

# 人工智能安全课程

## 第6讲：人工智能隐私性

主讲人：王志波 杨子祺

浙江大学计算机科学与技术（网络空间安全学院）



# 课程大纲

一

隐私保护基本定义

二

机器学习隐私攻击

三

隐私保护机器学习算法

四

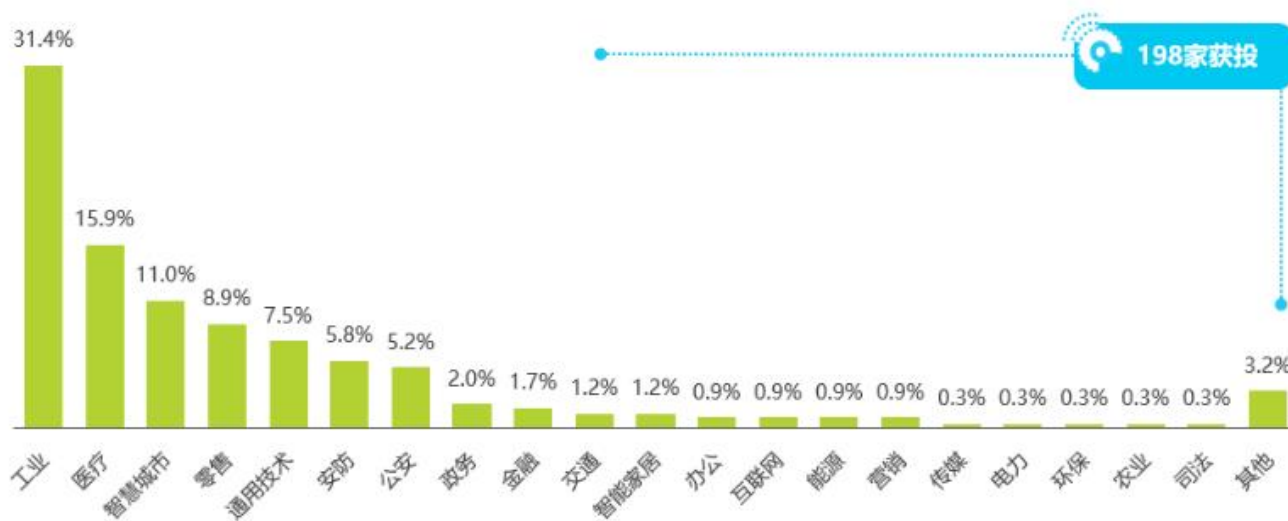
各隐私方案对比分析



## □ 人工智能的成功

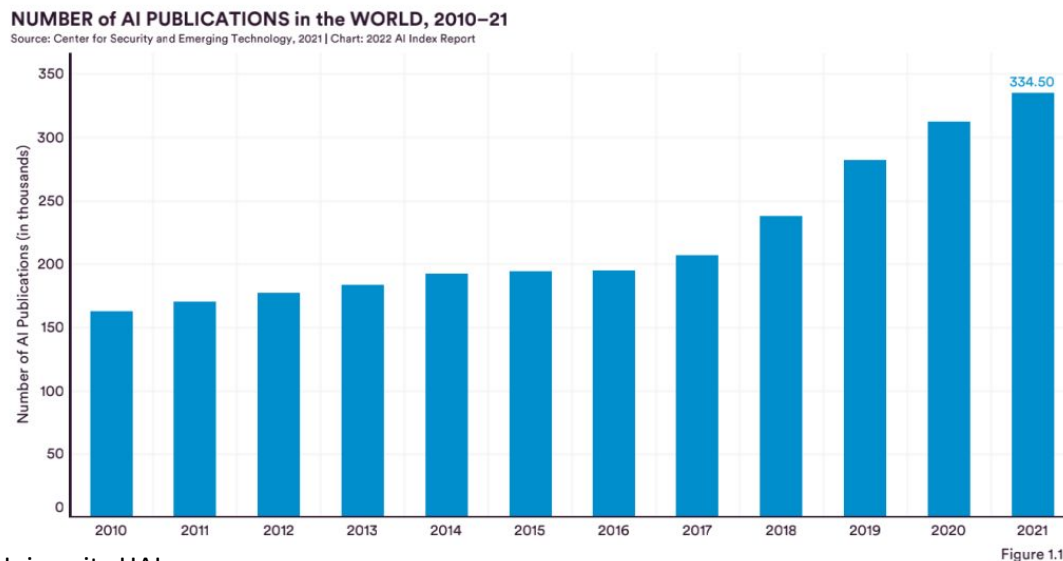
- 2006年, Hinton和他的学生在《Science》上发表了一篇名为 “Reducing the Dimensionality of Data with Neural Networks” 的文章, 开启了深度学习在学术界和工业界的浪潮, 同时也将人工智能推向了一个新的高潮。
- 到目前, 在人工智能的几乎所有领域, 深度学习技术已经远远超过了传统方法的性能, 包括计算机视觉、音频处理、自然语言处理、大数据分析等。

2017年-2021年11月中国计算机视觉获投企业业务领域分布情况



## □ 人工智能的成功

- 推动深度学习在各个领域取得巨大成功主要有以下三因素：
  - **数据井喷**，全球数据中心数据量在未来几年年均增速 40%
  - **计算能力突破**，基于大型 GPU 集群的强大计算能力，使得训练深度神经网络的速度从 2006 年到 2016 年提升了 255 倍。
  - **算法突破**，算法突破推动 AI 技术成熟和实用化



过去的10年，全球AI论文发表量实现翻番，从2010年的162444篇增长到334497篇，且逐年递增。

## □ 隐私泄露风险

- 虽然深度学习带来了巨大的好处，但它需要收集大量的数据，这些数据涉及用户的隐私信息，例如用户兴趣、爱好、个人信息等，这些隐私数据的泄露会导致不可预估的财产以及生命安全问题。
- 2021年7月30日，亚马逊因违反《通用数据保护规范》（GDPR）被欧盟处以创纪录的 8.88 亿美元（人名币57亿）罚款。



**GDPR**的目标是保护欧盟公民免受隐私和数据泄露的影响，同时重塑欧盟的组织机构处理隐私和数据保护的方式。

## □ 隐私泄露风险


- 国外安全研究团队Cyble在一次日常安全监控期间，发现暗网正在销售超过2亿份中国公民的个人信息。经分析，这些数据很可能来自微博、QQ等多个社交媒体，其中还发现了大量湖北省“公安县”的公民数据。





## □ 隐私泄露风险

- 2017年**全国首例AI犯罪案**告破：绍兴的虞小姐收到了好友在QQ上发来的网购代付请求，支付1000余元后，该好友又要求再支付，发觉不对劲的虞小姐立即向好友电话求证，发现好友QQ被盗，立即报了案。

 [绍兴警方侦破全国首例利用AI犯罪案-新华网 - XINHUANET.com](http://www.xinhuanet.com)

- 这条黑色产业链由主要是由**数据获取、数据撞库、数据销售、数据犯罪**四个环节组成
- 撞库：一般而言，黑产最初盗取的账号密码信息往往是粗糙的。但由于人们的同一个邮箱，通常也是多个网站的登陆账号，同样的密码往往也在多个网站使用。因此黑产会通过利用已有的账号密码信息，去批量尝试这些账号密码能否在更多不同的平台上登陆。



## □ 隐私泄露风险

- 撞库的过程中最主要的障碍就是各个网站设置的验证码
- 早期的打码平台，是通过众包让分布在各地电脑前的打码小工来完成的，需要很大的人工成本，并且效率低下
- 因此像 **快啊** 这样的打码平台就开始运用**深度学习技术**训练机器，提高识别验证码的精度和效率，准确率基本能达到98%以上。



网络黑产撞库流程图



## □ 隐私泄露风险

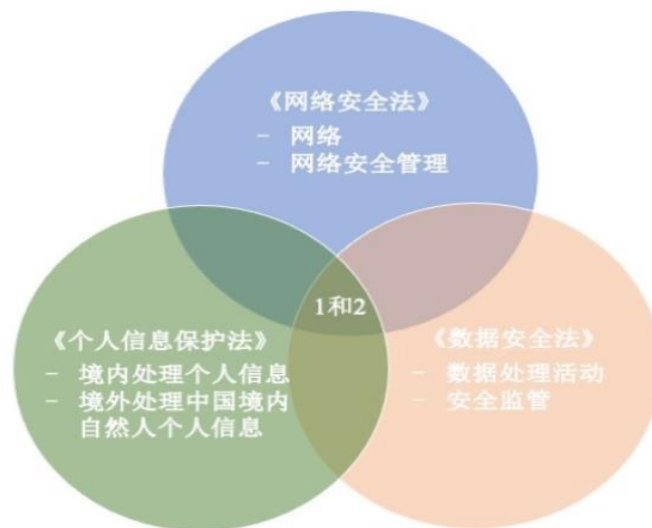
- 网络黑产撞库时，与打码平台是这样合作的：
  - 黑产把已窃取的账号密码信息导入到撞库软件，撞库软件模拟登录协议，向互联网公司的服务器发送登录请求。服务器检测到登录异常时，会通过验证码来进行拦截；
  - 撞库软件将收到的验证码图片发送给打码平台，请求将图片转化为字符。
  - 打码平台后台破解验证码，将字符结果返回给撞库软件，完成撞库流程，得到更多的用户信息。
  - 随后这些信息可能被贩卖、用于诈骗犯罪等，这也就是一开始虞小姐被骗的那一幕了。



网络黑产撞库流程图

## □ 隐私保护法律

- 《中华人民共和国**网络安全法**》于2017年6月1日正式实行
- 《中华人民共和国**数据安全法**》于2021年9月1日正式实行
- 《中华人民共和国**个人信息保护法**》于2021年11月1日正式实行
- 我国形成了以这三法为核心的网络法律体系，为数字时代的网络安全、数据安全、个人信息权益保护提供了基础制度保障。填补了数字法律板块的空白，体现了我国对个人隐私保护的重视。



## □ 隐私保护的挑战

- 在下图场景中，如果我们想要与研究者分享病人的医疗数据，怎么才能保护病人的隐私？
- 传统的做法是将数据匿名化合理
- 这样的做法看似很合理，但是并不能真的保护病人的隐私，因为匿名化后的数据往往保留着许多可能泄露隐私的信息



Birth Date	Gender	ZIP	Disease
1960/01/01	F	10000	flu
1965/02/02	M	20000	dyspepsia
1970/03/03	F	30000	pneumonia
1975/04/04	M	40000	gastritis

match

Medical Records

Name	Birth Date	Gender	ZIP
Alice	1960/01/01	F	10000
Bob	1965/02/02	M	20000
Cathy	1970/03/03	F	30000
David	1975/04/04	M	40000

攻击者从别处获得的信息

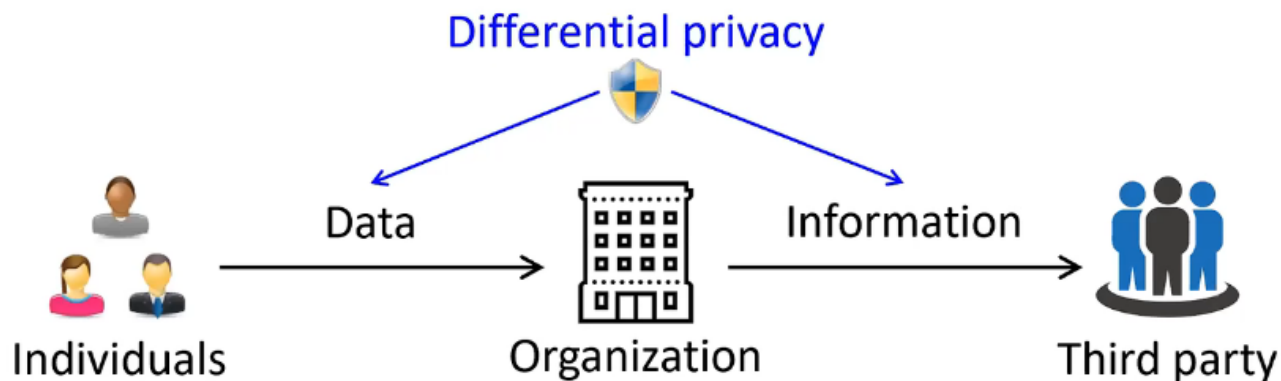
## □ 隐私保护的挑战

- 九十年代中期，美国马萨诸塞州一政府部门就曾遭受这样的攻击，他们发布了匿名化的员工医疗记录用于研究
- 后果：当时州长的医疗记录被泄露
- 后续研究表明：63%的美国人口有着**唯一的**{出生日期、性别、邮编}组合
  - (1)**直接标识符**：利用该属性能直接识别出数据主体，如姓名、身份证号码等
  - (2)**准标识符**：仅利用该属性不能直接识别出数据主体，但联合其他属性或结合背景数据后，能识别出数据主体，如邮编、生日、性别等联合起来在数据集中就可能识别出具体的数据主体

(1)	(2)	Name	Birth Date	Gender	ZIP	Disease
		Alice	1960/01/01	F	10000	flu
		Bob	1965/02/02	M	20000	dyspepsia
		Cathy	1970/03/03	F	30000	pneumonia
		David	1975/04/04	M	40000	gastritis

## □ 隐私保护基本定义

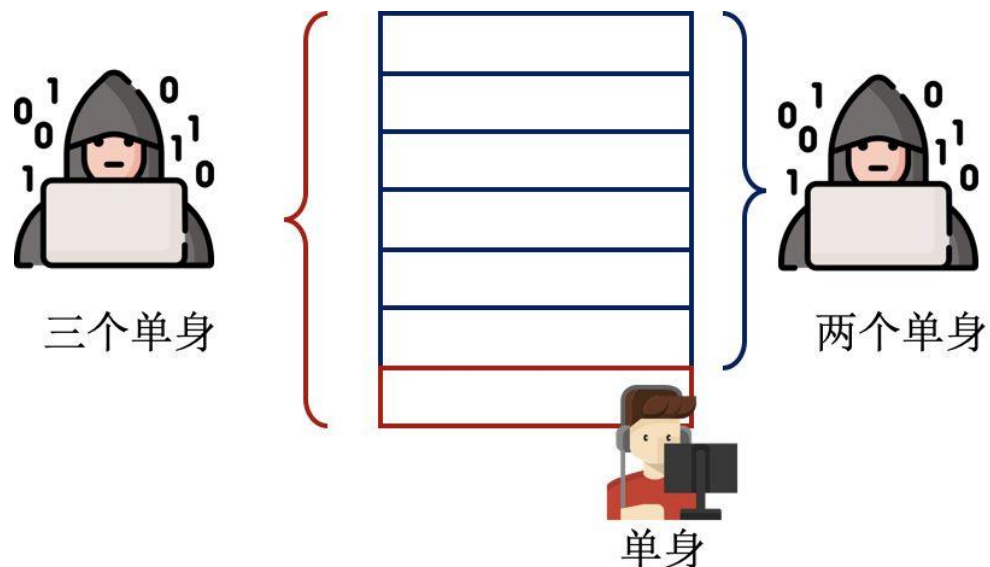
- 隐私保护的概念：是指使个人或集体等实体不愿意被外人知道的信息得到应有的保护<sup>[1]</sup>。
- 隐私保护的目​​的：我们希望，数据使用隐私保护技术后，可以安全发布，攻击者难以去匿名化，同时又最大限度的保留原始数据的整体信息，保持其研究价值。
- 隐私保护目前没有标准的形式化定义，被主流接受的是通过**差分隐私**定义的隐私保护。
- 差分隐私：Dwork等人于2006年提出的一套关于隐私保护的理论框架
- 差分隐私既可以用于数据收集，也可以用于信息分享





## □ 差分隐私 (Differential Privacy)

- 差分攻击：假设现在有一个婚恋数据库，2个单身97个已婚，只能查有多少人单身。攻击者第一次查询发现，2个人单身；张三登记了自己婚姻状况后，攻击者再次查询发现三个单身，所以张三单身。张三作为一个样本的出现，使得攻击者获得了新的知识。
- 差分隐私的目的是**防止差分攻击**，使得攻击者的知识不会因为这些新样本的出现而发生变化。



## □ 差分隐私 (Differential Privacy)

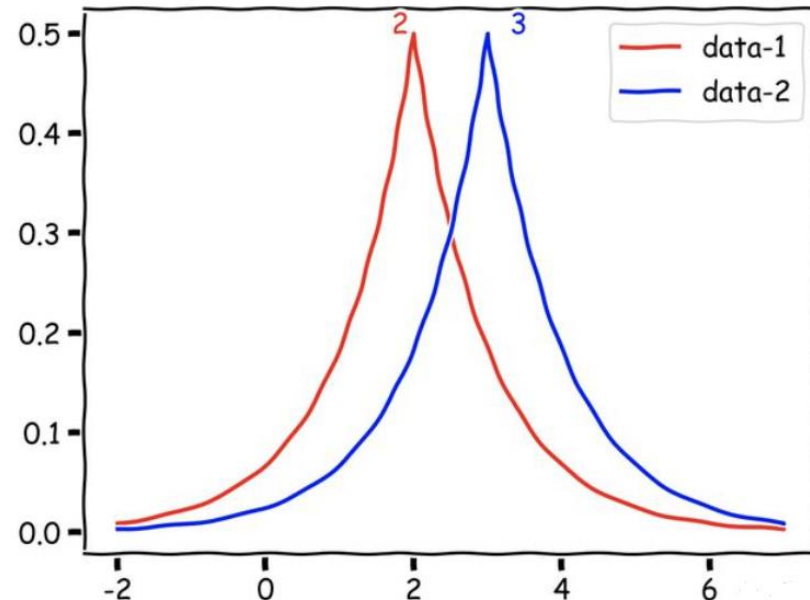
- 实现原理：差分隐私要求我们所发布的原始数据要经过一个 随机算法 来处理，随机算法会对原始数据做一定程度的扰动，即加入随机噪声，来使这个攻击者无法利用扰动后的信息反推某个人是否存在于原数据里面。

- 刚才的例子中，本来两次查询结果是确定的2和3，加入随机噪声后，变成了两个随机变量，画出概率密度函数图

- 如果张三不在数据库的话，得到结果可能是2.5；

张三在的话，得到的结果也可能是2.5；

两个数据集查询得到某一个结果的概率非常接近，以至于我们根本分不清这个结果来自于哪一个数据集，这样也就实现了攻击者的知识不会因为张三这个样本的出现与否而发生变化。



概率密度函数图

## □ 差分隐私的数学描述

- 设有两个数据集分别为 $D$ 和 $D'$ ，将 $D$ 和 $D'$ 中共有的记录从 $D$ 和 $D'$ 中删除，然后将 $D$ 和 $D'$ 合并所形成的新的数据集成为 $D$ 和 $D'$ 的**对称差**，记做 $D \triangle D'$ 。 $|D \triangle D'|$ 表示 $D \triangle D'$ 中记录的数量
- 现有两个数据集 $D$ 和 $D'$ ，它们满足 $|D \triangle D'| = 1$ ， $M$ 为一随机化算法，查询 $f$ 是一个将数据集 $D$ 映射到抽象范围 $R$ 的函数： $f : D \rightarrow R$ ，随机噪声用 $r$ 表示， $M(D) = f(D) + r$ ， $S$ 表示算法 $M$ 的所有可能输出构成的集合。如果算法 $M$ 满足，

$$\frac{\Pr[M(D) \in S]}{\Pr[M(D') \in S]} \leq e^\varepsilon$$

- 则该算法满足 $\varepsilon$ -差分隐私，其中 $\Pr$ 为概率， $\varepsilon$ 为隐私预算。
- 敏感度 $\Delta f$ ：数据集 $D$ 和 $D'$ 满足 $|D \triangle D'| = 1$ ，两次查询函数 $f$ 的最大变化。

$$\Delta f = \max \| f(D) - f(D') \|_1$$

敏感度捕捉的是我们后续需要通过加入噪声来隐藏多大程度的变化差异（对加入的噪声有影响）

## □ 差分隐私的实现机制

- 差分隐私实现隐私保护中，需要处理两大类数据，一类是数值数据比如数据集中已婚人士数量；另一类是非数值型数据，比如喜欢人数最多的颜色
  - 数值型数据，一般采用拉普拉斯机制和高斯机制，即对得到数值结果加入不同分布的随机噪声
  - 非数值型数据，一般采用指数机制
- 拉普拉斯机制：在真实查询  $f$  上添加服从某种拉普拉斯分布的噪声
  - 拉普拉斯分布：

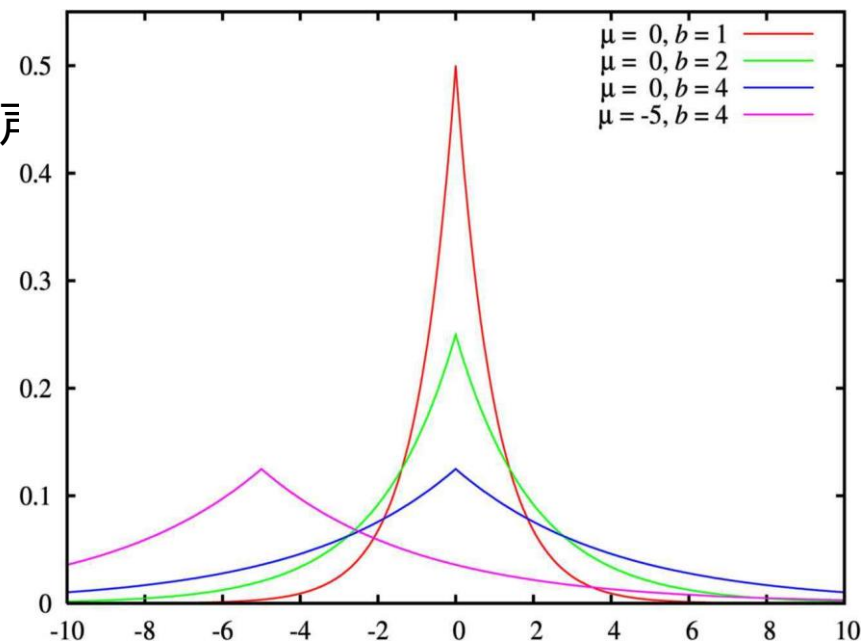
$$Y(x | \mu, b) = \frac{1}{2b} \exp\left(-\frac{|x-u|}{b}\right)$$

$\mu$ 为位置参数， $b$ 为尺度参数

期望为 $\mu$ ，方差为 $2b^2$ 。 $b$ 越大，概率密度函数曲线越“肥”

当添加的分布满足 $\mu = 0$ ， $b = \frac{\Delta f}{\epsilon}$  ( $\Delta f$ 为敏感度)时，

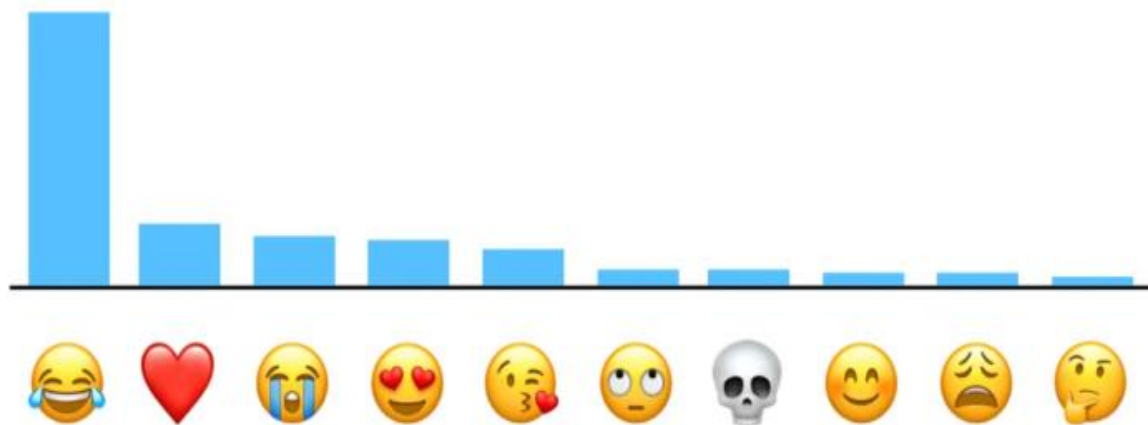
拉普拉斯机制  $M(D) = f(D) + Y$  满足 $\epsilon$ -差分隐私



拉普拉斯概率密度函数图

## □ 差分隐私保护实例

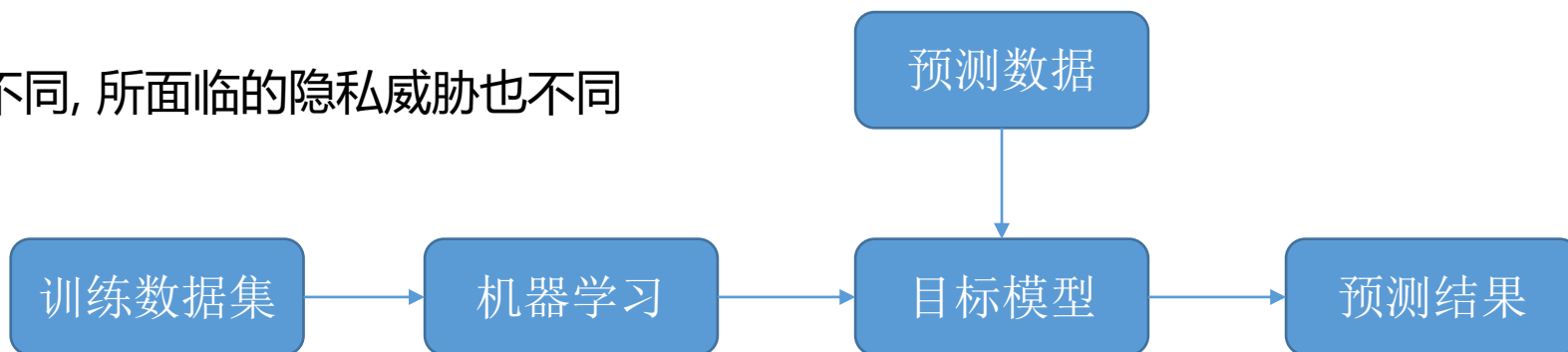
- Apple公司用差分隐私技术收集用户Emoji使用次数，能够统计出最受欢迎的 Emoji，同时又无法确定个人到底最喜欢哪个 Emoji
- 小米和魅族使用差分隐私在定位场景。具体而言，对于一些对定位精准度要求不高的使用场景（比如区域天气查询），用户可以选择提供模糊定位。模糊定位，顾名思义，就是手机会在上传定位信息之前，加上随机噪声，使得上传的数据与真实数据不同，但又在某一范围之内。





## □ 人工智能隐私分类

- 人工智能的热门领域就是机器学习
- 机器学习过程包括训练和预测两个阶段，流程图显示了其解决问题的过程
- 机器学习模型的训练用到大量训练数据，预测结果的准确性直接取决于可用于训练的数据量。而一些数据涉及人们的隐私如个人喜好、身份信息、地理位置、健康数据等，**用户**不希望这些敏感信息被泄漏甚至被攻击者使用
- 同时，机器学习即服务中（MLaaS）**服务商**在为用户提供付费服务时，也希望保护自己的模型不被用户获取
- 由于角色不同，所面临的隐私威胁也不同



## □ 人工智能隐私分类

- 根据攻击对象的不同, 将人工智能隐私分为**数据隐私**和**模型隐私**。

- **数据隐私**

攻击者不能从模型的输出推测出输入数据、训练数据集的有用信息

- **模型隐私**

模型的参数、结构不被攻击者掌握、获取

# 课程大纲

一 隐私保护基本定义

二 机器学习隐私攻击

三 隐私保护机器学习算法

四 各隐私方案对比分析



## □ 隐私攻击分类

### ■ 数据隐私

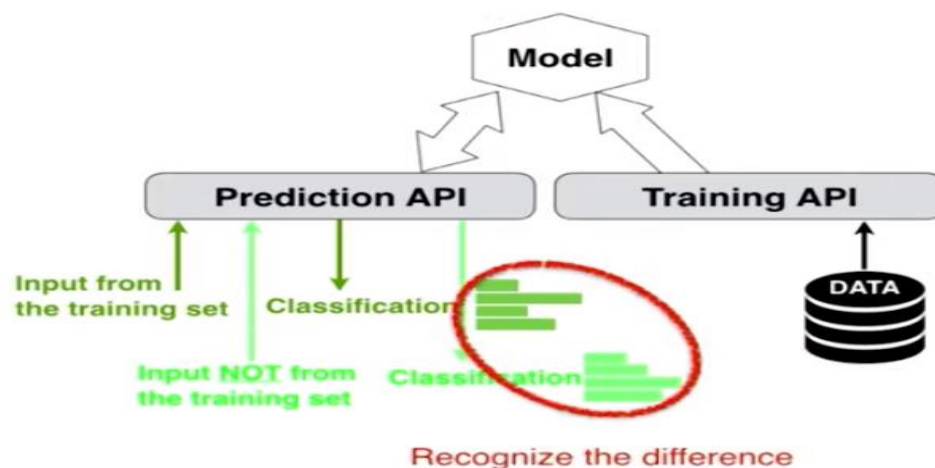
- 成员推断攻击 (Membership Inference Attack)
- 模型逆向 (反演) 攻击 (Model Inversion Attack)

### ■ 模型隐私

- 模型窃取攻击 ( Model Extraction Attacks )

## □ 成员推断攻击

- 攻击者目的：攻击者试图判断某个目标输入样本是否存在于目标模型的训练数据集中
- 攻击者能力：黑盒攻击设置，攻击者只能够访问模型，输入样本可以得到输出向量
- 攻击原理：机器学习模型在其训练数据和非训练数据上的输出向量往往有着不同，在训练数据上表现的比较自信，识别没见过的非训练数据时结果向量分布比较均匀，可据此推断某条数据是否在其训练数据集中



攻击原理示意图



## □ 成员推断攻击

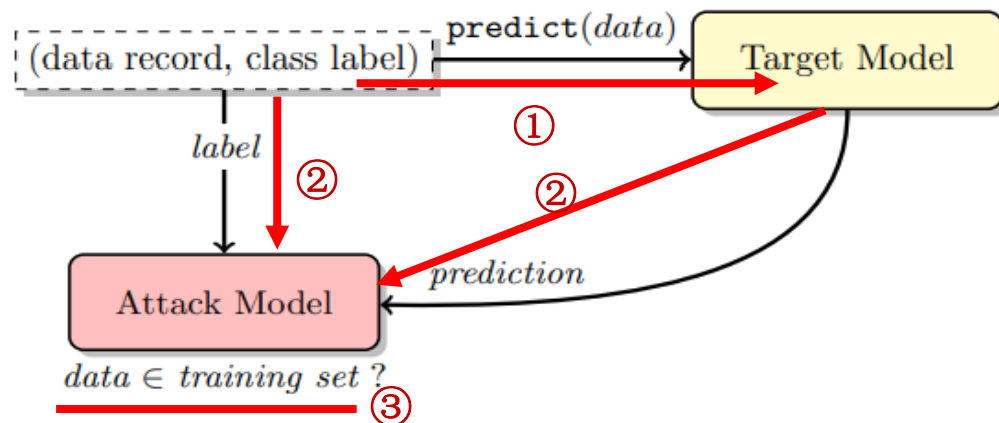
■ 攻击对象：以Google Prediction API为例（用户上传数据集，训练后然后获得模型API）

■ 攻击流程：

① 攻击者将一个目标样本输入到目标模型中，并获得相应的预测结果

② 然后将样本的标签、目标模型的预测结果输入到攻击模型里

③ 攻击模型判断目标样本是否在目标模型的训练集中（故攻击模型是二分类器）



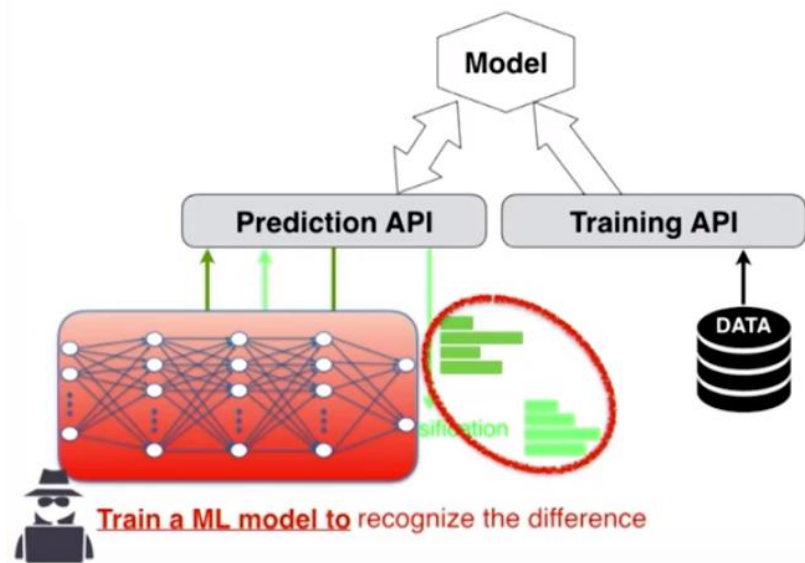
## □ 成员推断攻击

■ 攻击模型训练：（如果能获得目标模型训练集）

① 对于样本 $(\mathbf{x}, y)$ ，输入目标模型，得到输出为预测向量  $\mathbf{y}'$

② 根据该样本是否在目标模型的训练集中，构建  $(y, \mathbf{y}', in)$  或  $(y, \mathbf{y}', out)$  作为攻击模型的训练集， $y, \mathbf{y}'$  是攻击模型的输入， $in$  和  $out$  是标签，训练一个二分类模型。

■ 实际中，最大的挑战在于攻击者显然无法获得目标模型的训练集，于是提出了影子学习技术。



## □ 成员推断攻击

### ■ 影子学习：

步骤一：首先构建与原目标模型训练集相似的数据集

步骤二：在此数据集上训练目标模型的同种模型（影子模型，shadow model），

步骤三：用影子模型来构建  $(y, y', in)$  或  $(y, y', out)$ ，并训练攻击模型



利用影子模型构建攻击模型过程

## □ 成员推断攻击

### ■ 影子学习步骤一：构建与目标模型相似的训练集

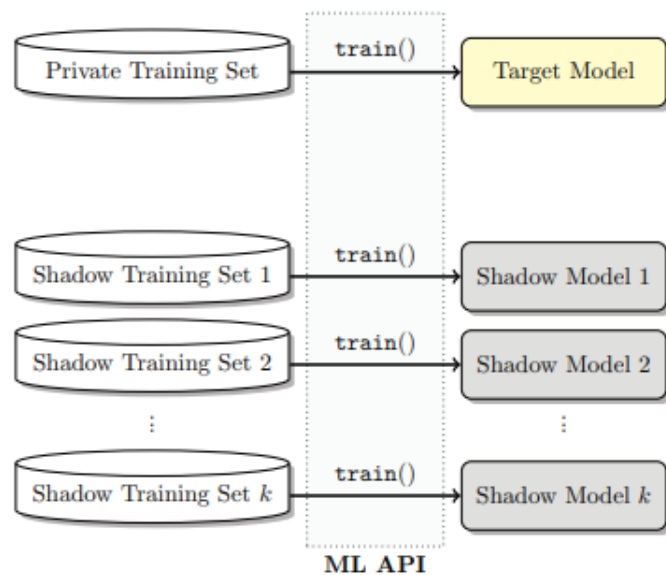
构造训练集有两种方法

- 真实数据：攻击者也许可以获得和目标模型训练集数据相似的数据（知道训练集的类别），可以认为是目标模型训练集的噪声版本，直接利用
- 合成数据
  - a) 基于模型的合成数据：通过不断请求目标模型,使人造数据尽可能接近训练集在模型上的表现
  - b) 基于统计的合成数据：利用有关于目标模型训练数据的统计信息(比如不同特征的边缘分布),合成训练集

## □ 成员推断攻击

### ■ 影子学习步骤二：训练影子模型

- 攻击者利用构造的与目标训练集相似的数据集训练  $k$  个影子模型
- 影子模型的意义就是模仿目标模型的行为，阴影模型越多，攻击模型的攻击效果越好



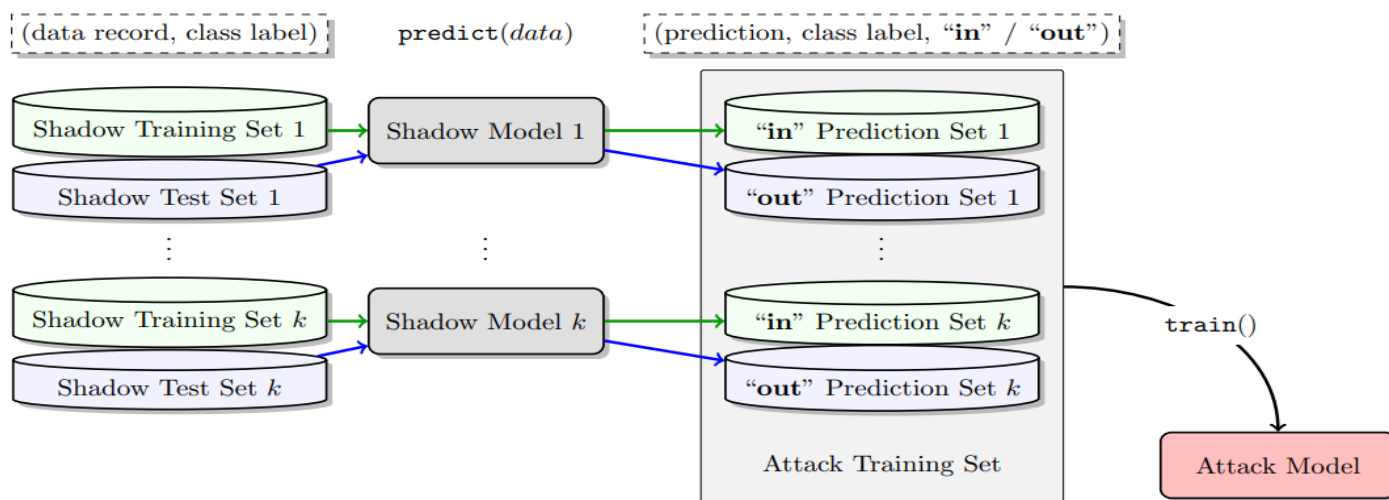
影子模型训练流程



## □ 成员推断攻击

### ■ 影子学习步骤三：训练攻击模型

- 对于影子模型，利用其训练集构建  $(y, y', in)$  作为样本加入到攻击模型的训练集中。
- 再对每一个影子模型构建一个与其训练集同样大小且不相交的测试集，利用其构建  $(y, y', out)$  加入到攻击模型的训练集中。
- 根据 $y$ 的取值将训练集分为 $m$  ( $y$ 的类别数) 部分，每一部分分别训练一个攻击模型（二分类器）



## □ 成员推断攻击

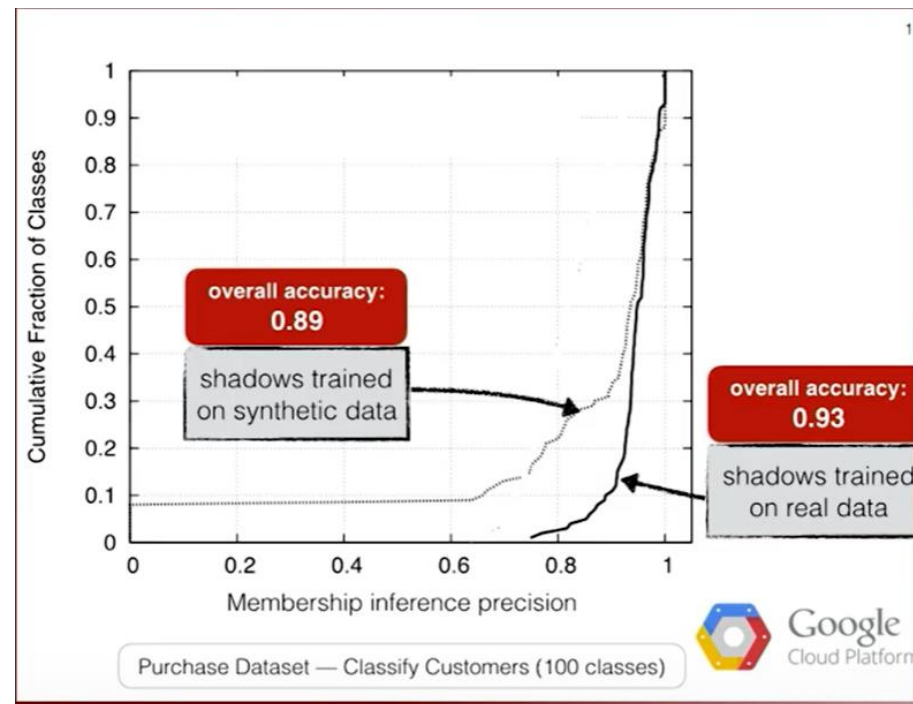
### ■ 攻击设置：

- 数据集：**Purchases**，数千人的shopping历史数据，每个人有不同的购物风格，模型任务是判断一个人的购物风格类别（100类）
- 从目标模型的训练集和测试集中,选出相同数量的样本向攻击模型查询,使攻击模型的baseline accuracy为0.5（等同于随机猜测）。

- 影子模型数量  $k_{\max} = 128$

### ■ 攻击结果：

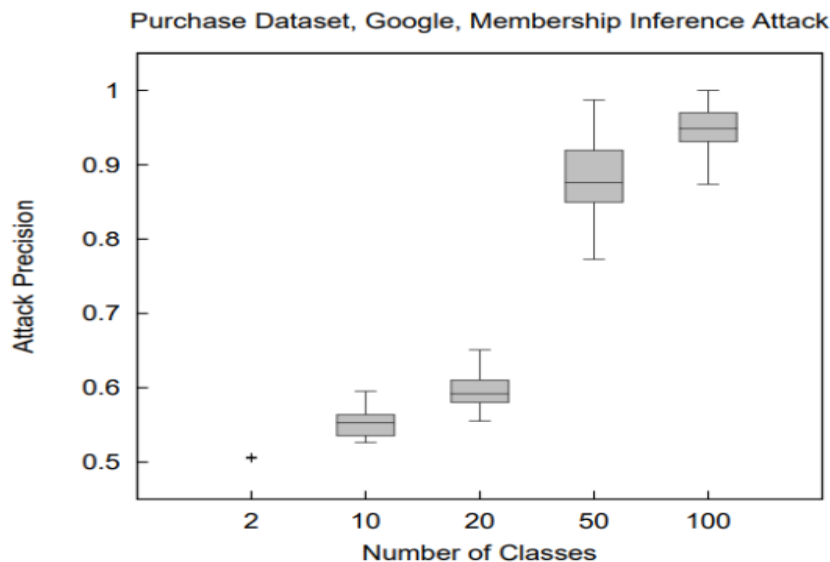
- 影子模型使用真实数据训练：所有类的攻击成功率为0.93
- 影子模型使用合成数据训练：所有类的攻击成功率为0.89



## □ 成员推断攻击

### ■ 攻击结果：

- 由下图可知，目标模型的输出类的数量决定了模型泄漏的程度。类别越多，攻击成功率越高
- 类较少的模型泄漏的关于其训练输入的信息较少。随着类数的增加，模型需要从数据中提取出更具特色的特征，以便能够高精度地对输入进行分类。非正式地说，具有更多输出类的模型需要记住更多关于其训练数据的信息，因此它们会泄漏更多的信息。



## □ 模型逆向攻击

- 攻击者目的：通过模型的输出反推训练集中某条目标数据的部分或全部属性值
- 攻击分类：
  - 白盒模型逆向攻击：攻击者知道模型结构、参数
  - 黑盒模型逆向攻击：攻击者只能访问目标模型，得到输出向量，不知道模型的结构、参数、数据集

## □ 模型逆向攻击

- 案例一：白盒模型逆向攻击
- 攻击者能力：白盒攻击设置，知道模型结构、参数
- 攻击原理：将模型逆向攻击问题转变为一个优化问题，优化目标为使逆向数据的输出向量与目标数据的输出向量差异尽可能地小，即假如攻击者获得了属于某一类别的输出向量，那么他可以利用梯度下降的方法使逆向的数据经过目标模型的推断后，仍然能得到同样的输出向量

## □ 模型逆向攻击

- 攻击对象：人工训练的灰度人脸识别分类器（输入图片，返回置信度向量）

分类器可以建模为以下函数：

$$\tilde{f}: [0,1]^n \mapsto [0,1]^m$$

- 图片具有  $n$  个像素，像素值映射在 $[0,1]$ 范围内，模型输出对应于  $m$  类的概率值向量，第  $i$  个分量对应于图片属于第  $i$ 类的概率。我们将输出的第  $i$ 个分量记作： $\tilde{f}_i(\mathbf{x})$
- 攻击场景：我们已知一个人脸标签label（一个人的姓名），现在我们要通过攻击得到训练集中的图像，本质就是要获得图像像素强度的完整向量，每个强度对应 $[0,1]$ 范围内的浮点值



## □ 模型逆向攻击

### ■ 攻击实现:

---

**Algorithm 1** Inversion attack for facial recognition models.

---

```
1: function MI-FACE( $label, \alpha, \beta, \gamma, \lambda$ )
2:    $c(\mathbf{x}) \stackrel{\text{def}}{=} 1 - \tilde{f}_{label}(\mathbf{x}) + \text{AUXTERM}(\mathbf{x})$ 
3:    $\mathbf{x}_0 \leftarrow \mathbf{0}$ 
4:   for  $i \leftarrow 1 \dots \alpha$  do
5:      $\mathbf{x}_i \leftarrow \text{PROCESS}(\mathbf{x}_{i-1} - \lambda \cdot \nabla c(\mathbf{x}_{i-1}))$ 
6:     if  $c(\mathbf{x}_i) \geq \max(c(\mathbf{x}_{i-1}), \dots, c(\mathbf{x}_{i-\beta}))$  then
7:       break
8:     if  $c(\mathbf{x}_i) \leq \gamma$  then
9:       break
10:  return  $[\arg \min_{\mathbf{x}_i} (c(\mathbf{x}_i)), \min_{\mathbf{x}_i} (c(\mathbf{x}_i))]$ 
```

---

算法描述

- ① 定义一个损失函数  $c$  和一个特定于具体情况的函数  $\text{AuxTerm}$ ，它会把任何可用的辅助信息添加到损失函数中（没有即为0）

$$c(\mathbf{x}) \stackrel{\text{def}}{=} 1 - \tilde{f}_{label}(\mathbf{x}) + \text{AUXTERM}(\mathbf{x})$$

$\tilde{f}_{label}(\mathbf{x})$  是  $\mathbf{x}$  被识别为  $label$  的概率大小，我们希望它越大越好，则  $c(\mathbf{x})$  越小越好。

- ②  $\mathbf{x}$  初始化为0，采用梯度下降优化，使  $\mathbf{x}$  朝着梯度方向改变

## □ 模型逆向攻击

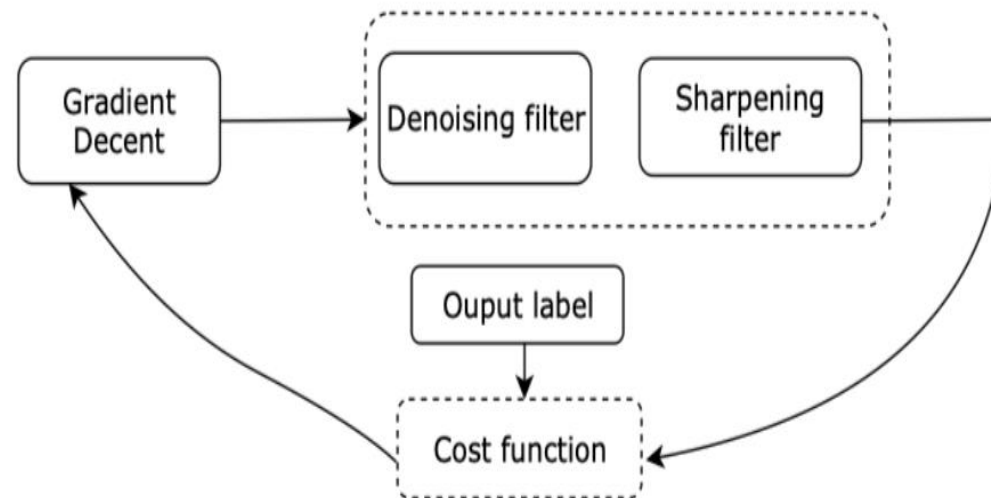
### ■ 攻击实现：

**Algorithm 1** Inversion attack for facial recognition models.

```
1: function MI-FACE( $label, \alpha, \beta, \gamma, \lambda$ )
2:    $c(\mathbf{x}) \stackrel{\text{def}}{=} 1 - \tilde{f}_{label}(\mathbf{x}) + \text{AUXTERM}(\mathbf{x})$ 
3:    $\mathbf{x}_0 \leftarrow \mathbf{0}$ 
4:   for  $i \leftarrow 1 \dots \alpha$  do
5:      $\mathbf{x}_i \leftarrow \text{PROCESS}(\mathbf{x}_{i-1} - \lambda \cdot \nabla c(\mathbf{x}_{i-1}))$ 
6:     if  $c(\mathbf{x}_i) \geq \max(c(\mathbf{x}_{i-1}), \dots, c(\mathbf{x}_{i-\beta}))$  then
7:       break
8:     if  $c(\mathbf{x}_i) \leq \gamma$  then
9:       break
10:  return  $[\arg \min_{\mathbf{x}_i} (c(\mathbf{x}_i)), \min_{\mathbf{x}_i} (c(\mathbf{x}_i))]$ 
```

算法描述

- ③ 每次梯度下降后的  $\mathbf{x}$  被提供给一个后处理函数 Process，它可以根据攻击的需要执行各种图像处理，例如去噪和锐化
- ④ 如果损失在  $\beta$  次迭代中未能减小，或者小于等于  $\gamma$ ，则终止梯度下降过程并返回  $c(\mathbf{x})$  最小的  $\mathbf{x}$ 。



Process流程图

## □ 模型逆向攻击

- 攻击结果：
  - 下图中左边为提取出的人脸，右边是训练集中的人脸
- 能够恢复出肉眼可以辨别的模糊图像



攻击结果

## □ 模型逆向攻击

### ■ 案例二：黑盒模型逆向攻击

#### ■ 攻击者能力：

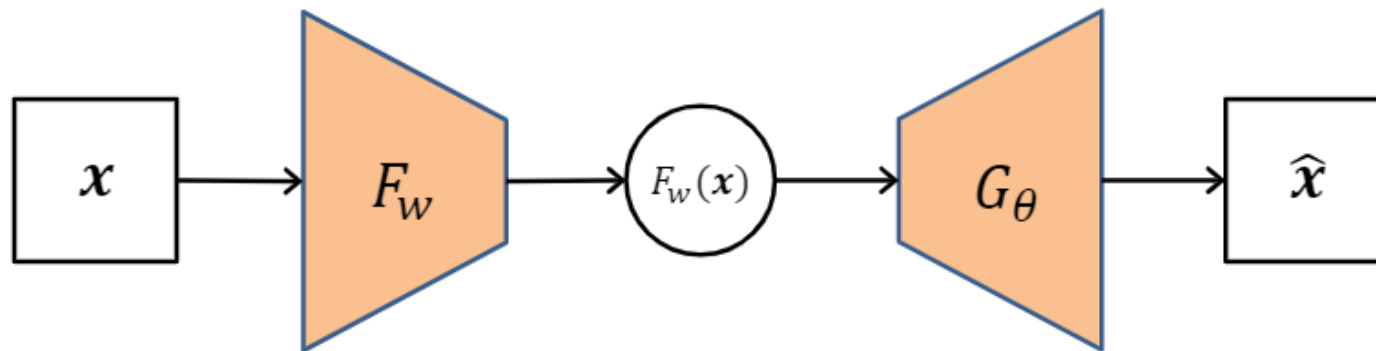
- 黑盒攻击设置，攻击者不知道目标模型结构、参数和训练集
- 知道模型训练数据集的大致分布，可以从更“通用”的数据分布中采样
- 知道目标模型输入和输出数据结构
- 访问黑盒模型得到的输出不是完整向量，是截断向量

截断向量：保留置信度向量  $g$  中前  $m$  个大的维度，其他取0，记为  $\text{trunc}_m(g)$ 。如下  $m = 2$

$$\text{trunc}_2((0.6, 0.05, 0.06, 0.2, 0.09)) = (0.6, 0, 0, 0.2, 0)$$

## □ 模型逆向攻击

- 案例二：黑盒模型逆向攻击
- 攻击原理：训练一个额外的反演模型进行样本重构
  - $x$  是目标模型输入样本， $F_w$  是目标模型， $F_w(x)$  是输出预测向量。
  - 我们现在需要训练一个反演模型  $G_\theta$ ，反演模型将给定预测向量作为输入并输出重构的样本  $\hat{x}$
  - $F_w$  是给定的不会修改的， $F_w$  的训练集攻击者并不知道



黑盒模型逆向攻击原理示意图

## □ 模型逆向攻击

- 案例二：黑盒模型逆向攻击
- 攻击对象：人脸识别分类器
- 攻击模型训练：

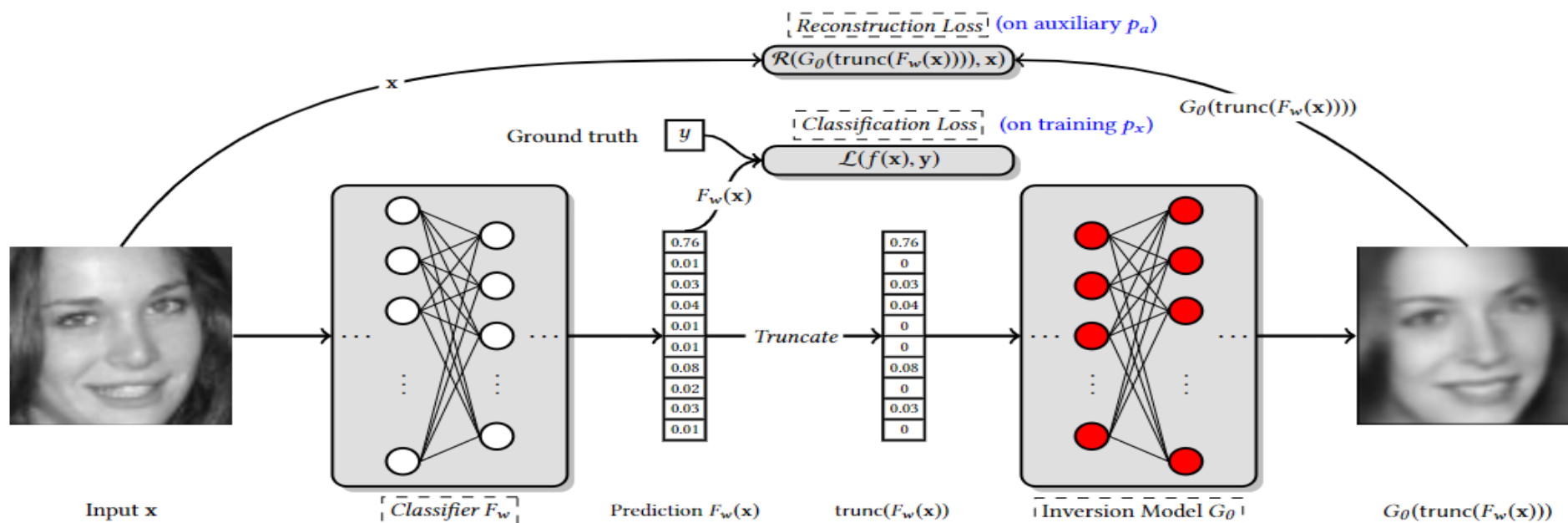
- 步骤一：获得  $G_\theta$  的训练集（辅助集）

从比目标模型训练数据集分布更广的数据分布中采样。以人脸识别为例，假设分类器分类1000个人，则辅助集可以是网络上公共人脸数据集的总和中选取。



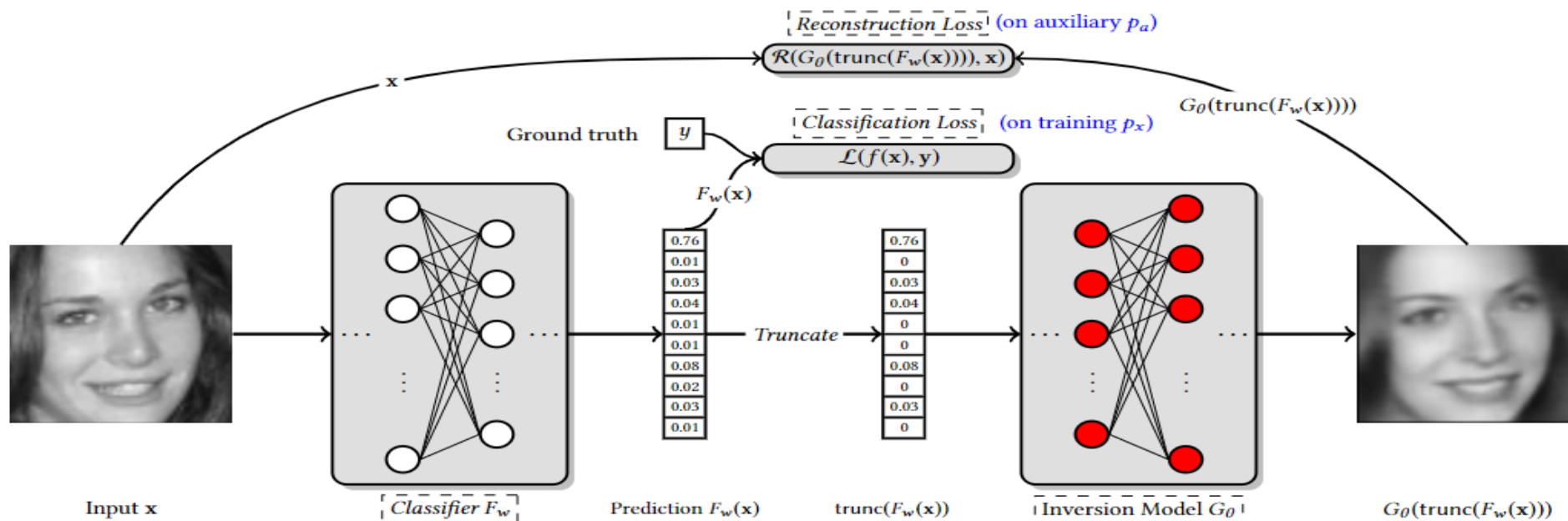
## □ 模型逆向攻击

- 案例二：黑盒模型逆向攻击
- 步骤二：提出一种截断方法训练  $G_\theta$  模型
  - 将分类器在辅助集样本上的预测向量截断到和训练集样本上预测向量相同维度，并使用它们作为输入特征来训练反演模型，使得它被迫从截断的预测最大限度地重建输入数据。



## □ 模型逆向攻击

- 案例二：黑盒模型逆向攻击
- 步骤二：提出一种截断方法训练 $G_\theta$ 模型
- 截断过程可以理解为 通过移除 $F_w(a)$ （辅助集中的样本 $a$ 经过目标模型的预测输出）中具有小置信度的类，来进行特征选择。它有助于减少 $G_\theta$ 的过拟合，并且仍然可以从保留的重要类中重建输入数据。



## □ 模型逆向攻击

### ■ 案例二：黑盒模型逆向攻击

### ■ 步骤二：提出一种截断方法训练 $G_\theta$ 模型

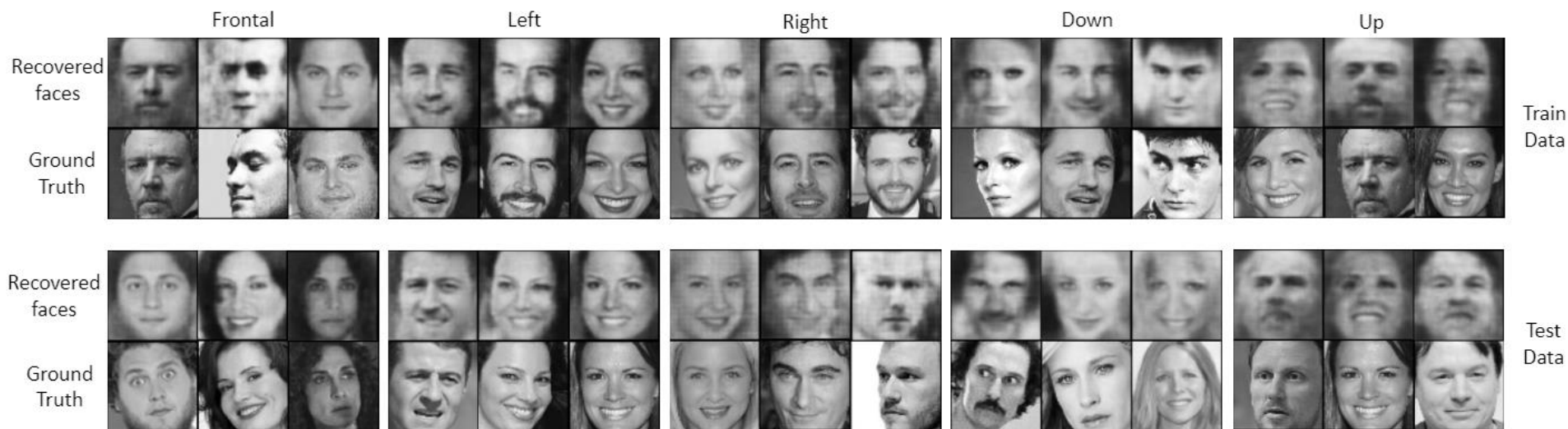
- $G_\theta$ 训练损失函数：
$$C(G_\theta) = \mathbb{E}_{a \sim p_a} [\mathcal{R}(G_\theta(\text{trunc}_m(F_w(a))), a)]$$

$a$ 为辅助集中的样本， $F_w(a)$ 表示经过目标模型的预测输出， $\text{trunc}_m(F_w(a))$ 表示目标模型输出的截断，

$G_\theta(\text{trunc}_m(F_w(a)))$ 是重构输出， $\mathcal{R}$ 是重构损失，用来衡量重构后的输出和辅助集原输出的误差，我们希望这个误差越小越好

## □ 模型逆向攻击

- 案例二：黑盒模型逆向攻击
- 实验设定与结果：
  - 目标模型：用FaceScrub 数据集训练的人脸分类器
  - 辅助集：使用CelebA官方数据集进行不同角度的裁剪得到辅助集
  - 结果：绝大多数重构图片能够被肉眼识别出



## □ 模型窃取攻击

- 攻击者目标：试图通过访问目标模型窃取模型参数或者功能的攻击
- 攻击者能力：目标模型参数不公开,攻击者不知道或者已知部分模型结构信息和标签信息
- 攻击者发起模型窃取攻击的动机：
  - 模型商业价值：避免向模型训练服务缴费
  - 提高对抗样本攻击成功率：模型被窃取后，攻击者可以进一步部署白盒对抗攻击来欺骗在线模型，这时模型的泄露会大大增加攻击的成功率。

## □ 模型窃取攻击

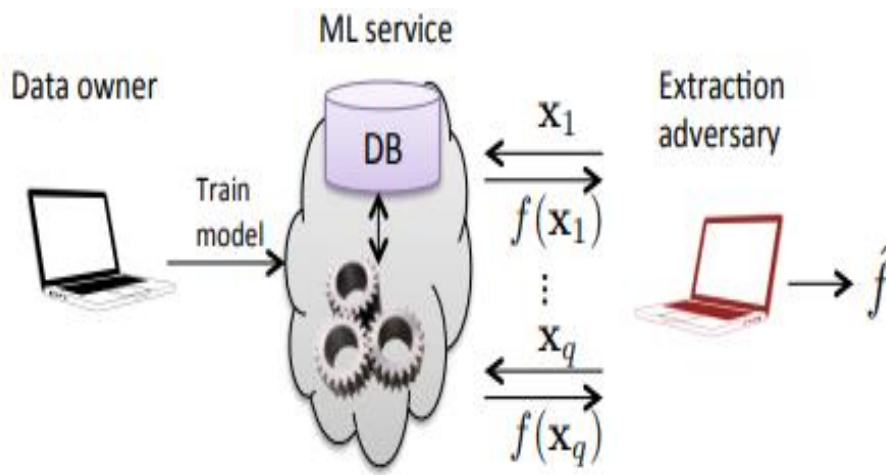
### ■ 主要分类：

- 方程求解攻击
- 基于替代模型的模型窃取



## □ 方程求解攻击

- 攻击对象：针对支持向量机（SVM）等传统的机器学习方法的模型窃取攻击。
- 攻击者知识：攻击者可以先获取模型的算法、结构等相关信息，能够查询模型获取结果
- 攻击原理：对于逻辑回归（LR）、支持向量机(SVM)、神经网络（NN）这类算法而言，我们可以将其模型看做是一个函数 $f(x)$ ，模型的输入是 $x$ ，模型的输出结果是 $f(x)$ ，模型在训练过程中通过优化函数 $f(x)$ 里的参数来达到分类的目的。要进行模型窃取就是求解 $f(x)$ 参数值，实际上就是解方程过程。



攻击原理示意图

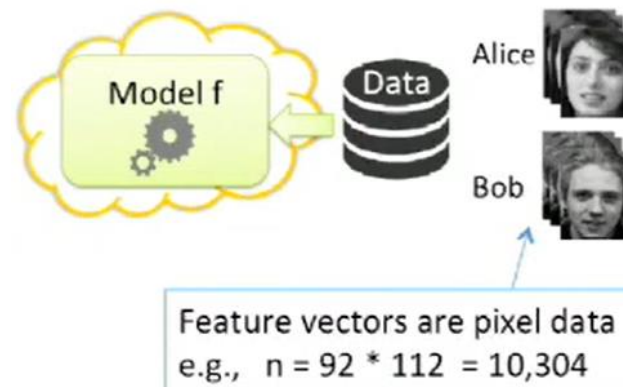
## □ 方程求解攻击

■ 案例：考虑两个人的人脸分类任务， $f > 0.5$  分类为 “Alice”，反之为 “Bob”

- 目标模型使用逻辑回归模型：

$$f(x) = \frac{1}{1 + e^{-(w \cdot x + b)}}$$

$$\text{变形为 } \ln\left(\frac{f(x)}{1-f(x)}\right) = w \cdot x + b$$



- $w \cdot x + b$  是线性函数，函数的参数是  $w$  和  $b$ ，其中  $w$  是  $n$  维的权重向量， $b$  是偏置向量。
- $w$  是  $n$  维的权重向量， $b$  是 1 维的偏置向量，因此总共有  $n+1$  个参数需要求解，也就是说，至少需要  $n+1$  个等式就可以解出这个方程。所以从理论上而言，我们只需要查询  $n+1$  次便可窃取到这个逻辑回归模型。

## □ 方程求解攻击

### ■ 攻击结果：

- 在Amazon 提供的逻辑回归机器学习即服务上，攻击成功率接近百分之百（Digits数据集较简单，训练的模型简单，查询次数少，速度快；Adult数据样本属性多，模型较复杂，查询次数多）

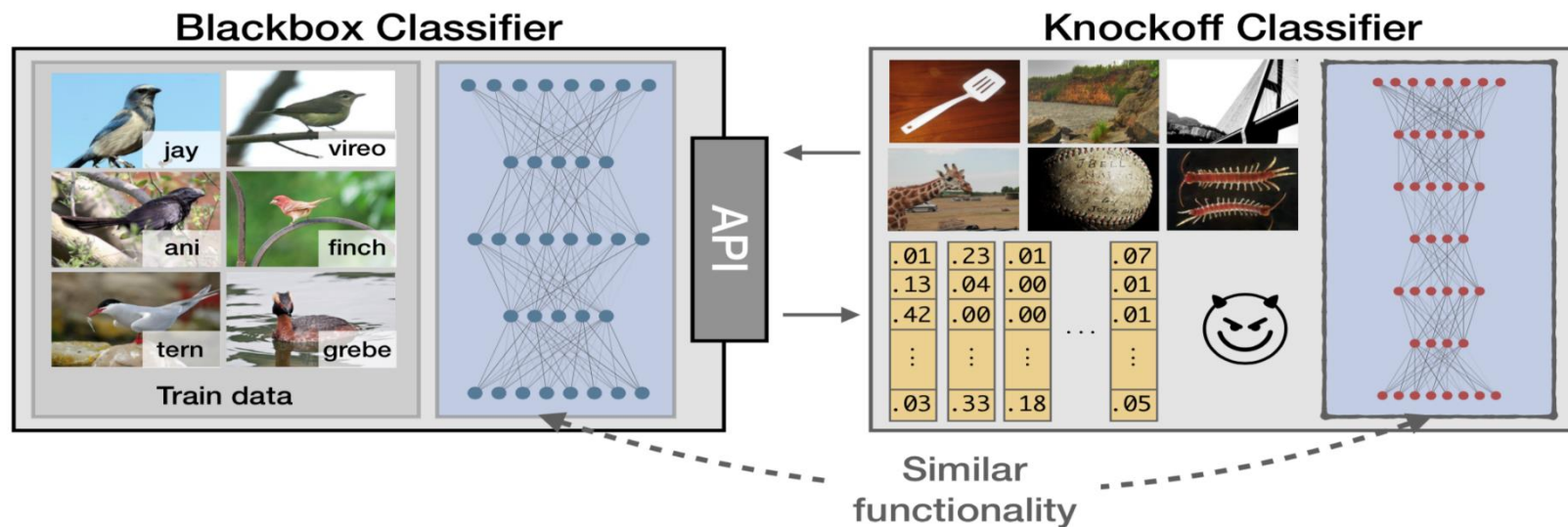
Service	Model Type	Data set	Queries	Time (s)
Amazon	Logistic Regression	Digits	650	70
	Logistic Regression	Adult	1,485	149

攻击结果示意图

- 缺点：**方程求解攻击需要攻击者了解目标算法的类型、结构、训练数据集等信息，无法应用于复杂的神经网络模型。

## □ 基于替代模型的模型窃取攻击

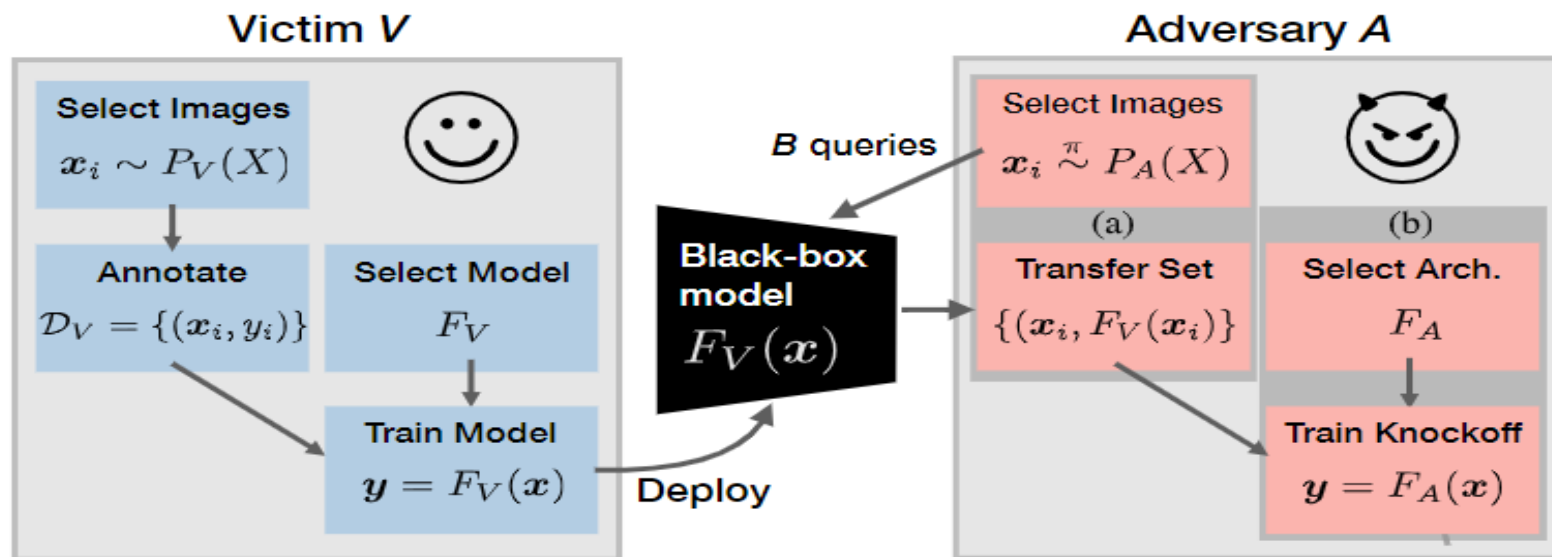
- **攻击原理**：在本地训练一个与目标模型任务相同的替代模型，当经过大量训练之后，该模型就具有和目标模型相近的性质
- **区别**：与方程求解攻击不同，攻击者对于目标模型的**具体结构**并不了解，训练替代模型不是为了获取目标模型的具体参数，而只是利用替代模型去拟合目标模型的功能。



攻击原理示意图：  
1.向目标模型查询一组输入图像，并获得模型给出的预测  
2.使用上一步得到的“图像-预测”对训练一个knockoff（即替代模型）

## □ 基于替代模型的模型窃取攻击

- 攻击方案：
- Victim V（受害者V）：为一个具体任务训练一个CNN模型  $F_V$ ，提供模型黑盒访问接口
- Adversary A（攻击者A）：通过黑盒访问受害者模型，训练一个替代模型
- 攻击者这些信息是不可知的：1.  $F_V$ 模型的内部构造；2. 训练集和测试集；3. 训练集K个类别的语义信息

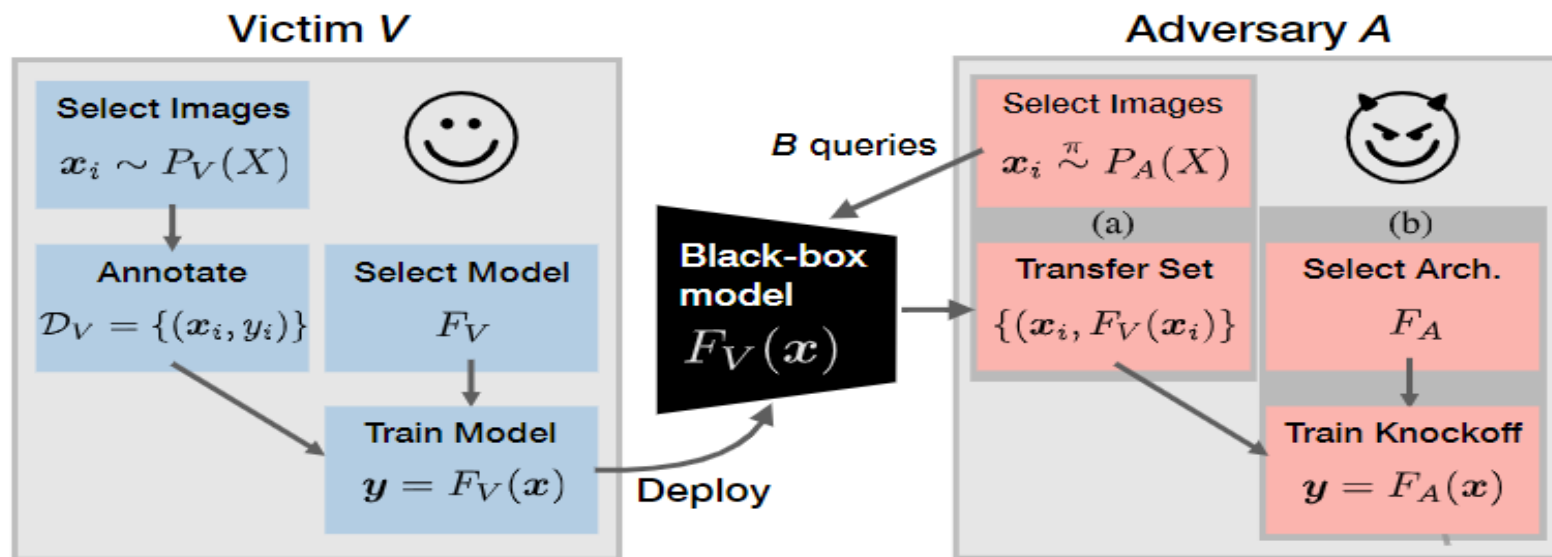


攻击方案示意图

## □ 基于替代模型的模型窃取攻击

■ Victim V: 为了训练模型, 需要

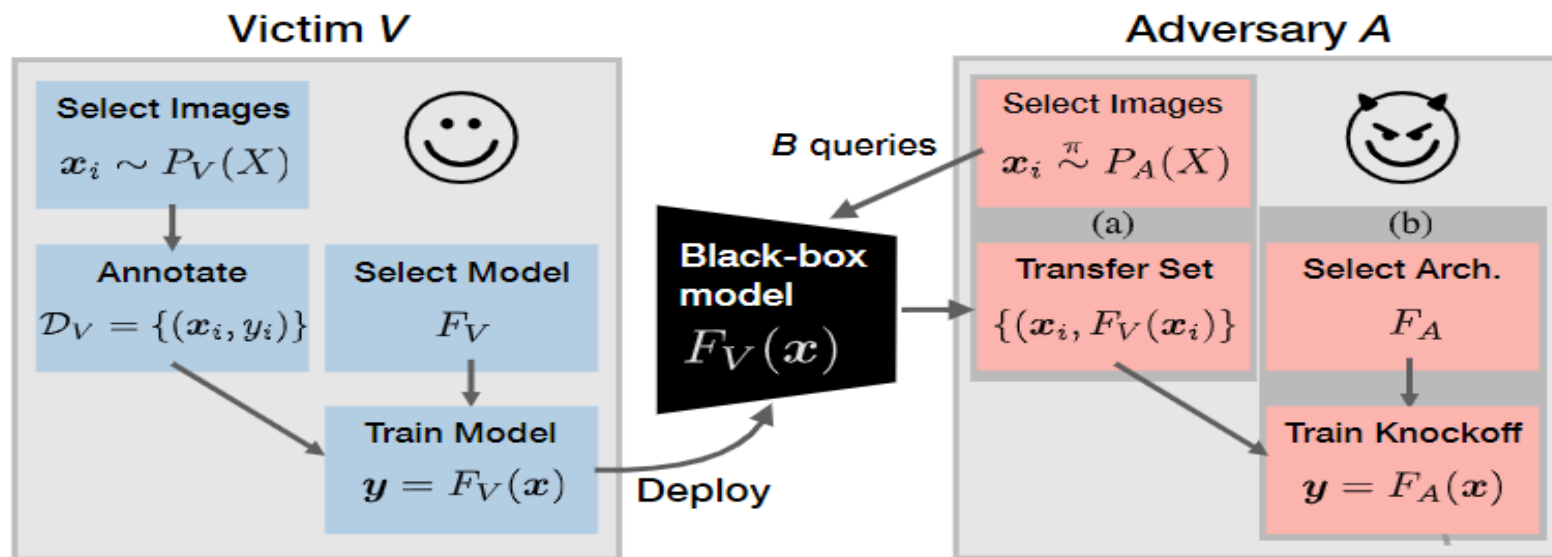
- 收集相关图像  $x_i \sim P_V(X)$ , 并对其打标签, 构成数据集  $\mathcal{D}_V = \{(x_i, y_i)\}$
- 选择在测试集上可以实现最佳性能模型  $F_V$
- 该模型将会以黑盒的形式部署, 输入图像  $x$ , 输出置信度  $y = F_V(x)$



攻击方案示意图

## □ 基于替代模型的模型窃取攻击

- Adversary A: 为了训练一个替代模型来攻击, 需要:
  - 攻击步骤一: 使用策略 $\pi$ 在  $P_A(X)$  分布中采集图像, 经过目标模型识别得到伪标签, 构造出图像的“迁移集”  $\{(x_i, F_V(x_i))\}$ ;
  - 攻击步骤二: 为knockoff选择一个架构并训练其模仿  $F_V$  在迁移集上的行为



攻击方案示意图



## □ 基于替代模型的模型窃取攻击

### ■ 攻击步骤一：构造迁移集

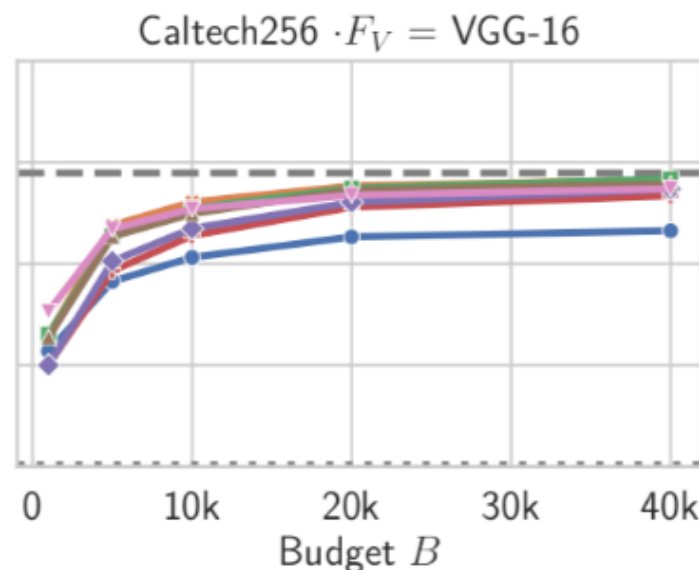
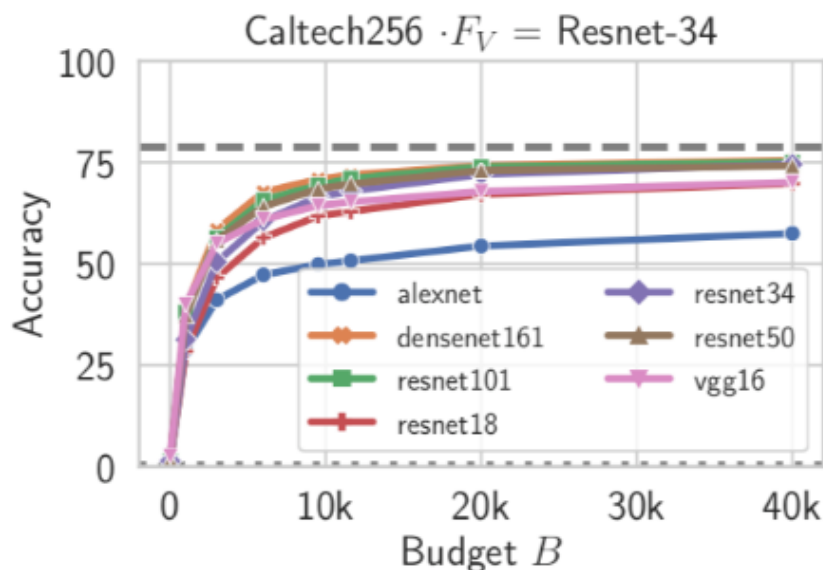
- 首先要选择图像分布  $P_A(X)$  来采样图像，这是一个比较大的离散数据集，例如可以用ImageNet数据集（1.2M张图像）作为  $P_A(X)$
- 然后选择采样策略
  - 随机策略：随机采样图像，极端情况（攻击者采样的图像可能和学习任务无关，比如  $F_V$  是用于分类鸟的，但是采样的图像却是狗）
  - 自适应策略：将每次查询得到的反馈信息集成合并（选择的好给出正反馈不好则给负反馈），学习出一个查询策略

$$\mathbf{x}_t \sim \mathbb{P}_\pi(\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{t-1})$$

## □ 基于替代模型的模型窃取攻击

### ■ 攻击步骤二：替代模型结构的选取

- 由下图可知，knockoff的性能按模型复杂度排序：Alexnet结构最简单，但是性能最差，更复杂的Resnet-101/Densenet-161性能却更好，这说明在选择替代模型架构时，选一个复杂的架构更好
- 从中也能看出，查询次数越多，攻击效果越好



Budget为查询次数

不同替代模型结构攻击结果示意图

## □ 基于替代模型的模型窃取攻击

- 攻击结果：（目标模型和替代模型结构均为Resnet-34）
- $P_A = D$ ：被采样的数据集成了所有用到的数据集（2.2M+2129class），原黑盒模型数据集一定是替代模型采样集  $P_A$  的子集
- 随机策略下，替代模型对原模型功能窃取较为成功，恢复了原模型0.84~0.97的表现
- 采用自适应策略，替代模型识别准确率不仅比随机策略提高了（除了Diabetic5），有些甚至超过原模型

	$P_A$	Caltech256	CUBS200	Indoor67	Diabetic5
$F_V$		78.1(1×)	76.5(1×)	74.9(1×)	58.1(1×)
Random	D	76.6(0.97 ×)	68.3(0.89 ×)	68.3(0.91 ×)	48.9(0.84 ×)
adaptive	D	82.7(1.05 ×)	74.7(0.98 ×)	76.3(1.02 ×)	48.1(0.83 ×)

76.6(0.97 ×)表示在该训练集上识别率为76.6%，实现了原模型97%的功能

## □ 基于替代模型的模型窃取攻击

### ■ 攻击结果:

- 采样分布 $P_A = \text{ImageNet}$ , 受害者模型 $F_V$ 训练数据集和ImageNet有少数重复, 如下:  
Caltech256(42%ImageNet); CUBS200 (1%) ;Indoor67(15%);Diabetic5(0%)
- 这种情况下, 替代模型没见过测试集的大多数图像类别, 随机策略仍然能达到原始黑盒表现的0.82~0.96;
- 自适应策略在Diabetic5数据集上表现不好, 因为对所有图片都没见过, 没有足够的反馈信号, 导致学习出坏的采样策略

	$P_A$	Caltech256	CUBS200	Indoor67	Diabetic5
$F_V$		78.1(1×)	76.5(1×)	74.9(1×)	58.1(1×)
Random	ILSVRC	75.4(0.96 ×)	68.0(0.89 ×)	66.5(0.89 ×)	47.7(0.82 ×)
adaptive	ILSVRC	76.2(0.97 ×)	69.7(0.91 ×)	69.9(0.93 ×)	44.6(0.77 ×)

# 课程大纲

一

隐私保护基本定义

二

机器学习隐私攻击

三

隐私保护机器学习算法

四

各隐私方案对比分析



## □ 隐私威胁的一些防范措施

### ■ 对数据隐私的保护

- 预测阶段的数据隐私保护算法：在模型的预测阶段施加防御手段
- 训练阶段的数据隐私保护算法：在模型的训练阶段施加防御手段

### ■ 对模型版权的保护

- 数字水印 (Digital Watermark)

## □ 对数据隐私的保护

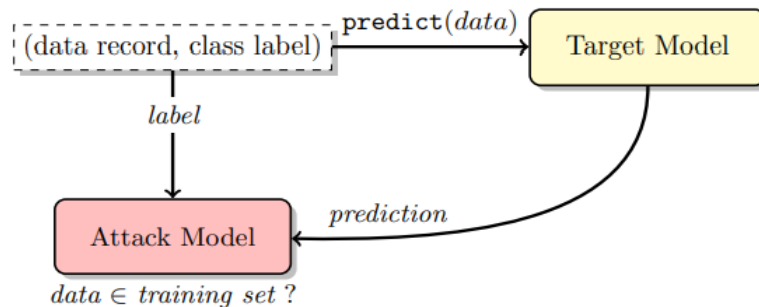
- 案例一：预测阶段的数据隐私保护算法：MemGuard
  - 防御者目的：防御成员推断攻击并使原始模型输出的可用性不下降
  - 防御者能力：
    - 对于原始的分类模型：白盒攻击设置，知道模型结构、参数
    - 对于攻击者的模型：黑盒，既无法知道攻击者的模型结构、参数，也无法访问攻击者模型

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

#### ■ 防御原理

- 成员推断攻击会训练一个判断样本输出的置信度向量是否为训练数据的二分类器，输入置信度向量，输出该样本是否为训练数据
- 可以采用对抗样本的思想，通过向在置信度向量中添加微小的噪音（扰动），使原置信度向量移动到攻击者的分类器的决策边界，从而使攻击者无法从置信度向量分类出该样本是否属于原训练数据





## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤一：训练攻击者模型的替代二分类模型 $g$
- 由于防御者无法知道攻击者所使用的攻击分类器，故为了得到关于攻击者二分类器的对抗样本，防御者自己训练一个成员推断攻击的模型  $g : s \mapsto [0, 1]$  ( $s$ 为 $g$ 的输入：置信度向量)，其中 $g$ 以置信度向量为输入并输出得到该置信度向量的样本为训练数据的概率

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤二：寻找使置信度向量变为对抗样本的最小扰动
  - 和对抗样本相同，我们寻找最小的扰动 $r$ ，使得分类器 $g$ 无法正确分类给定样本是否为原训练数据
  - 但是同时，我们需要满足一些限制条件来保证加噪（即扰动）后的置信度向量仍然有较高的可用性
  - 我们将寻找最小扰动的过程抽象为下式的有约束的优化问题

$$\min_{\mathbf{r}} d(\mathbf{s}, \mathbf{s} + \mathbf{r}) \quad (1)$$

$$\text{subject to: } \operatorname{argmax}_j \{s_j + r_j\} = \operatorname{argmax}_j \{s_j\} \quad (2)$$

$$g(\mathbf{s} + \mathbf{r}) = 0.5 \quad (3)$$

$$s_j + r_j \geq 0, \forall j \quad (4)$$

$$\sum_j r_j = 0, \quad (5)$$

$\mathbf{r}$ : 具体的噪音向量

$g$ : 防御者训练的成员推断攻击的二分类器

$\mathbf{s}$ : 原始置信度向量

$j$ : 向量维度的一维

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤二：寻找使置信度向量变为对抗样本的最小扰动

- 使用 $d(s, s + r)$ 来衡量两个置信度之间的差距，其中 $d(s, s + r) = \|r\|_1$

- (1) 的目的是将添加噪声之后的置信度与原置信度之间的距离最小化

- (2) 的限制使得模型的最终预测标签不发生改变

$$\min_r d(s, s + r) \quad (1)$$

$$\text{subject to: } \operatorname{argmax}_j \{s_j + r_j\} = \operatorname{argmax}_j \{s_j\} \quad (2)$$

$$g(s + r) = 0.5 \quad (3)$$

$$s_j + r_j \geq 0, \forall j \quad (4)$$

$$\sum_j r_j = 0, \quad (5)$$

$r$ : 具体的噪音向量

$g$ : 防御者训练的成员推断攻击的二分类器

$s$ : 原始置信度向量

$j$ : 向量维度的一维

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤二：寻找使置信度向量变为对抗样本的最小扰动
  - (3) 表示攻击者难以从加噪声后的置信度向量分析出该样本是否属于训练数据
  - (4) (5) 保证了加噪声后的置信度向量每一项非负且总和为1，即保证加噪声后的置信度向量仍然为概率分布

$$\min_{\mathbf{r}} d(\mathbf{s}, \mathbf{s} + \mathbf{r}) \quad (1)$$

$$\text{subject to: } \operatorname{argmax}_j \{s_j + r_j\} = \operatorname{argmax}_j \{s_j\} \quad (2)$$

$$g(\mathbf{s} + \mathbf{r}) = 0.5 \quad (3)$$

$$s_j + r_j \geq 0, \forall j \quad (4)$$

$$\sum_j r_j = 0, \quad (5)$$

$\mathbf{r}$ : 具体的噪音向量

$g$ : 防御者训练的成员推断攻击的二分类器

$\mathbf{s}$ : 原始置信度向量

$j$ : 向量维度的一维

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤二：寻找使置信度向量变为对抗样本的最小扰动
  - 显然同时优化所有的约束条件是困难的，所以我们将这个问题的约束条件进行简化
  - 约束 (4) (5) 是为了保证添加扰动后的置信度向量仍然为概率分布
  - 使用 $z$ 来表示原始分类器 $\text{softmax}$ 函数的输入，我们在 $z$ 上添加扰动 $e$ 来代替在最后的 $\text{softmax}$ 函数上添加扰动
  - 将 $z + e$ 通过 $\text{softmax}$ 函数得到 $s + r$ ，就可以保证添加扰动后的置信度向量仍然为概率分布

$$s_j + r_j \geq 0, \forall j \quad (4) \quad s + r = \text{softmax}(z + e)$$

$$\sum_j r_j = 0, \quad (5) \quad (6)$$

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤二：寻找使置信度向量变为对抗样本的最小扰动

➤ 通过上述的简化，我们将约束 (4) (5) 消去，并将原问题转化为 (7) (8) (9)

$$\begin{aligned} s_j + r_j &\geq 0, \forall j & (4) & \quad \mathbf{s} + \mathbf{r} = \text{softmax}(\mathbf{z} + \mathbf{e}) & \quad \min_{\mathbf{e}} d(\text{softmax}(\mathbf{z}), \text{softmax}(\mathbf{z} + \mathbf{e})) & (7) \\ \sum_j r_j &= 0, & (5) & \quad & \quad \text{subject to: } \underset{j}{\operatorname{argmax}}\{z_j + e_j\} = \underset{j}{\operatorname{argmax}}\{z_j\} & (8) \\ & & (6) & \quad & \quad g(\text{softmax}(\mathbf{z} + \mathbf{e})) = 0.5. & (9) \end{aligned}$$

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤二：寻找使置信度向量变为对抗样本的最小扰动

#### ➤ 简化约束函数 (9)

- 由于 $g$ 为判断置信度所属样本是否为原训练数据的二分类器，我们可以设置 $g$ 的最终激活函数为 $\text{sigmod}$ 函数，令 $h$ 为 $g$ 中通过 $\text{sigmod}$ 以前的等效函数，那么我们可以将 (9) 看成是 (10) = 0.5
- 显然，当 (10) = 0.5时， $h(\text{softmax}(z+e)) = 0$ ，我们将其转化为目标函数 (11)，此时 $L_1$ 越小，越接近我们的目标（ $g$ 无法判断准确给定样本是否为原训练数据）

$$\begin{aligned} g(\text{softmax}(z+e)) &= 0.5. & g(\text{softmax}(z+e)) &= \frac{1}{1 + \exp(-h(\text{softmax}(z+e)))} & L_1 &= |h(\text{softmax}(z+e))| \\ (9) & & (10) & & (11) \end{aligned}$$

$g$ ：防御者训练的成员推断攻击的二分类器

$h$ ： $g$ 中通过 $\text{sigmod}$ 以前的等效函数

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤二：寻找使置信度向量变为对抗样本的最小扰动

➤ 简化约束函数 (8)

➤ 由于 $\text{softmax}$ 函数并不会改变每个输出元素的相对大小，故我们将 (8) 转化为 (12)

$$\operatorname{argmax}_j \{z_j + e_j\} = \operatorname{argmax}_j \{z_j\}$$

(8)

$$L_2 = \operatorname{ReLU}(-z_l - e_l + \max_{j|j \neq l} \{z_j + e_j\})$$

(12)

ReLU：激活函数，其中 $\operatorname{ReLU}(v) = \max(0, v)$       $l$ ：样本经过原始分类器最终得到的标签值



## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤二：寻找使置信度向量变为对抗样本的最小扰动

➤ 自此，我们将原来的有约束的问题转化为了一个无约束的优化问题，目标函数为 (13)

➤  $L_3$  (即式 14) 衡量加噪后的差距大小，即表示 (7) 的目标函数，其中

$$d(\text{softmax}(z), \text{softmax}(z+e)) = \|( \text{softmax}(z+e) - \text{softmax}(z) ) \|_1$$

$$\min_e L = L_1 + c_2 \cdot L_2 + c_3 \cdot L_3$$

(13)

$$L_3 = d(\text{softmax}(z), \text{softmax}(z+e))$$

(14)

$c_2, c_3$ : 平衡三个目标函数的超参数

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤二：寻找使置信度向量变为对抗样本的最小扰动
  - 使用梯度下降的算法来解决这个无约束的优化问题
  - 由于添加的扰动是微小的，所以通过增大 $c_3$ 来寻找合适的解

#### Algorithm 1 Phase I of MemGuard

**Input:**  $\mathbf{z}$ ,  $max\_iter$ ,  $c_2$ ,  $c_3$ , and  $\beta$  (learning rate).

**Output:**  $\mathbf{e}$

```
1: //Predicted label
2:  $l = \operatorname{argmax}_j \{z_j\}$ 
3: while True do
4:   //A new iteration to search  $c_3$ 
5:    $\mathbf{e} = \mathbf{0}$ 
6:    $\mathbf{e}' = \mathbf{e}$ 
7:    $i = 1$ 
8:   while  $i < max\_iter$  and  $(\operatorname{argmax}_j \{z_j + e_j\} \neq l$  or  $h(\operatorname{softmax}(\mathbf{z})) \cdot h(\operatorname{softmax}(\mathbf{z} + \mathbf{e})) > 0)$  do
9:     //Gradient descent with normalized gradient
10:     $\mathbf{u} = \frac{\partial L}{\partial \mathbf{e}}$ 
11:     $\mathbf{u} = \mathbf{u} / \|\mathbf{u}\|_2$ 
12:     $\mathbf{e} = \mathbf{e} - \beta \cdot \mathbf{u}$ 
13:     $i = i + 1$ 
14:   end while
15:   //Return the vector in the previous iteration if the predicted label changes or the sign of  $h$  does not change in the current iteration
16:   if  $\operatorname{argmax}_j \{z_j + e_j\} \neq l$  or  $h(\operatorname{softmax}(\mathbf{z})) \cdot h(\operatorname{softmax}(\mathbf{z} + \mathbf{e})) > 0$  then
17:     return  $\mathbf{e}'$ 
18:   end if
19:    $c_3 = 10 \cdot c_3$ 
20: end while
```

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤二：寻找使置信度向量变为对抗样本的最小扰动
- 对于每轮 $c_3$ ，我们在有限的迭代次数内查看当前所添加的扰动是否满足限制 (8) (9)，若不满足则进行梯度下降算法优化
- 我们使用  $h(\text{softmax}(z)) \cdot h(\text{softmax}(z+e)) \leq 0$  来近似  $h(\text{softmax}(z+e)) = 0$ ；由于我们选择的学习率很小，所以当两者符号开始不一致时，我们认为 $h(\text{softmax}(z+e))$  在0附近

$$\underset{j}{\operatorname{argmax}}\{z_j + e_j\} = \underset{j}{\operatorname{argmax}}\{z_j\}$$

(8)

$$g(\text{softmax}(z + \mathbf{e})) = 0.5.$$

(9)

### Algorithm 1 Phase I of MemGuard

**Input:**  $z$ ,  $\max\_iter$ ,  $c_2$ ,  $c_3$ , and  $\beta$  (learning rate).

**Output:**  $\mathbf{e}$

```
1: //Predicted label
2:  $l = \operatorname{argmax}_j\{z_j\}$ 
3: while True do
4:   //A new iteration to search  $c_3$ 
5:    $\mathbf{e} = \mathbf{0}$ 
6:    $\mathbf{e}' = \mathbf{e}$ 
7:    $i = 1$ 
8:   while  $i < \max\_iter$  and  $(\operatorname{argmax}_j\{z_j + e_j\} \neq l$  or  $h(\text{softmax}(z)) \cdot h(\text{softmax}(z + \mathbf{e})) > 0)$  do
9:     //Gradient descent with normalized gradient
10:     $\mathbf{u} = \frac{\partial L}{\partial \mathbf{e}}$ 
11:     $\mathbf{u} = \mathbf{u} / \|\mathbf{u}\|_2$ 
12:     $\mathbf{e} = \mathbf{e} - \beta \cdot \mathbf{u}$ 
13:     $i = i + 1$ 
14:   end while
15:   //Return the vector in the previous iteration if the predicted label changes or the sign of  $h$  does not change in the current iteration
16:   if  $\operatorname{argmax}_j\{z_j + e_j\} \neq l$  or  $h(\text{softmax}(z)) \cdot h(\text{softmax}(z + \mathbf{e})) > 0$  then
17:     return  $\mathbf{e}'$ 
18:   end if
19:    $c_3 = 10 \cdot c_3$ 
20: end while
```

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

#### • 防御步骤三：添加扰动（噪声）

- 我们以一定的概率 $p$ 选择添加噪声来使得添加噪声后替代模型 $g$ 输出的样本为训练数据的概率接近0.5（即式15）并使其满足（16）的对于噪声大小期望的限制，那么我们可以解得 $p$ 为式（17）
- 显然如果攻击者知道我们的防御策略，并且知道 $\epsilon$ 的话，攻击者可以很容易地通过多次询问相同的图片来得到两个不同的置信度向量 $s_1, s_2$ ，并且可以通过（17）计算出 $p$ 的值，通过 $p$ 和攻击者采集到的 $s_1, s_2$ 出现的次数推测出真实的置信度向量

$$p = \underset{p}{\operatorname{argmin}} |p \cdot g(s + r) + (1 - p) \cdot g(s + 0) - 0.5| \quad (15)$$

$$\text{subject to: } p \cdot d(s, s + r) + (1 - p) \cdot d(s, s + 0) \leq \epsilon \quad (16)$$

$$p = \begin{cases} 0, & \text{if } |g(s) - 0.5| \leq |g(s + r) - 0.5| \\ \min(\frac{\epsilon}{d(s, s+r)}, 1.0), & \text{otherwise.} \end{cases} \quad (17)$$

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

- 防御步骤三：添加扰动（噪声）

采用哈希的思想，将每张图片量化为一个**哈希值**，使用图片被量化以后的哈希值作为伪随机数生成的 $seed$ 来生成 $[0,1]$ 之间的随机数 $p'$ ，当 $p' < p$ 时，在输出上加入噪声，反之不加噪声。由于输入的图片相同时，我们会得到相同的随机数 $p'$ ，就可以保证相同图片的输出相同

$$p = \underset{p}{\operatorname{argmin}} |p \cdot g(\mathbf{s} + \mathbf{r}) + (1 - p) \cdot g(\mathbf{s} + \mathbf{0}) - 0.5| \quad (15)$$

$$\text{subject to: } p \cdot d(\mathbf{s}, \mathbf{s} + \mathbf{r}) + (1 - p) \cdot d(\mathbf{s}, \mathbf{s} + \mathbf{0}) \leq \epsilon \quad (16)$$

$$p = \begin{cases} 0, & \text{if } |g(\mathbf{s}) - 0.5| \leq |g(\mathbf{s} + \mathbf{r}) - 0.5| \\ \min(\frac{\epsilon}{d(\mathbf{s}, \mathbf{s} + \mathbf{r})}, 1.0), & \text{otherwise.} \end{cases} \quad (17)$$

## □ 对数据隐私的保护

### ■ 案例一：预测阶段的数据隐私保护算法：MemGuard

#### • 实验结果

- 图为不同的成员推断攻击方法在三个不同数据集上，当对于噪声大小的限制增大时，攻击成功率的变化
- 当限制为0时，即不添加任何噪声
- 如图可知随着可以添加噪声的上限增加时，各种攻击的成功率逐步下降，并最终基本等于50%（即每次随机猜测的成功率）

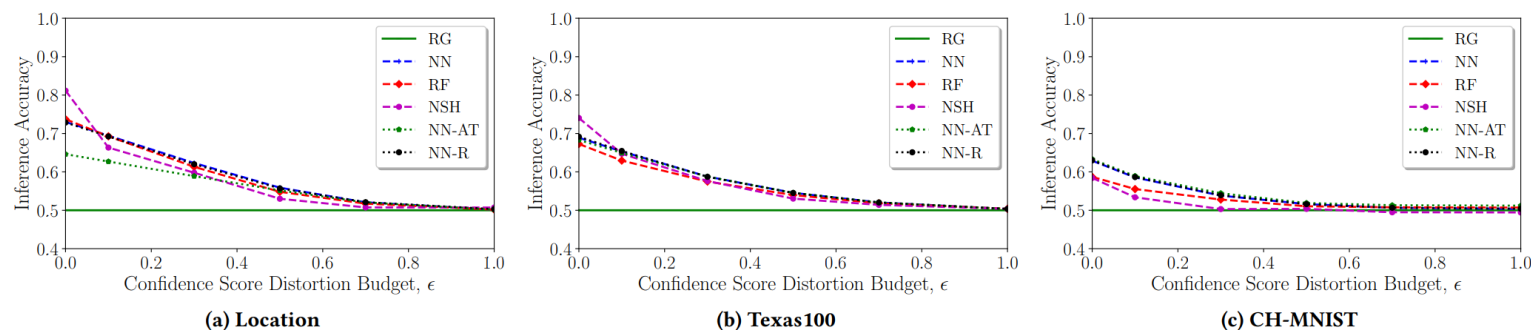


Figure 1: Inference accuracies of different attacks as the confidence score distortion budget (i.e.,  $\epsilon$ ) increases.

## □ 对数据隐私的保护

### ■ 案例二：训练阶段的数据隐私保护算法：DP-SGD

- 防御者目的：保护模型训练时训练数据的隐私，即模型保留的训练数据的信息

- 防御者能力：

- 对于原始的分类模型：白盒攻击设置，知道模型结构、参数，且参与到原始分类模型的训练阶段中去

- 对于攻击者的模型：黑盒，既无法知道攻击者的模型结构、参数，也无法访问攻击者模型

## □ 对数据隐私的保护

### ■ 案例二：训练阶段的数据隐私保护算法：DP-SGD

- 防御原理
  - 模型训练时对每个训练数据执行SGD时计算的梯度决定了模型保存训练数据的信息量
  - 训练数据在模型中信息是通过训练阶段的计算出的每个样本的梯度来体现的
  - 可以在模型训练的计算梯度的过程中添加噪声来保护训练数据的隐私



## □ 对数据隐私的保护

### ■ 案例二：训练阶段的数据隐私保护算法：DP-SGD

- 松弛差分隐私

- 我们在纯差分隐私 $\Pr(M(D') \in S) \leq e^\epsilon \Pr(M(D) \in S)$ 上引入一个极小的松弛项 $\delta$ ，来表示我们可以接受在一个很小的程度上不满足差分隐私

$$\Pr(M(D') \in S) \leq e^\epsilon \Pr(M(D) \in S) + \delta$$

$D, D'$ ：相邻数据

$e^\epsilon$ ：设定的相邻数据输出概率的差异上限

$M$ ：添加噪声的函数

$\delta$ ：衡量与纯差分隐私的差距的参数

## □ 对数据隐私的保护

### ■ 案例二：训练阶段的数据隐私保护算法：DP-SGD

#### • 高斯噪声机制

- 高斯噪声机制即将输出添加符合高斯分布的噪声

- 其中添加的噪声符合均值为0，方差为  $\frac{2s^2 \log(1.25/\delta)}{\epsilon^2}$

$$M(D) = f(D) + N(\sigma^2)$$
$$\text{where } \sigma^2 = \frac{2s^2 \log(1.25/\delta)}{\epsilon^2}$$

$D, D'$ : 相邻数据

$f$ : 原始查询函数

$M$ : 添加噪声的函数

$s$ : 函数 $f$ 的敏感度

## □ 对数据隐私的保护

### ■ 案例二：训练阶段的数据隐私保护算法：DP-SGD

- 防御步骤一：选择训练批次大小并由原分类器的损失函数计算出原始的梯度
  - 在总数据集 $N$ 中选择大小为 $L_t$ 的批次，并计算每个样本关于原分类器的损失函数产生的梯度 $g_t(x_i)$

$$g_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, x_i)$$

$g_t(x_i)$ :  $t$ 时样本 $x_i$ 的梯度

$\theta$ : 模型参数

$L$ : 损失函数

## □ 对数据隐私的保护

### ■ 案例二：训练阶段的数据隐私保护算法：DP-SGD

#### • 防御步骤二：梯度裁剪

- 我们希望每个数据对于模型的贡献不会超过一定值，过大的贡献会导致模型保留下关于样本的信息过多，这会可能会导致隐私问题
- 由于没有每个训练数据对于模型影响的先验知识，我们将每个数据产生的梯度裁剪到 $C$
- 我们以梯度的二范数作为衡量的标准，样本产生梯度的二范数大于 $C$ 的，我们将样本产生的梯度的二范数裁剪到 $C$ ，反之样本产生梯度的二范数小于等于 $C$ 的，我们不改变其梯度

Clip gradient

$$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$$

## □ 对数据隐私的保护

### ■ 案例二：训练阶段的数据隐私保护算法：DP-SGD

- 防御步骤三：添加噪声并进行参数的更新
  - 对于每个样本产生的梯度，我们采用差分隐私中的高斯机制来在梯度中加入噪声，并进行参数的更新
  - 由于我们将梯度裁剪到 $C$ ，所以两个梯度的差距不会超过 $C$ ，可以认为两个数据产生的梯度值差异不大于 $C$

$$\tilde{g}_t \leftarrow \frac{1}{L} \sum_i (\bar{g}_t(x_i) + N(0, \sigma^2 C^2))$$

**Descent**

$$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{g}_t$$

## □ 对数据隐私的保护

### ■ 案例二：训练阶段的数据隐私保护算法：DP-SGD

#### • 实验结果：

- 图为经过DP-SGD的方法训练后测定的所满足的差分隐私的预算 $\epsilon$
- 其中预先设定 $q = L_t/N = 0.01$  ( $L_t$ 为训练时的批次大小,  $N$ 为训练数据总量), 高斯噪声的超参数 $\sigma = 4$ , 且允许打破预算的概率为 $\delta = 1e - 5$
- 可以看到DP-SGD的方法在moments accountant (一种关于机器学习模型的差分隐私预算计算方式) 的隐私预算计算下, 当epoch=400时, DP-SGD可以满足(2.55,1e-5)-差分隐私

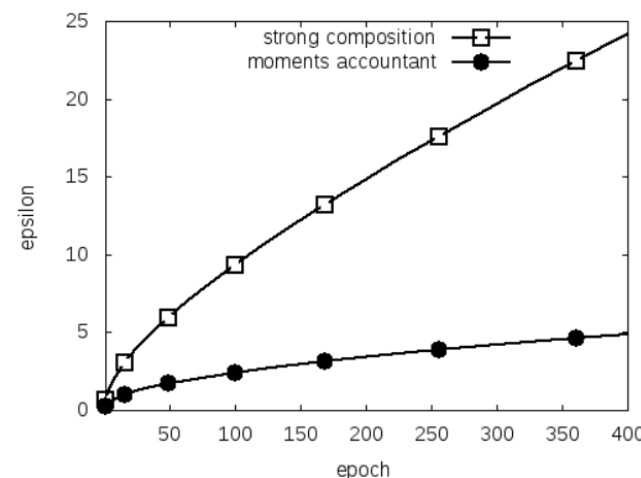


Figure 2: The  $\epsilon$  value as a function of epoch  $E$  for  $q = 0.01$ ,  $\sigma = 4$ ,  $\delta = 10^{-5}$ , using the strong composition theorem and the moments accountant respectively.

## □ 对模型隐私的保护

### ■ 案例三：数字水印

- 防御者目的：保护模型版权，在模型被窃取后可以在被窃取的模型中恢复出嵌入的水印
- 防御者能力：
  - 对于原始的分类模型：白盒攻击设置，知道模型结构、参数，且参与到原始分类模型的训练阶段中去
  - 对于被窃取的模型：白盒，可以从中提取水印

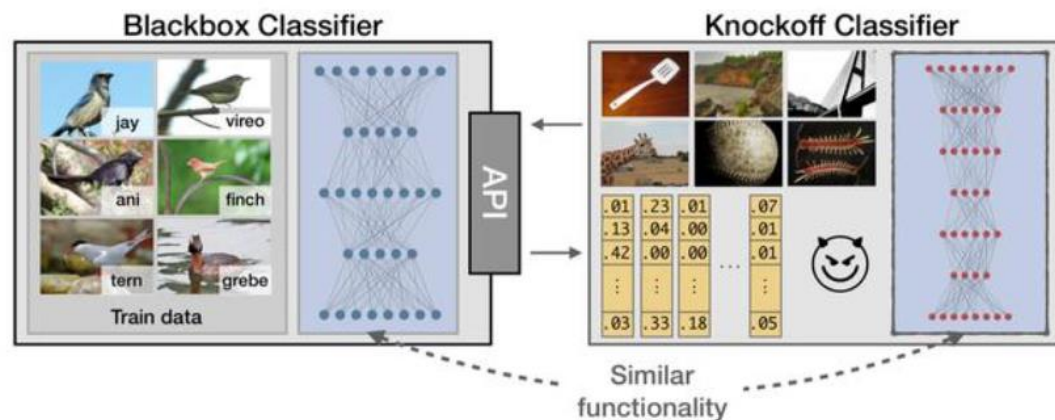


## □ 对模型隐私的保护

### ■ 案例三：数字水印

#### • 数字水印的防御场景

- 即使只是提供黑盒的访问权限，攻击者还是可以从恢复出模型的**基本信息**（模型窃取攻击，包括恢复出原始模型的完整结构与每一层的参数），从而造成严重的模型版权问题
- 数字水印可以在模型被窃取时验证模型的版权，可以作为有效的防御模型窃取的方法



## □ 对模型隐私的保护

### ■ 案例三：数字水印

#### • 数字水印概述

- 数字水印，是指将**特定的信息**（一般为特定长度的0或1的比特流）嵌入数字信号中，数字信号可能是音频、图片、模型或是视频等
- **若要拷贝有数字水印的信号，所嵌入的信息也会一并被拷贝**
- 在神经网络模型训练时注入数字水印，并在模型训练完后可以从模型中提取出注入的数字水印来验证模型的版权

## □ 对模型隐私的保护

### ■ 案例三：数字水印

- 嵌入的目标水印：水印嵌入的目标是嵌入 $T$ 位的水印  $b \in \{0, 1\}^T$ ， $b$ 的每一位为0或1
- 水印的目标嵌入环境：
  - 我们将数字水印嵌入神经网络中的某一层参数中
  - 假设我们将水印嵌入到的卷积层参数形状为 $S*S*D*L$ ，其中 $(S*S)$ 为卷积核大小， $D$ 为传入的通道数， $L$ 为输出的通道数，故这个卷积层的参数总量为 $S*S*D*L$
  - 将参数根据输出的维度取均值消去这个维度  $\overline{W}_{ijk} = \frac{1}{L} \sum_l W_{ijkl}$ ，得到平均后的参数  $w \in \mathbb{R}^M$  ( $M = S \times S \times D$ )

## □ 对模型隐私的保护

### ■ 案例三：数字水印

#### ■ 水印提取方式：

- 我们使用水印提取矩阵  $X \in R^{T * M} (M = S * S * D)$  来提取水印，通过将  $X$  与  $w$  相乘得到值  $X'$ ，
- 然后我们使用激活函数  $y_i = \text{sigmoid}(X_i')$  来进一步提取水印
- 最后根据  $y_i$  来最终提取水印：当  $y_i$  大于我们设定的阈值（超参数）时我们将这一位（即第  $i$  位）赋值为1，反之赋值为0

$$b_j = s(\sum_i X_{ji} w_i)$$

$X$ : 提取水印的辅助超参数

$s$ : sigmoid函数 + 赋值[0,1]的函数

## □ 对模型隐私的保护

### ■ 案例三：数字水印

- 防御主要步骤：设定模型训练的损失函数进行模型训练
  - 我们将前文提到的**水印提取的损失**作为一个**正则项**加入到原来的神经网络的损失函数中，这样我们就可以在神经网络训练时嵌入水印
  - 我们使用从神经网络中**提取的水印与真实水印**的*cross entropy*损失来表示水印嵌入的损失函数，并将其加入到原损失函数中去，这样在网络训练阶段就可以嵌入水印

$$E(w) = E_0(w) + \lambda E_R(w)$$

$E_0(w)$ : 原始网络的损失函数

$E_R(w)$ : 水印嵌入的损失函数

$\lambda$ : 平衡两个损失函数的超参数

$$E_R(w) = - \sum_{j=1}^T (b_j \log(y_j) + (1 - b_j) \log(1 - y_j)) \quad y_j = \sigma(\sum_i X_{ji} w_i) \quad \sigma(x) = \frac{1}{1 + \exp(-x)}$$

$b$ : 希望嵌入的目标水印

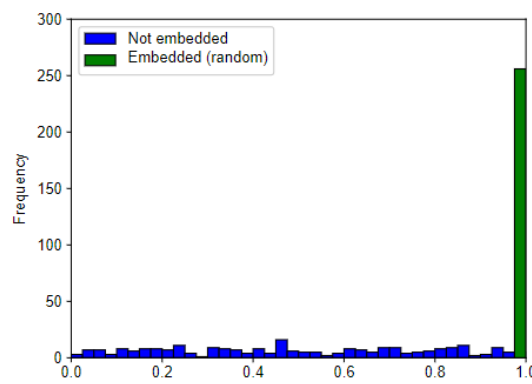
$y$ : 提取出经过最终赋值[0,1]前的水印值

## □ 对模型隐私的保护

### ■ 案例三：数字水印

#### • 实验结果：

- 此图为在提取水印时的  $\sigma(\sum_i X_{ji} w_i)$  的值出现的频率（ $\sigma$ 代表 $sigmoid$ 函数），其中嵌入的水印  $b = \{1\}^T$ ，提取的超参数为高斯分布  $N(0,1)$  随机采样，蓝色为未嵌入水印的情况，而绿色为嵌入水印的情况
- 可以发现嵌入水印后的值集中出现在1.0附近，故水印提取阈值取在  $[0,1]$  中的任何数时我们都可以完整地将水印提取出来

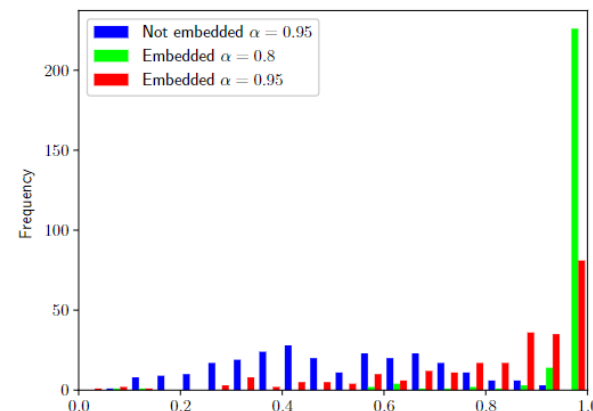


(c) random

## □ 对模型隐私的保护

### ■ 案例三：数字水印

- 实验结果：鲁棒性测试
  - 模型窃取攻击会将模型进行裁剪，所以模型裁剪后水印的保留率至关重要
  - 此图为在不同的裁剪度 $\alpha$ 下的 $\sigma(\sum_i X_{ji} w_i)$ 值出现的频率，其中嵌入的水印 $b = \{1\}^T$ ，水印提取矩阵为高斯分布 $N(0,1)$ 随机采样，蓝色为未嵌入水印的情况，而绿色和红色为嵌入水印的情况
  - 可以发现，即使在裁剪率80%的情况下（绿色矩形）还是可以在水印提取阈值选择0.5的情况下完全正确地提取出水印且与未嵌入水印的模型有着明显的区分



# 课程大纲

一

隐私保护基本定义

二

机器学习隐私攻击

三

隐私保护机器学习算法

四

各隐私方案对比分析





# 机器学习中的隐私威胁回顾

分类	威胁形式	发生阶段	敌手能力	敌手知识
针对数据	模型逆向攻击	预测阶段	访问目标模型	黑盒/白盒
	成员推断攻击	预测阶段	访问目标模型	黑盒
针对模型	模型窃取攻击	预测阶段	访问目标模型	黑盒

# 机器学习中的隐私保护思想对比

隐私保护思想	原理	实例	保护主体
添加随机扰动	对数据添加噪声, 具有随机性	差分隐私, MemGuard	训练数据/预测数据
嵌入水印	在模型训练中加入水印来保证版权	数字水印	模型版权

# 针对各种隐私威胁的防范措施

威胁形式	防范措施	防范对象
模型逆向攻击	输出模糊	不良用户
成员推理攻击	差分隐私, MemGuard	不良用户
模型窃取攻击	在模型中添加象征版权的数字水印	不良用户、不良参与方

# 各隐私方案对比分析

## □ 隐私方案的时间成本分析

- 基于训练阶段的隐私方案如DP-SGD和数字水印的时间成本体现在模型训练的阶段，通过修改模型的损失函数和梯度来添加防御措施
- 基于训练阶段的隐私方案对于用户来说在时间上的区分并不大，但是由于在训练阶段增加了隐私防御的目标，通常会导致模型收敛慢，训练准确率下降等问题
- 基于预测阶段的隐私方案如MemGuard的时间成本体现在模型预测的阶段，通过修改模型的置信度向量来添加防御措施
- MemGuard对于用户的影响较大，每次需要较长的时间来寻找置信度向量的对抗样本

# 讨论与思考

---

- ✓ 白盒模型逆向攻击和黑盒模型反演攻击各适用于什么样的场景？课程中的模型逆向攻击都需要最终的输出向量，是否有可能进行仅获得标签的模型逆向攻击？
- ✓ 模型窃取攻击中，替代模型方法异常的大量查询不仅仅会增加窃取成本，更会被模型拥有者检测出来，你能想到什么解决方法来避免过多的向目标模型查询？
- ✓ 在MemGuard的防御场景下，如果攻击者在输入图像上添加扰动可以破坏单次随机的设定，你认为防御者应该如何应对？
- ✓ 数字水印可以保护模型版权，但是无法防御攻击者窃取模型的过程，是否有方法可以直接防止模型被窃取？

# 谢 谢

**浙江大学网络空间安全学院**

**<https://icsr.zju.edu.cn/>**

