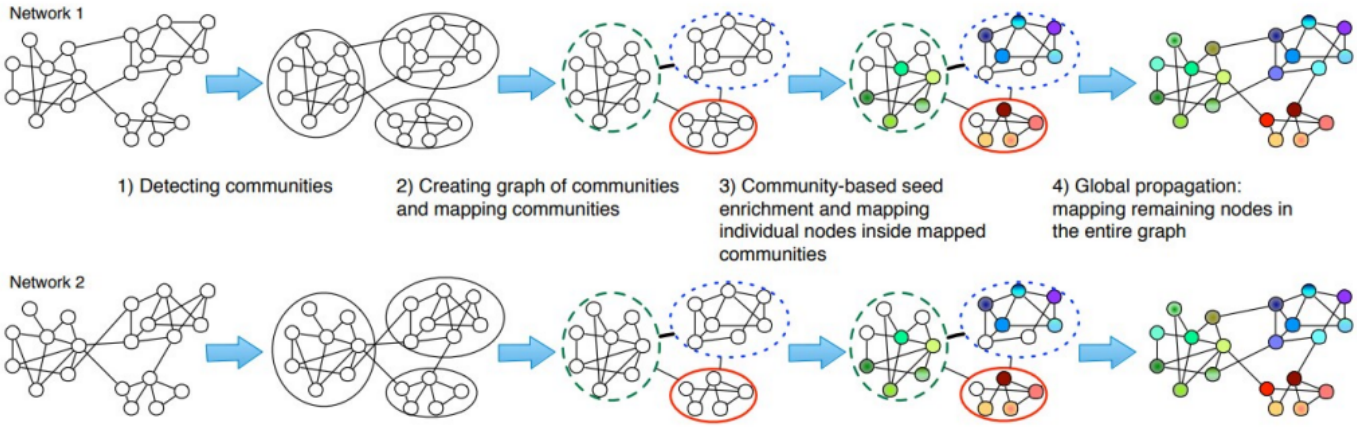
算法为以下几步：

- 对每个社交网络中的用户生成一个特征向量
  - 包含邻居信息：比如邻居的数量、邻居的邻居的数量、邻居之间的连接数等
  - 包含结构特征：比如聚类系数、平均路径长度、介数中心性等
  - 对每个用户，将其邻居信息和结构特征拼接成一个特征向量，并进行归一化处理
- 对每个社交网络中的特征向量进行局部敏感哈希，将相似的向量映射到相同的哈希桶中
  - 局部敏感哈希是对每个特征向量，计算它与一些随机超平面的符号（正或负），并将这些符号拼接成一个二进制串作为哈希值
- 对每个社交网络中的用户，从其他社交网络中的相同哈希桶中找出最近邻的一些候选匹配，并根据相似度阈值和一致性规则确定最终的匹配
- 其中最近邻的距离度量采用余弦相似度
- 相似度阈值是可以自定义的参数，用来过滤掉不够相似的候选匹配

- 一致性规则是指如果一个用户在多个社交网络中都有相同的候选匹配，那么这个候选匹配就更有可能是真正的匹配

在实践中，作者引入了一些seed匹配来初始化去匿名化的过程和评估匹配的准确性，取得了不错的性能提升

3.3 Community-Enhanced De-anonymization of Online Social Networks[3]



这篇论文提出了“社区”这一结构，它是多个有关联的用户组成的社交网络的子图，介于宏观与微观之间。本文使用社区结构提高了去匿名化的效果与鲁棒性。

算法为以下几步：

- 在每个社交网络中，使用社区检测算法来划分用户所属的社区，并为每个社区分配一个唯一的标签
  - 社区检测算法可以是*Louvain*算法，根据网络的模块度（即连接紧密性）来划分；可以是标签传播算法，根据节点的邻居来更新节点的标签，直到标签达到稳定；也可以是基于属性的算法，根据节点的属性（如年龄、性别、地理位置等）的相似性来划分社区
- 在每个社交网络中，为每个用户计算一个特征向量
  - 包括他们的度数、邻居的度数、社区标签和社区内外的边数等
- 在两个社交网络之间，使用贪心算法来匹配相似的特征向量，并将匹配的用户对应起来
- 在匹配的过程中，使用一些启发式规则来优化匹配结果
  - 启发式规则比如优先匹配度数高的用户、避免匹配同一社区内的用户等

同样，作者也引入了一些种子节点来初始化去匿名化的过程和评估匹配的准确性，他们在Flickr，LiveJournal，Orkut和YouTube数据集上进行了实验，并发现该方法还能够处理不同网络之间的结构异构性，即不同网络的社区划分可能不一致或不完全重合的情况。

3.4 Fast De-anonymization of Social Networks with Structural Information[5]

本文介绍了一种快速无seed去匿名化算法RoleMatch，它仅根据结构信息对网络进行去匿名化处理，得益于新的相似性度量 RoleSim++，可以高精度地计算节点相似性。此外在节点匹配阶段，除了节点相似度外，RoleMatch还利用邻域信息来改善映射结果

本文实现的去匿名化算法RoleMatch，输入两个网络G1和G2，输出匹配的节点

- 根据结构信息，计算G1和G2所有节点对之间的相似度
  - 这个相似度计算可以由RoleSim++或者α-RoleSim++等度量实现
  - 值得一提的是，这个相似度计算过程是seedless的，但是它也可以接受种子匹配信息
- 根据相似度得分调用findNodeMatch生成最终节点匹配映射
  - findNodeMatch中调用了NeighborMatch算法，综合相似度和邻域信息得到匹配
- NeighborMatch
  NeighborMatch基于两个观察结果：首先，正确的映射往往具有更高的相似性分数，其次，如果一对节点的邻居是正确的映射，则它们更有可能成为正确的映射
  - 开始时NeighborMatch使用相似性得分最高的一对作为种子
  - 然后，重复匹配匹配邻居数高于阈值 r 的节点对，直到不再有至少匹配 r 个邻居的不匹配对

RoleSim 的详细说明在4.3节

4. Methodology

4.1 Anonymization Algorithms

根据对社交网络匿名化的研究，匿名化算法可以分为 3 类:

- (1) K-匿名算法
- (2) 边随机匿名化算法
- (3) 基于聚类的泛化匿名化算法

K-匿名方法通过边的加减来修改图结构，使得图中的每一个点至少与 K−1 个其他的点在一定的结构特征上不可区分，例如拥有相同的度数，或拥有相同的邻居结构。该方法在匿名化结果上有较好的表现，但是一方面，最优化的 K-匿名方法实现相对复杂；另一方面，对图结构的改变程度比较大，可能对图结构的研究有一定的影响。

边随机方法通过随机添加边、删除边、交换边来修改图结构，它在一定概率上
保证了隐私安全，并且实现起来最简单。

基于聚类的泛化方法首先将结点聚类，然后把每一个子图匿名为一个不带有个体结点具体信息的超点。这样的方法在防止身份识别方面有较好的效果，但是损失了很多个体信息，也损失了规模信息，从而对社交网络分析可能有一定的妨害。

本次实验采用了包括随机添加边、删除边、交换边在内的边随机方法，以及简单更改节点标号的方法，来作为对数据集的匿名化策略

4.2 Seed Identification

在 *"De-anonymizing social networks"* 中，最核心的部分就是种子查找算法 Seed Identification
该算法分为两个阶段。首先，攻击者鉴别出一小部分“种子”节点，这些节点既存在于匿名目标图中，也存在于攻击者的辅助图中，并将它们相互映射。主要的传播阶段是一个自我加强的过程，使用网络的拓扑结构将种子映射扩展到新节点，并将新的映射反馈给算法。最终的结果是辅助网络和目标网络的子图之间的一个大映射，重新识别了后者中所有映射过的节点。

种子查找算法的输入包括：（1）目标图，（2）k个辅助图中的种子节点，（3）k个节点度值，（4）k个公共邻居计数对，以及（5）误差参数θ。算法在目标图中搜索一个独特的k个节点的完全图，其节点度和公共邻居计数在$1 \pm \theta$的因子范围内匹配。如果找到，则算法将完全图中的节点映射到辅助图中相应的节点；否则，报告失败。

除此之外，算法还需要目标图和辅助图之间的部分“种子”映射μ_S。它输出一个映射μ。可以考虑概率映射，但我们发现将注意力集中在确定性的一一映射$\mu : V_1 \to V_2$上更简单。直观地，该算法使用网络的拓扑结构和先前构建的映射的反馈来寻找新的映射。它对轻微的拓扑结构修改具有鲁棒性，例如由数据脱敏引入的修改。在每次迭代中，算法从V_1和V_2之间的累积映射对列表开始。它选择一个未映射的任意节点u在V_1中，并为V_2中的每个未映射节点v计算一个分数，该分数等于已映射到v的邻居中与u相邻的节点数。如果匹配的强度超过阈值θ，则将u和v之间的映射添加到列表中，下一次迭代开始。

还有一些额外的细节和启发式方法需要阐明：

1. Eccentricity：Eccentricity是一种启发式方法，在去匿名化数据库的上下文中使用。它衡量了集合X中的一个项目与其他项目之间的差异程度，其中max和max_2分别表示最高值和第二高值，σ表示标准差。我们的算法测量了映射得分集合的离心率（即v_1中的单个节点与v_2中每个未映射节点之间的映射分数），并且如果离心率得分低于阈值θ，则拒绝该匹配。

$$Eccentricity = \frac{max(X) - max_2(X)}{\delta(X)}$$

2. Edge directionality：为了计算节点对u和v之间的映射得分，算法计算了两个得分——第一个得分仅基于u和v的入边，第二个得分仅基于出边。然后将这些得分相加。
3. 节点度数：如上所述的映射得分有利于具有高度度数的节点。为了补偿这种偏差，每个节点的得分都除以其度数的平方根。与余弦相似度的原理相同。
4. 重新访问节点：在算法的早期阶段，需要处理的映射较少，因此算法会产生更多的错误。随着算法的进行，映射节点的数量增加，错误率降低。因此需要重新访问已映射的节点：重新访问节点时计算的映射可能会不同，因为有新的映射可用。
5. 反向匹配：该算法对两个图的语义完全不关心。G_1是目标图，G_2是辅助图，还是反过来都无所谓。每次节点u映射到v时，都会使用交换输入图的映射得分进行计算。如果v映射回u，则保留映射；否则，拒绝映射。
6. 复杂度：忽略重新访问节点和反向匹配，该算法的复杂度为$O(|E_1|d_2)$，其中d_2是V_2中节点度数的上界。

通过2，3的描述可以看出该算法在计算score时采用的余弦相似度作为度量：

$$cos(X, Y) = \frac{|X \cap Y|}{\sqrt{|X||Y|}}$$

算法伪代码如下：

```
function propagationStep(lgraph, rgraph, mapping)

  for lnode in lgraph.nodes:
    scores[lnode] = matchScores(lgraph, rgraph, mapping, lnode)
    if eccentricity(scores[lnode]) < theta: continue
    rnode = (pick node from right.nodes where
          scores[lnode][node] = max(scores[lnode]))

    scores[rnode] = matchScores(rgraph, lgraph, invert(mapping), rnode)
    if eccentricity(scores[rnode]) < theta: continue
    reverse_match = (pick node from lgraph.nodes where
          scores[rnode][node] = max(scores[rnode]))
    if reverse_match != lnode:
      continue

  mapping[lnode] = rnode

function matchScores(lgraph, rgraph, mapping, lnode)

  initialize scores = [0 for rnode in rgraph.nodes]

  for (lnbr, lnode) in lgraph.edges:
    if lnbr not in mapping: continue
    rnbr = mapping[lnbr]
    for (rnbr, rnode) in rgraph.edges:
      if rnode in mapping.image: continue
        scores[rnode] += 1 / rnode.in_degree ^ 0.5

  for (lnode, lnbr) in lgraph.edges:
    if lnbr not in mapping: continue
    rnbr = mapping[lnbr]
    for (rnode, rnbr) in rgraph.edges:
      if rnode in mapping.image: continue
        scores[rnode] += 1 / rnode.out_degree ^ 0.5

  return scores

function eccentricity(items)

  return (max(items) - max2(items)) / std_dev(items)

until convergence do:
  propagationStep(lgraph, rgraph, seed_mapping)
```

4.3 RoleSim

在 *"Fast De-anonymization of Social Networks with Structural Information"* 中，提出了描述节点相似性的RoleSim++度量

对于两个点u∈G1，u∈G1，我们用$N_1^{out}(u)$，$N_1^{in}(u)$表示u的出边邻居集合和入边邻居集合，同理有$N_2^{out}(v)$，$N_2^{in}(v)$

下面定义u和v之间的最大出入度数
$\Delta^{out}(u,v) = max|N_1^{out}(u)|, |N_2^{out}(v)|$
$\Delta^{in}(u,v) = max|N_1^{in}(u)|, |N_2^{in}(v)|$

再定义$N_1^{out}(u)$和$N_2^{out}(v)$之间的匹配$M^{out}(u,v)$，$N_1^{in}(u)$和$N_2^{in}(v)$之间的匹配$M^{in}(u,v)$

就可以定义$N_1(u)$和$N_2(v)$之间所有可能匹配的最大出边/入边相似性得分
$\Gamma^{out}(u,v) = max_{\{M^{out}(u,v)\}} \sum_{(x,y)\in M^{out}(u,v)} Sim(x,y)$
$\Gamma^{in}(u,v) = max_{\{M^{in}(u,v)\}} \sum_{(x,y)\in M^{in}(u,v)} Sim(x,y)$
其中Sim(x,y)是点x和y的相似性

最终可以定义RoleSim++：

$$Sim(u,v) = (1-\beta)\frac{\Gamma^{out}(u,v) + \Gamma^{in}(u,v)}{\Delta^{out}(u,v) + \Delta^{in}(u,v)} + \beta$$

其中β是一个0到1的系数

5. Experiments & Results

5.1 Generating Data and Anonymization Algorithms

Enron 数据集记录了 36691 名用户之间的发送邮件关系，数据集中每行由两个数字 v_i v_j 构成，表示标号 v_i v_j 的用户之间存在邮件通信。本数据集是无向图，这意味着如果有边 v_i v_j，就会有边 v_j v_i

我们在 Enron 数据集的基础上构建了生成数据集脚本，可以模拟问题的流程：
设 Enron 数据集 $G = \{V, E\}$
我们首先在 Enron 数据集中挑选一个节点子集 $V_s \subset V$，用 V_s 的生成子图 G_s 作为问题的考虑范围

为了使未匿名化的匿名化网络 G'_a 和爬虫网络 G_c 满足关系 $G'_a \cup G_c = G_s$ ，$|V_{G'_a \cap G_c}| = overlap * |V_s|$，我们先从 G_s 中随机选择出一个重合的 $overlap * |V_s|$ 大小的节点子集，然后将 V_s 中不属于此集合的部分平均分配给 V_a 和 V_c，并在它们的基础上构造生成子图 G'_a 和 G_c

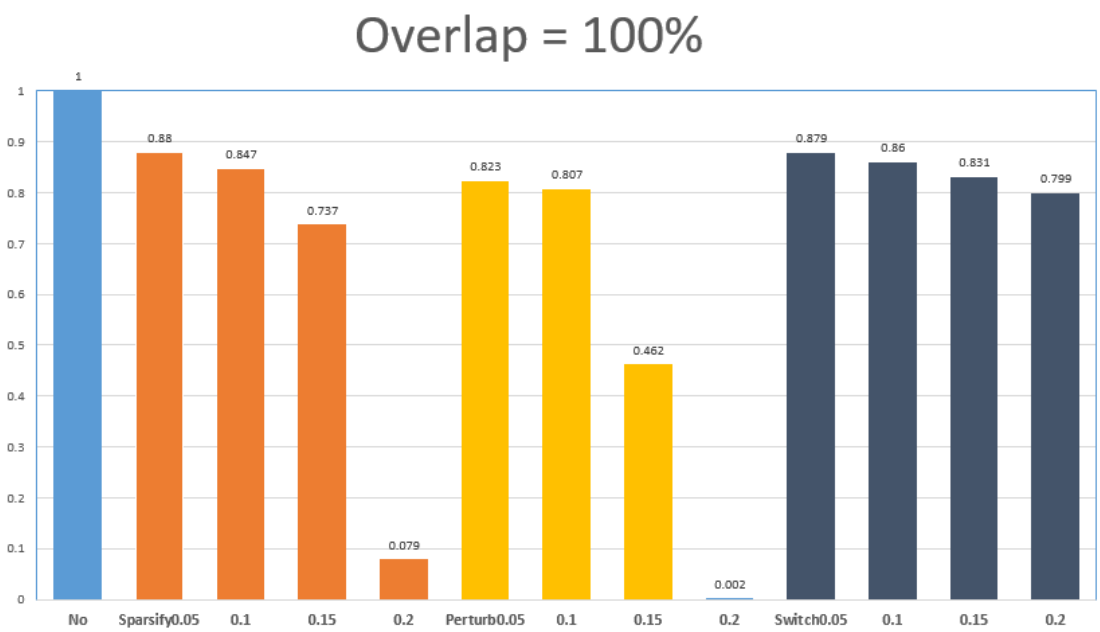
此时需要在 G'_a 上运行匿名化算法，生成最终的匿名网络 G_a

在这里列举在实验中用到的所有匿名化算法。

- 朴素匿名化(Naive):该算法简单地打乱结点的标志(编号)，保持结构不变；
- 稀疏化(Sparsify):稀疏化方法随机移除 p|E|条边，其中，参数 p 控制匿名化程度；
- 扰动(Perturb):扰动方法先与稀疏化一样随机移除边，然后增加不存在的边，直到总的边数和匿名化之前一样。该方法可以看做是对社交网络演化的一种模拟，或者是无目的性的匿名化；
- 交换(switch):交换方法随机选择两条边(u1，u2)，(v1，v2)，并且不存在边(u1，v2)，(u2，v1)。然后交换两条边，也就是说:边(u1，u2)，(v1，v2)被删除，而边(u1，v2)，(u2，v1)被添加。这样的过程重复 p|E|/2 次。

5.1 The Performance of RoleMatch

本部分展示了我们复现的RoleMatch算法的准确率。我们在三个重叠程度的数据集（100%、75%、50%）中做了尝试，数据规模都是3000个节点

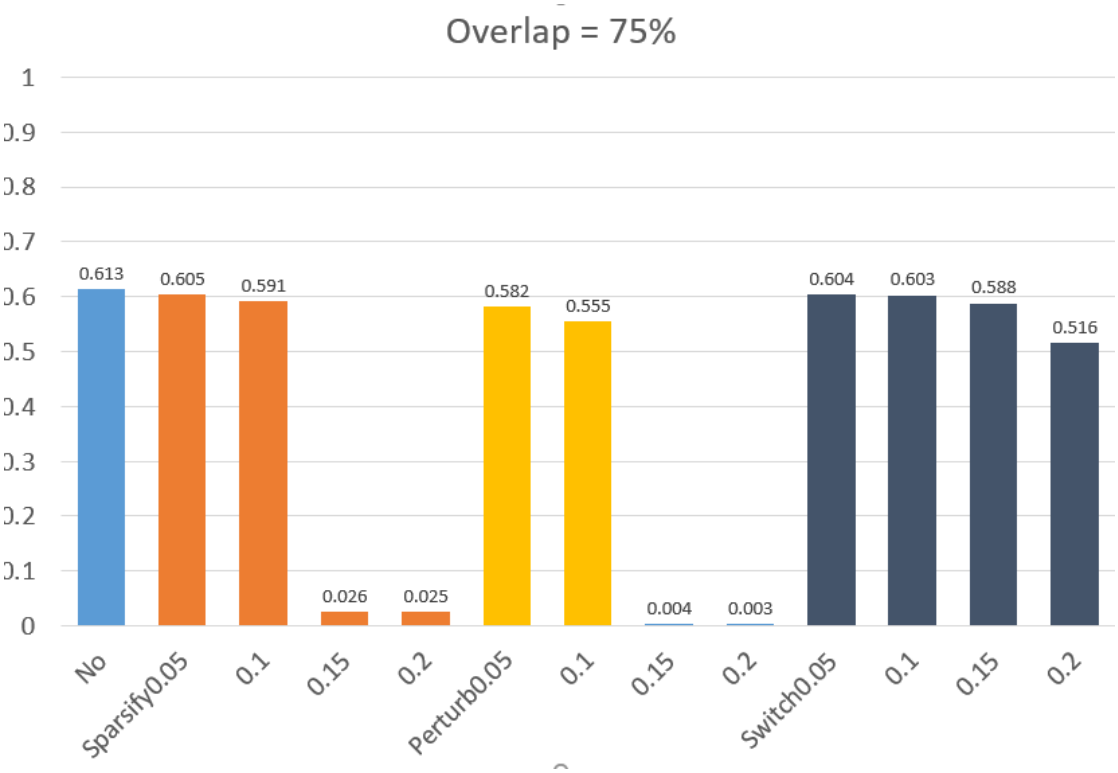


本图是 RoleMatch 算法在重叠程度为 100% 时，能够去匿名化的节点比例，即：

$$\frac{\text{去匿名化节点数}}{\text{重叠比例} \times \text{总节点数}}$$

可以发现，在对于没有进行匿名化的情况下，去匿名化比例为100%
同时，在匿名化程度都不算很高的情况下，去匿名化算法都可以取得良好的效果。

但是当匿名化程度达到0.2，即匿名化网络中 20% 的边被修改（删除、重构）之后，算法的性能急剧下降。这可能和 RoleMatch 算法中的相似性度量 Rolesim++ 的阈值有关，算法不能接受过多的去匿名化操作。也有可能与数据集的结构有关，当随机修改（删除、重构）边之后，大部分节点的特征都变得不再有特点。

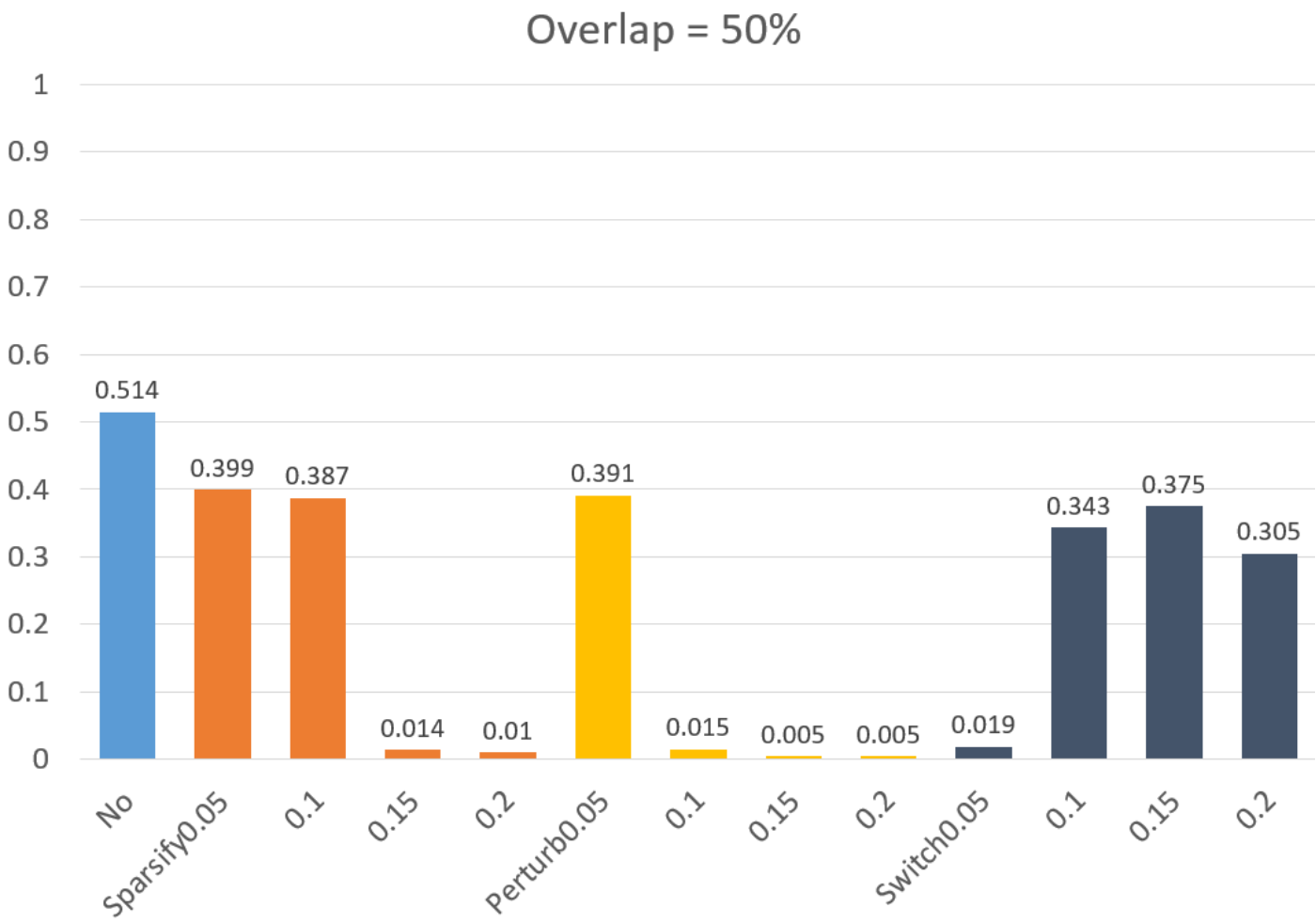


本图是 RoleMatch 算法在重叠程度为 75% 时，能够去匿名化的节点比例

可以注意到此时成功去匿名化的节点比例整体下降了，原因如下：

可能算法将爬虫网络 G_c 与匿名网络 G_a 不重叠的部分匹配到一起了。它们之间可能有某些节点并不对应，但是在 Rolesim++ 度量的计算中它们达到了阈值，故被误认为是对应同一节点。

同时匿名化算法使去匿名化算法崩掉的界限也由 0.2 下降到了 0.15，这也说明了一点，对于更小的重叠来说，去匿名化算法本就要忍受更多多余信息的噪声，所以对匿名化算法的匿名程度也更敏感。

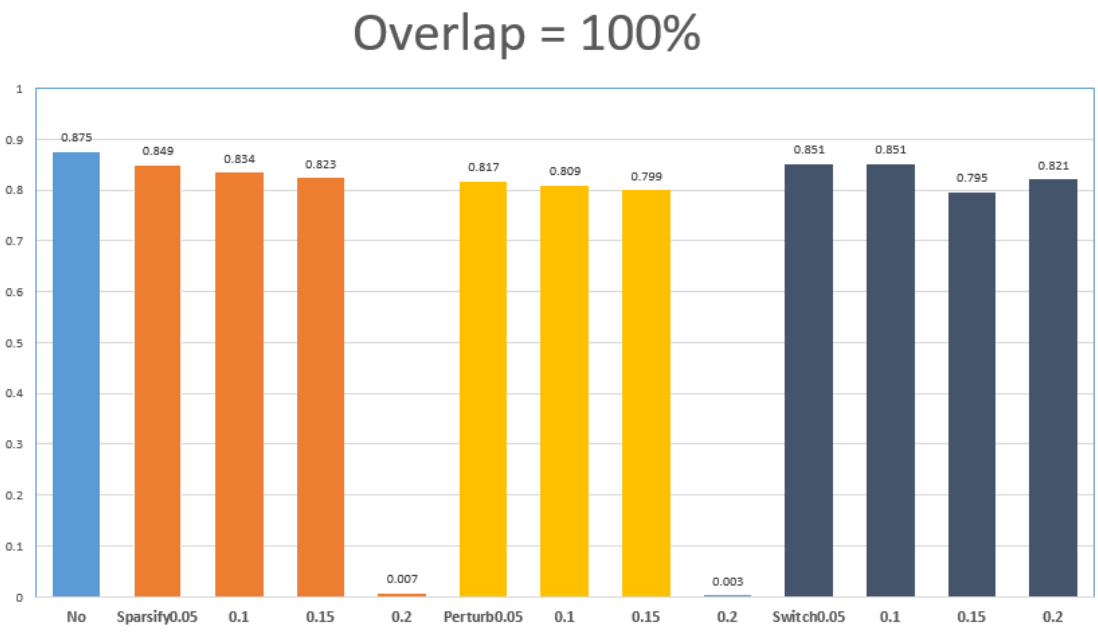


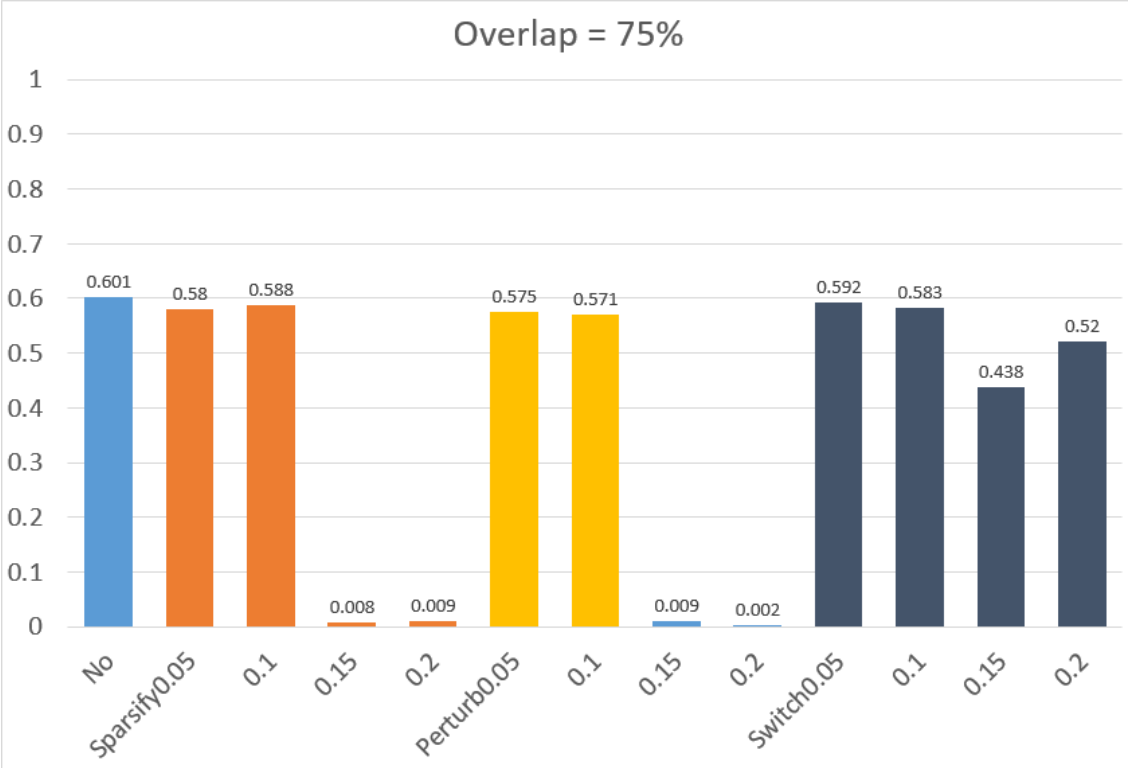
本图是 RoleMatch 算法在重叠程度为 75% 时，能够去匿名化的节点比例

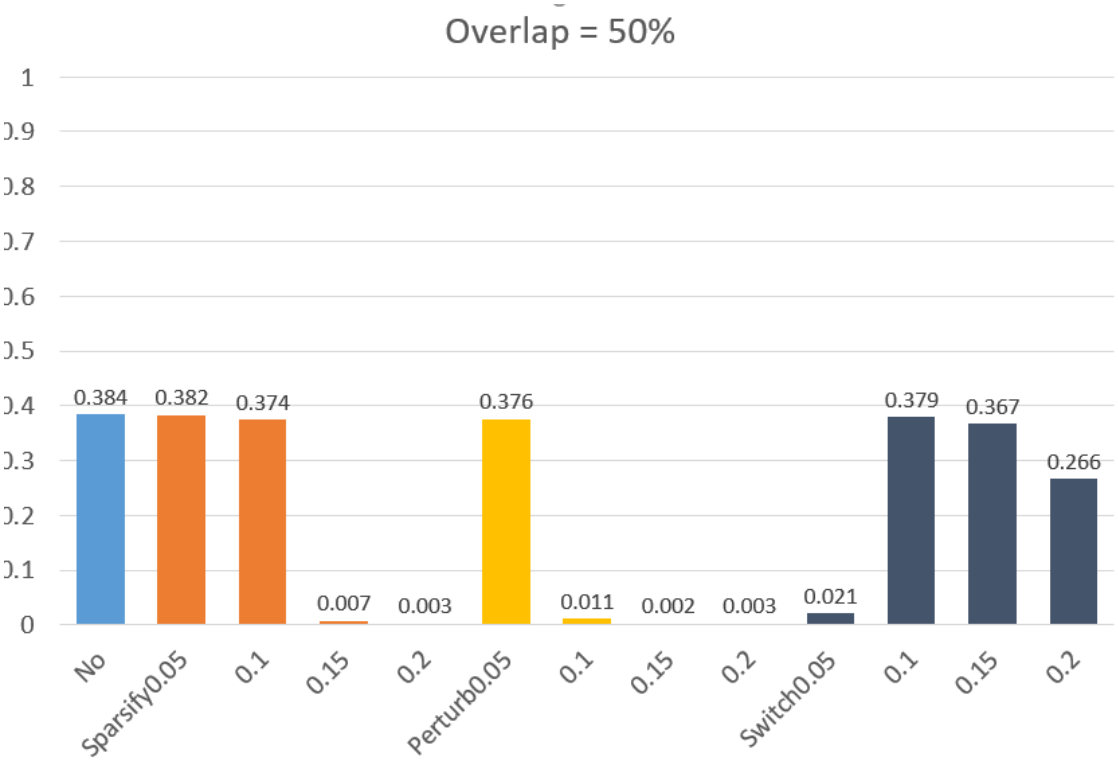
可以注意到此时成功去匿名化的节点比例整体继续下降了，但是即便如此也达到了近 40% 的准确程度

5.2 The Performance of α-RoleMatch

本部分展示了我们复现的α-RoleMatch算法的准确率。我们在三个重叠程度的数据集（100%、75%、50%）中做了尝试，数据规模都是3000个节点

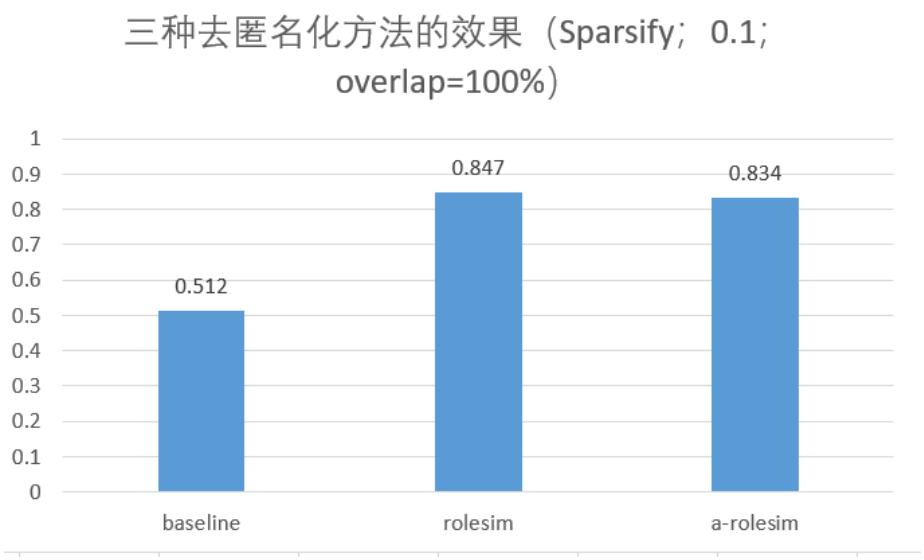






可以发现 α-RoleMatch 与 RoleMatch 相差不大，但是运行时间而言 α-RoleMatch 快了许多

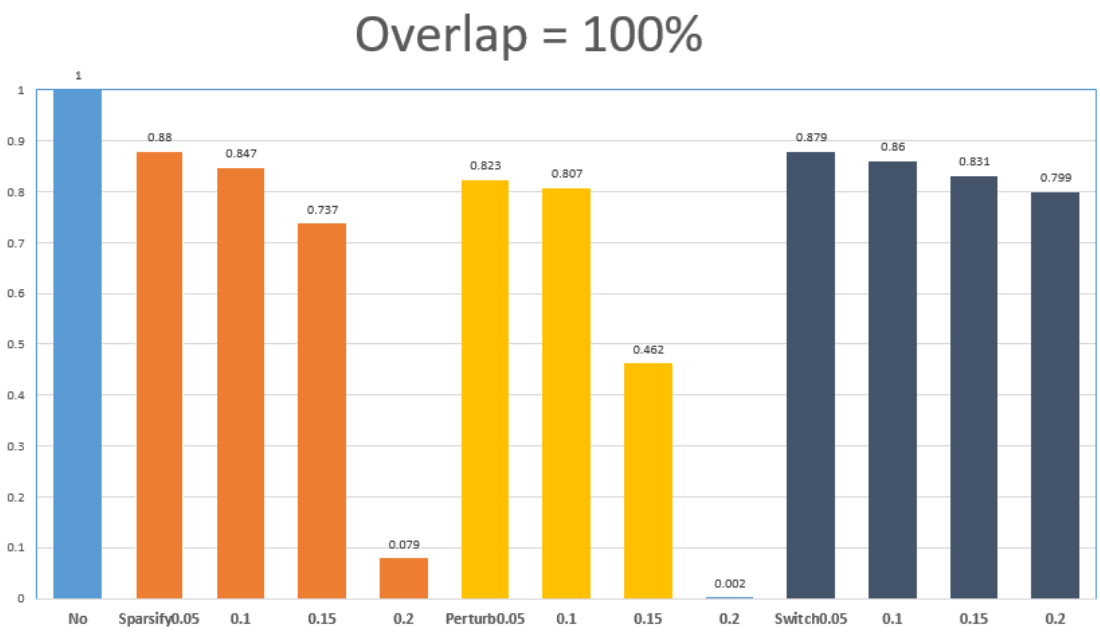
下面是 α-RoleMatch 、RoleMatch 和 baseline 算法的准确率比较，针对重叠率 100% ，10% 稀疏匿名化的情况：



可以发现 RoleMatch 和 α-RoleMatch 的准确率远高于 baseline，说明这两种算法的优越性

5.3 Anonymization Algorithms against De-anonymization Algorithms

在尝试了去匿名化算法的性能之后，我们又想反过来研究不同的匿名化方法和匿名化程度对去匿名化算法的影响。这个工作是很有意义的，因为匿名化和去匿名化是安全攻防的矛与盾。我们相信这部分的结果对社交网络公司采取什么匿名化方案发布网络有着借鉴意义。



本图以 RoleMatch 算法在数据规模3000个节点，重叠程度100%为例：
首先显然匿名化程度越高，去匿名化算法的效果就越差。并且当匿名化程度达到一定高度时，去匿名化算法会彻底崩掉。但是对于社交网络公司而言，选择如此之高的匿名化算法也失去了原有的意义：当对原社交网络结构做出很多改变时，发布这个匿名网络以供研究也失去了效果。

对比不同的匿名化方法，可以发现在相同的匿名化程度下，去匿名化算法的效果：交换(switch) > 稀疏化(Sparsify) > 扰动(Perturb)

这个结果也是可以理解的：

- 交换(switch):交换方法随机删除两条边(u1，u2)，(v1，v2)，然后添加边(u1，v2)，(u2，v1)，它对社交网络的改动是最小的，仅仅影响四个节点，并且这个改动的范围很小（譬如不影响每个点的出入边度数），因此它抵抗去匿名化算法的效果也最差，并且在本数据集中即使是 0.2 程度的交换，也没有让去匿名化算法崩掉
- 稀疏化(Sparsify):稀疏化方法随机移除边，它对社交网络的改动更大了一些，也因此更能抵御去匿名化攻击。但是正因为它的改动更大，在删除了 20% 边的时候，去匿名化算法完全崩掉，可以说明此时网络结构也发生了显著变化
- 扰动(Perturb):扰动方法先与稀疏化一样随机移除边，然后增加不存在的边。它相较于稀疏化而言，还添加了一些随机的误导信息，因而对去匿名化攻击的抵御更强，同时对网络的影响也更大

6. Discussion

我们小组在本学期中阅读并总结了一些关于去匿名化社交网络的论文；构造了一系列数据集；复现了一些去匿名化算法，尝试并比对了它们在数据集上的性能；我们还构造了一系列的匿名化算法，来观察它们对抗去匿名化算法的效果。

我们发现现有的几种去匿名化算法大多是基于度量图的某些结构，其相似性度量往往很需要技巧，且比较复杂。好处是图算法比较传统，相较机器学习、深度学习等方法而言计算开销并不大，但是算法可能太注意技巧而不具有通用意义。我们认为将机器学习、深度学习等方法引入此领域可以使得这一问题的解决更一般化。

从上文的结果可见，针对攻击者的反匿名化攻击，我们依然可以有效地降低损失，针对各种反匿名化攻击，对数据进行有效的匿名化处理可以使被破译的数据量降低，从而保护客户的数据。因此，对于数据保护领域，应该对传统的匿名化方法进行优化，从而更好地应对不断到来的反匿名化攻击。

Reference

[1]: "A Survey of Privacy-Preservation of Graphs and Social Networks" https://link.springer.com/chapter/10.1007/978-1-4419-6045-0_14

[2]: "De-anonymizing social networks" https://ieeexplore.ieee.org/abstract/document/5207644

[3]: "Community-Enhanced De-anonymization of Online Social Networks" https://homes.luddy.indiana.edu/kapadia/papers/community-based-deanon.pdf

[4]: "An efficient reconciliation algorithm for social networks" https://arxiv.org/abs/1307.1690

[5]: "Fast De-anonymization of Social Networks with Structural Information" https://link.springer.com/article/10.1007/s41019-019-0086-8#Sec13