

**毕 业 设 计**(**论文**)

题 目

深度学习中的生成对抗网络攻击设计与实现

|  |  |
| --- | --- |
| 院 系 | 控制与计算机工程学院计算机系 |
| 专业班级 | 计算机科学与技术专业1401班 |
| 学生姓名 | 孙昱伟 |
| 指导教师 | 王洪涛 |

**二○一八年六月**

深度学习中的生成对抗网络攻击设计与实现

摘要

近年来，深度神经网络由于可以自动学习特征，在诸多领域受到了广泛的关注。比起其他机器学习算法，它大大提高了这些领域的表达能力。尤其是卷积神经网络和循环神经网络被应用于图像识别，自然语言处理等领域，为人们的生活带来了极大的便利。在当前大数据的时代，利用庞大的数据集，在服务器上进行机器学习的研究的方式已经非常普遍，而对这些用于机器学习而上传的个人用户数据，其安全性仍亟待考虑。现阶段就学习模型，主要有两种，集中式学习模型与分布式协同学习模型。一方面，集中式学习模型运行在服务器端，用户将数据上传到服务器端进行学习，容易暴露用户的隐私，另一方面，每个用户拥有的数据比较有限，容易出现过拟合的问题。在此基础上有学者提出了分布协同式的学习模型。用户可以在本地进行模型训练，而不用将数据上传给服务器，通过共享训练后的参数集而不是数据本身来达到协同训练的目的，同时也解决了用户训练数据太少过拟合的问题。

但是看似安全的这种学习方式其实蕴藏了极大的风险。共享的训练后的参数集中包含了用户原始数据的特征，通过对特征中隐藏的信息的提取与还原，有可能会盗取到原始训练时的用户数据。为了验证这一猜想，本文采用一种用生成对抗网络模型（GAN）去模拟攻击在这种深度学习用场景下的用户，诱使受害者泄露更多的个人信息的模型。通过利用GAN的这种攻击手法，对于协同型学习场景中的每个用户来说，他们都有可能成为攻击者来获取其他用户的个人信息。本文旨在为协同型学习的安全性问题的研究提供依据。

关键词: 深度学习、生成对抗网络、协同型深度学习、信息安全、机器学习

AN ATTACK ON DEEP LEARNING SYSTEM BASED ON GENERATIVE ADVERSIAL NETWORK

Abstract

In recent years, because that Deep Learning algorithm can automatically learn the features from the training set,it has received extensive attention in many fields. Compared to the machine learning algorithms so far, it has greatly improved the expressiveness in these fields. Especially the convolutional neural network and the recurrent neural network, these two neural network algorithms have been widely used. In this era of big data, research on machine learning on servers has become very common, and there are a large number of scholars working on it, and how is the security of personal user data uploaded for machine learning? Urgently to be considered. There are two main learning models at the present stage: the centralized learning model and the distributed collaborative learning model. On the one hand, the centralized learning model runs on the server side. Users upload data to the server for learning, which is easy to expose user's privacy. On the other hand, each user has limited data and is prone to overfitting. On this basis, some scholars have proposed a distributed collaborative learning model. The user can perform model training locally instead of uploading data to the server and achieve the purpose of collaborative training by sharing the parameter set after training instead of the original data, and also solves the problem that the user training data is too little to fit.

However, this seemingly safe way of learning actually contains a great deal of risk. The shared parameter set after training contains the characteristics of the user's original data. By extracting and restoring the hidden information in the feature, the user data in the original training may be stolen. In order to verify this conjecture, this paper proposes a method to use GAN to simulate attacks on users in this deep learning scenario, and even induces victims to reveal more personal information. By using GAN's attack techniques, For each user in the learning environment, they may all be attackers to obtain other users' personal information. This paper aims to provide a basis for the study of the security of collaborative learning.

Keywords: Deep learning、Generative Adversarial Network、Collaborative deep learning、Information security、Machine learning

目录

摘要 ……………………………………………………………………………………………Ⅰ

Abstract…………………………………………………………………………………………Ⅱ

1 绪论……………………………………………………………………………………………1

1.1 课题背景……………………………………………………………………………1

1.2 研究现状……………………………………………………………………………1

1.2.1 协同深度学习 …………………………………………………………1

1.2.2 深度学习中的隐私泄露 ………………………………………………2

1.3 本文的主要研究内容………………………………………………………………3

2 协同深度学习系统设计………………………………………………………………………4

2.1 深度学习原理………………………………………………………………………4

2.2 协同深度学习架构…………………………………………………………………6

2.3 协同深度学习算法实现……………………………………………………………7

3 基于生成对抗网络的隐私攻击………………………………………………………………9

3.1 生成对抗网络原理…………………………………………………………………9

3.2 隐私攻击框架设计…………………………………………………………………11

3.3 隐私攻击算法与实现………………………………………………………………11

3.4 结果分析……………………………………………………………………………16

研究意义与结论…………………………………………………………………………………17

参考文献…………………………………………………………………………………………18

致谢………………………………………………………………………………………………19

1绪论

1.1 课题的研究背景

深度学习是机器学习的一个新分支，它利用神经网络，这个概念可追溯到1943年，用来为各种复杂任务找到解决方案。神经网络受到人类大脑学习方式的启发，表明分布式人工神经网络也可以学习非平凡的任务，即使当前的架构和学习过程远没有大脑般的行为。

但是，深度学习会带来严重的隐私问题，因为训练后的模型包含有关训练集的基本信息。从模型中提取敏感信息相对比较简单。深度学习社区最近提出了生成对抗网络（GANs）这些网络仍在深入研究中。 GAN的目标不是将图像分为不同的类别，而是生成类似外观的样本（理想情况下具有相同的分布）。更重要的是，GAN生成这些样本而无需访问原始样本。 GAN只要与判别器端的深度神经网络相互作用，就可了解数据的分布。攻击的结果是任何作为内部人员的用户都可以从受害者的设备中推断出敏感信息。攻击者只需运行协同学习算法，便可重构存储在受害者设备上的敏感信息。攻击者还能够影响学习过程并欺骗受害者泄露更详细的信息。在这种情况下，任何用户都可以故意危害任何其他用户，使协同式学习网络更加不可取。

1.2 研究现状

1.2.1 协同深度学习

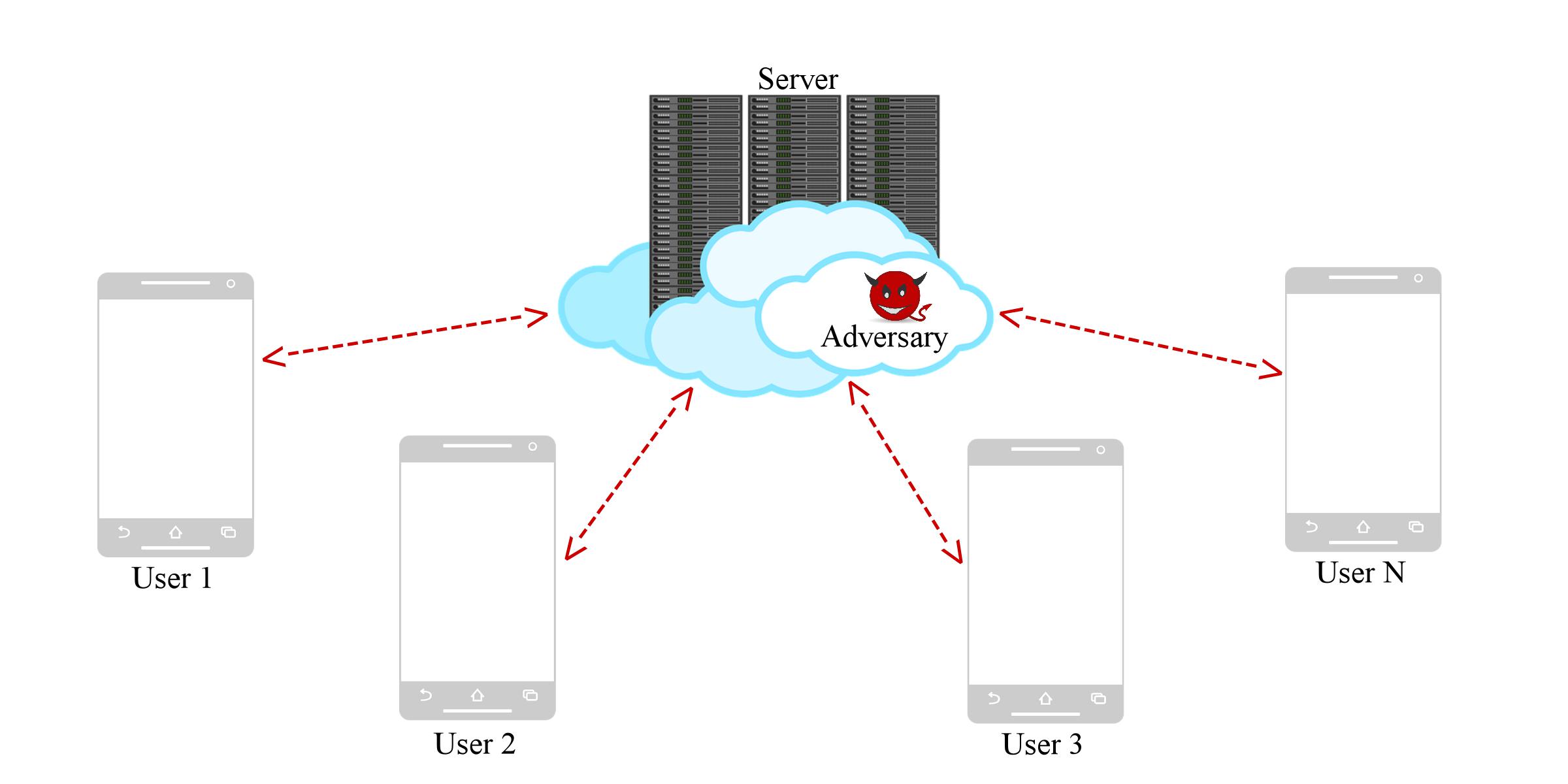
 深度学习模型在相关数据方面表现非常好，这对计算机视觉，图像处理，视频处理，人脸识别，语音识别。但是，深度学习会带来严重的隐私问题，因为训练后的模型包含有关训练集的基本信息。从模型中提取敏感信息相对比较简单。考虑图1所示的以下情况，其中N个用户在其各自的设备上存储私人信息的本地数据集，并且希望合作构建一个通用的区分机器。如图1.1所示，我们可以通过将所有数据集上传到单个位置（例如云）来构建分类器。服务操作员在组合的数据集上训练模型。这个集中式方法非常有效，因为该模型可以访问所有数据，但由于运营商可以直接访问敏感信息，因此它不会保护隐私。

图1.1 集中式学习模型

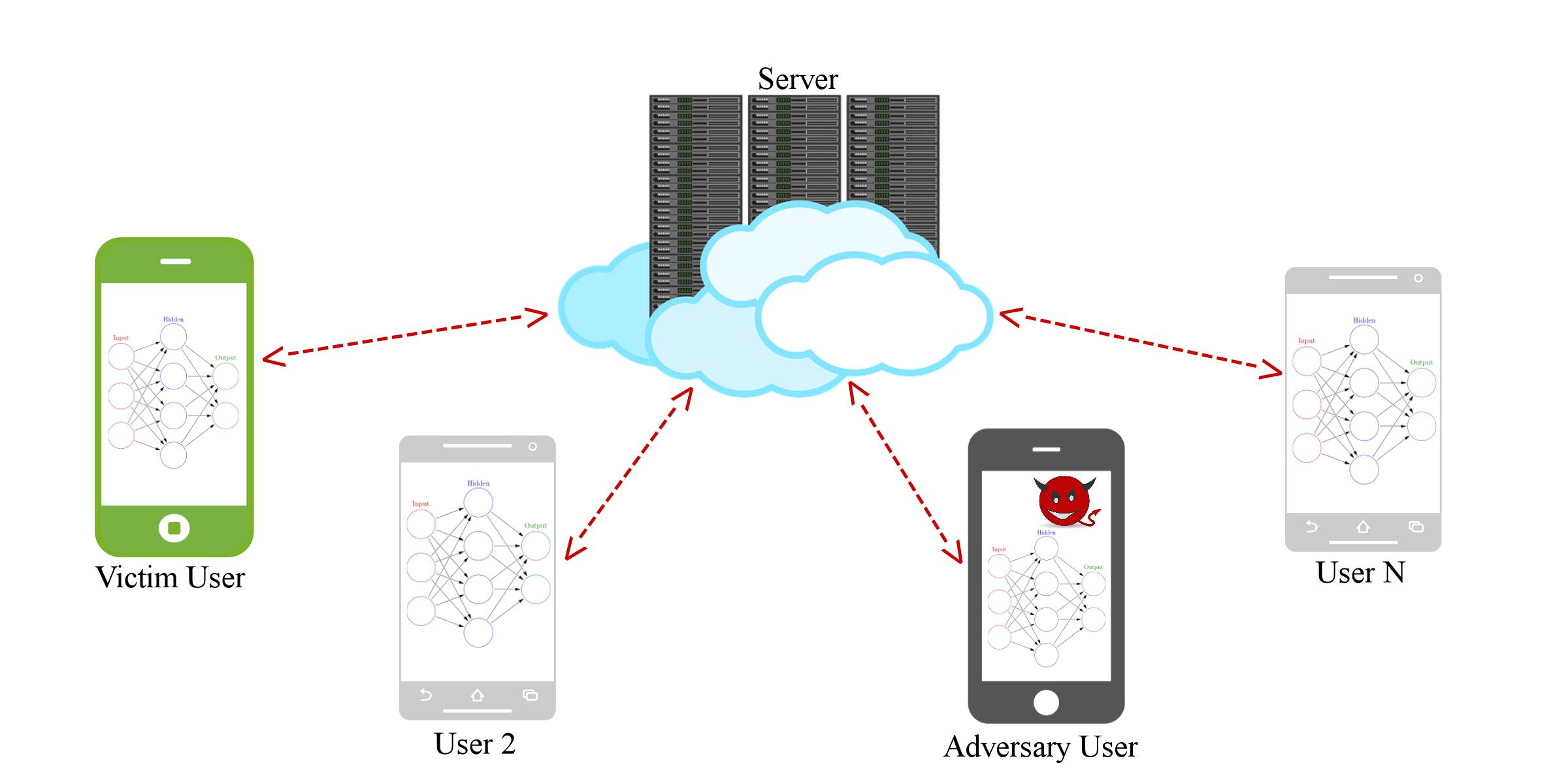
另一方面，许多数据拥有者，例如医疗机构可能由于数据隐私和保密问题，不能通过共享数据进行大规模的深度学习，而且当数据范围过于狭小时，训练过程容易出现过拟合的问题。就为解决这个问题，研究人员最近提出了协同型深度学习模式。采用合作学习算法，如图1（b）所示，其中每个参与者在他的设备上训练一个局部模型，并与其他用户共享模型的一小部分参数。 通过收集和交换这些参数，服务运营商可以创建一个训练有素的模型，与使用集中式方法构建的模型几乎一样精确。 由于数据集不直接暴露，因此分散式方法被认为对隐私更友好。 但它需要通过数据进行多次训练，用户在每个时期更新参数。各用户在自己的设备端进行模型的学习，然后通过上传自己的模型参数的一部分达到共享参数，协同学习的目的。而其他用户就可以从中央服务器下载这些参数进行模型的学习。多方协同深

图1.2 分布协同式学习模型

度学习系统将云端和多个终端构成的协同方按照一定协议共同完成深度神经网络模型的训练和运行过程。各个协同方训练一个本地深度模型，而云端服务器通过聚合本地模型共享的特征或参数得到全局深度模型并最终发布给各协同方。多方协同系统的搭建可以为地理上分布式的大量终端设备提供大规模智能化深度学习环境，在保护终端隐私数据的基础上协调云端和终端的决策。通过这种方式具有敏感度的数据着保存在用户自己的设备中，无需上传到中央服务器，保护了隐私。同时通过共享训练参数，其他用户可以进行大规模的学习与训练，得到具有更加精确结果的训练模型，在很大程度上防止了过拟合的问题。

1.2.2 深度学习中的隐私问题

研究人员考虑到深度学习的隐私影响。模型通常以集中的方式进行训练，即集中型深度学习，所有数据都传给中央服务器，用相同的训练算法进行处理。但考虑到如果数据是一个收集用户的私人数据，包括生活习惯、个人图片、地理位置等，如果中央服务器遭到攻击，则攻击者可以很容易地访问这些对于用户来说敏感的信息，最终造成个人信息的泄露。另一方面，通过在用户端构建一个GAN模型可以轻易地盗取其他任何一个用户设备中的隐私信息，即使他们并没有把这些数据统一上传到中央服务器。在协作学习中，任何用户都可能违反系统中其他用户的隐私而不涉及服务提供商。

关于机器学习中的隐私保护，现有文献主要针对传统的机器学习算法，如深度学习，并且涉及三个目标：用于学习模型的数据的隐私或作为对现有模型输入的隐私以及模型输出的隐私。基于安全多方计算（SMC）的技术有助于在多方对其专有输入执行协作机器学习时保护计算的中间步骤。 SMC被用于学习决策树，线性回归函数，关联规则，朴素贝叶斯分类器和k均值聚类。一般来说，SMC技术会带来不小的性能开销，并将其应用于隐私保护深入学习仍然是一个悬而未决的问题。保护模型隐私的技术包括隐私保护概率推理，隐私保护说话者识别以及加密数据计算。相比之下，我们的目标是合作培训一个神经网络，可以由每个参与者私下独立使用。差分隐私是隐私保护机器学习的一种流行方法。它已被应用于增强，主成分分析，线性和逻辑回归，支持向量机，风险最小化和连续数据处理。据我们所知，以前的工作都没有涉及到使用分布式随机梯度下降的多个参与者共同进行协作深度学习的这个问题。与以前提出的技术不同，协同式神经网络培训的背景下可以实现所有三个隐私目标：它保护培训数据的隐私性，使参与者能够控制学习目标以及揭示他们各自模型的多少，并让他们将联合学习的模型应用于他们自己的输入而不揭示输入或输出。此系统比安全多方计算或同形加密等密码技术低得多的性能成本实现这一目标深度学习中的部署。

1.3 本文的主要研究内容

多方协同深度学习系统将云端和多个终端构成的协同方按照一定协议共同完成深度神经网络模型的训练和运行过程。各个协同方训练一个本地深度模型，而云端服务器通过聚合本地模型共享的特征或参数得到全局深度模型并最终发布给各协同方。多方协同系统的搭建可以为地理上分布式的大量终端设备提供大规模智能化深度学习环境，在保护终端隐私数据的基础上协调云端和终端的决策。

但是，多方协同深度学习系统的训练过程中有可能会发生隐私数据泄露。本课题利用生成对抗网络技术实现对多方协同深度学习系统进行隐私探测攻击，发现多方协同深度学习系统的脆弱性。

2 协同深度学习系统设计

2.1 深度学习原理

2.1.1 神经网络

人工神经网络(Artificial Neural Networks,ANN)是在现代神经生物学研究 基础上提出的模拟生物过程以反映人脑某些特性的计算结构 。1943 年、神经生理学家，外科医的Mc　Culloch和伦理学家，数学家的Pitts提出了世界上第一个人工神经网络模型。神经网络代替人们解决复杂的问题。它成为了赋予机器智能重要的一个工具。1960年，Rosenblattが基于形式神经元的基础上提出了感知器。

人工神经元模型的三个基本组成要素：

（1）一组连接权(对应于生物神经元的突触)，连接强度由各连接上的权值表示， 权值为正表示激励，为负值表示抑制。

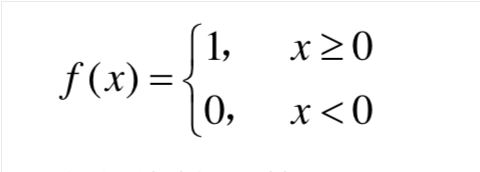
（2）一个求和单元，用于求取各输入信息的加权和(线性组合)。

（3）一个非线性激励函数，起非线性映射作用并限制神经元输出幅度在一定的 范围之内。此外还有一个阈值。阈值也被看作是一个输入分量，也就是阈值也是 一个权值。

激活函数主要有以下几种:

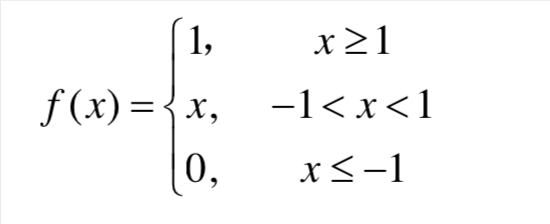
（1）阈值型函数

这种激活函数将任意输入转化为±1 或{0，1}两种状态，有时称为硬限幅函数， 用该函数可以把输入分成两类，其表达式如下:



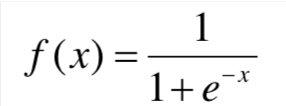
（2）分段线性函数

该函数在(-1，1)线性区间内的放大系数是一致的，这种形式的激活函数 可以看作是非线性放大器的近似，其表达式为:



（3）非线性转移函数

该函数为实数域 R 到[0，1]的非连续函数，代表了状态连续型神经元模型。最 常用的非线性转移函数是单极性 sigmoid 函数曲线，简称 S 型函数，其特点是函 数本身及其导数都是连续的，能够体现数学计算上的优越性，因而在处理上十分 方便，其表达式为：



一个感知器上可以接收多个输入，每个输入上有一个权值w，此外还有一个偏置项b，根据公式y=w\*x+b求出感知器的输出，并通过上述的激活函数来确定输出。

2.1.2 浅层学习与反向传播算法（BP）

多层感知机(Multilayer Perceptron，MLP)是单层感知机的推广，克服了 单层感知机不能对线性不可分数据进行识别的不足，多层感知机可以通过 BP 算法进行训练，因此多层感知机通常又称为 BP 神经网络。BP（Back Propagation）网络是1986年由Rumelhart和McCelland为首的科学家小组提出，是一种按误差逆传播算法训练的多层前馈网络，是目前应用最广泛的神经网络模型之一。典型的多层感知机由三层组成，即一个输入层，一个隐藏层和一个输出层，BP算法属于监督学习的一种，训练样本由输入特征x及对应的输出y（也叫标记）构成。BP算法基本上先正向输入训练神经网络模型，再通过反向依次计算每个隐藏层的误差项，直到与输入层相连的那个隐藏层，调整权值。但BP算法的问题是但迭代到一定程度时，容易出现过拟合的问题。

2.1.3 深度学习与卷积神经网络（CNN）

2006年，多伦多大学的教授—Geoffrey Hinton在《Science》上发表了一篇文章《Reducing the Dimensionality of Date with Neural Networks》[15]，该篇文章提出了深度学习的概念：机器学习中一种基于对数据进行特征学习的方法。并且指出：1.多隐层的人工神经网络具有优异的特征学习能力，学习得到的特征对数据有更本质的刻画，从而有利于可视化或分类；2.深度神经网络在训练上的难度，可以通过“逐层初始化”（layer-wise pre-training）来有效克服。在这篇文章中，逐层初始化是通过无监督学习实现的。这篇文章在当时引发了深度学习在学术界以及工业界的浪潮，使得人们开始关注深度学习。

2012年，Hinton教授的学生Krizhevsky[16]利用深度卷积神经网络(Convolution Neural Network)在2012年国际大规模视觉识别大赛(ImageNet Large Scale Visual Recogonition Chanllenge;简称：ILSVRC)中以较大的优势胜过了传统的目标检测算法。东京大学的松尾贵教授说道, “之前的人工智能的世界是以精度一点点向上提高，错误率一点点下降作为目的，进行着机器学习的研究。而这一次，据ILSVRC，这次多伦多大学开发的SuperVision将错误率一次性从２６％降到了１５％”。

深度学习神经网络与传统的浅层神经网络的最大区别在于前者具有更多的 层(通常在 3 层以上)，从而能够提取隐藏在数据中的某些深层次的模式。卷积神经网络与传统的 BP 神经网络最大的区别除了层数 外，前者采用了针对二维图像的处理结构，即卷积层和池化层。其中，卷积层 实现特征的提取，池化层实现特征的量化。池化操作的目的是为了对卷积提取 的特征进行空间的量化，这种量化操作能够在一定的条件下实现平移、旋转和 缩放等形变的不变性。通过若干层卷积层和池化层的操作，一个卷积神经网络 能够提取输入图像多层次的特征，并且高层特征较底层特征更稳定，也越适合 用来做模式识别和目标检测等任务。

2.1.4 Tensorflow

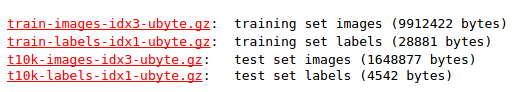
TensorFlow是一个深度学习库，由谷歌开源的第二代人工智能学习系统，从它的名字我们可以得知其运行原理：Tensor（张量）意味着N维数组，Flow（流）意味着基于数据流图的计算，TensorFlow为张量从流图的一端流动到另一端计算过程。

TensorFlow是将复杂的数据结构传输至人工智能神经网中进行分析和处理过程的系统。

TensorFlow是一个采用数据流图（data flow graphs），用于数值计算的开源软件库。节点（Nodes）在图中表示数学操作，图中的线（edges）则表示在节点间相互联系的多维数据数组，即张量（tensor）。它灵活的架构可以在多种平台上展开计算，例如台式计算机中的一个或多个CPU（或GPU），服务器，移动设备等等

2.1.5 测试数据集-MNIST

MNIST是一个入门级的计算机视觉数据集，它包含从“0”到“9”的各种手写数字图片，也包含每一张图片对应的标签用于说明图片是数字几。MNIST数据集被分成两部分：60000行的训练数据和10000行的测试数据集。其中每一个MNIST数据单元有两部分组成：一张包含手写数字的图片和一个对应的标签。每张图片为28\*28的大小，通道数为1.相当于编程时的“Hello world”，MNIST被广泛用于机器学习的实验中。数据集由四部分组成，分别是  ;也就是一个训练图片集，一个训练标签集，一个测试图片集，一个测试标签集；我们可以看出这个其实并不是普通的文本文件或是图片文件，而是一个压缩文件，下载并解压出来，我们看到的是二进制文件。



2.2 协同深度学习架构

2.2.1 服务器

参数服务器初始化参数向量w（全局），然后处理参与者的上传和下载请求。当某人上传梯度值时，服务器将上传的Δwj值添加到相应的全局参数中，并更新每个参数j的元数据和更新计数器statj。为了增加最近更新的参数的权重，服务器可以周期性地乘以衰减因子β，即stat：=β·stat。 这些统计数据在下载过程中使用，当参与者从服务器获取最大统计值参数的最新值时。 每个参与者通过设置θd来决定下载这些参数的哪一部分。在本研究中为了简化实验过程，采用默认全部上传，全部下载的手法。

2.2.2 本地训练

我们假设每个参与者维持一个神经网络参数w（i）的局部向量。参数服务器维护一个单独的参数向量w（全局）。每个参与者可以随机初始化他的本地参数，也可以从参数服务器下载其最新值。然后每个参与者使用标准的SGD算法训练神经网络，在许多迭代过程中，遍历他的本地训练数据。不同参与者在本地训练期间不需要进行任何协同。他们通过参数服务器间接影响对方的训练。

下图显示了分布式选择SGD（DSSGD）算法的伪代码。 DSSGD由每个参与者独立运行，每个学习时期由五个步骤组成。首先，参与者从服务器下载一个θd比例的参数，并用下载的值覆盖他的本地参数。然后，他在他的本地数据集上运行一个新的训练迭代步骤。这种训练可以按照一系列小批量完成;小批量是随机选择的，大小为M的训练数据的集合。无论参数服务器是否可信，训练数据之间是分离的，隐私保证仍然存在。例如，参与者可以匿名验证自己和他们上传的参数，而具有可提供安全性的可扩展匿名通信协议可用于隐藏参与者的身份。

选择初始参数w（i）和学习率α。重复下面步骤，直到获得近似最小值：

1.从服务器下载θd × w（i）参数并替换相应的本地参数。

2.根据（1），运行SGDonthelocal数据库和更新本地参数w（i）。

3.计算由于SGD导致的所有局部参数变化的矢量的梯度矢量Δw（i）。

4.将Δw（i）上传到参数服务器，其中S是集合S的指数最多为θu× w（i） 根据以下标准之一选择的渐变：

•最大值：将Δw（i）中的梯度按大小排序，并从最大值开始上传它们的θu比例的部分。

•随机与阈值：随机选择子值高于阈值τ的字样本。选择标准对于整个学习过程是固定的。

2.3 协同深度学习算法实现

假设有两个参与者A和V，为实现两者间的协同学习执行以下算法。

（1）建立并达成共同的学习结构和目标。

（2）V声明标签[a，b]和A标签[b，c]。

（3）针对多个用户运行协同深度学习协议，仅当参数服务器（PS）和两个本地模型的模型达到高于某个阈值的精度时才停止。

（4）受害者首先训练网络：

（a）V从PS下载一定百分比的参数更新他的本地模型。

（b）V的局部模型在[a，b]上训练。

（c）V上传他本地模型的参数选择到PS。

（5）其次，攻击者训练网络：

（a）A从PS下载一定比例的参数并更新他的本地模型。

（b）训练他当地的生成对抗网络（受害者不知道）模拟受害者的a类。

（c）A从GAN生成样本并将它们标记为c类。

（d）A的局部模型在[b，c]上训练。

（e）A将他本地模型的参数选择上传到PS。

（6）在4）和5）之间迭代直到收敛。

3 基于生成对抗网络的隐私攻击

3.1 生成对抗网络（GAN）

3.1.2 生成对抗网络原理

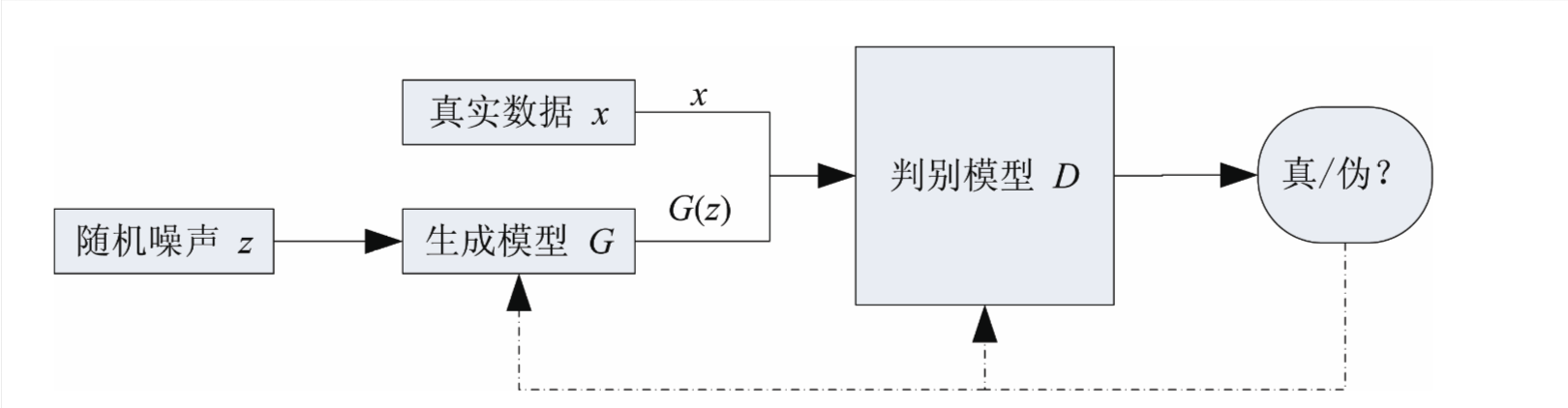
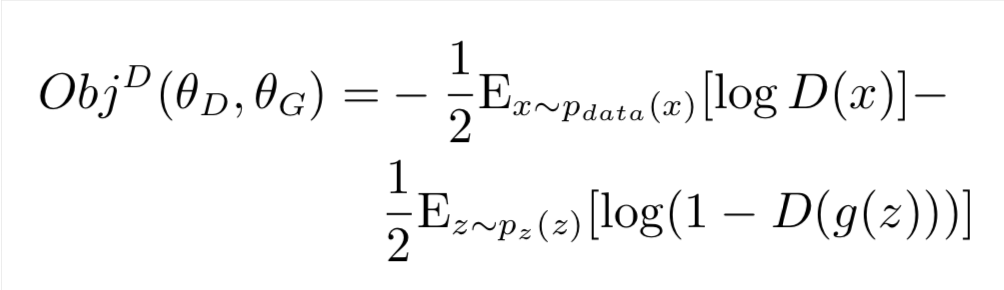
 生成式对抗网络 GAN (Generative adversarial networks) 是 Goodfellow 等[1] 在 2014 年提出的一 种生成式模型. GAN 在结构上受博弈论中的二人零和博弈 (即二人的利益之和为零, 一方的所得正是另一方的所失) 的启发, 系统由一个生成器和一个判别器构成. 生成器捕捉真实数据样本的潜在分布, 并生成新的数据样本; 判别器是一个二分类器, 判别输入是真实数据还是生成的样本. 生成器和判别器均可以采用目前研究火热的深度神经网络 . GAN 的 优化过程是一个极小极大博弈 (Minimax game) 问 题, 优化目标是达到纳什均衡[3] , 使生成器估测到数 据样本的分布. GAN模型从数学层面上看，一方面generator从训练数据中产生相同分布的samples，discriminator 则还是采用传统的监督学习的方法，最终达到所谓的纳什均衡，即对生成器来说目前采取的策略生成的图片为最接近原图的，同时对辨别器来说此时也是识别出生成器生成的图片的真假时的最优策略，达到全局最优。这一学习问题可被视为生成器和判别器之间的双人游戏——其中生成器提交尽可能接近真数据的内容，而判别器需要学习如何辨别真数据与假数据。这两个玩家都倾向于用尽量小的成本解决当前问题，博弈的解是纳什均衡，最终双方都不能单方面提高成本。

图3.1 生成对抗网络

在计算机视觉背景下，一个具有实际影响力的事实就是使用无限量的未标记图像和视频来学习良好的中间表示，然后应用到各种各样的监督学习任务中去，比如图像分类。生成对抗网络（GAN）可以在监督学习任务中重复利用生成模型和判别模型的一部分作为特征提取器。 GAN 有一个已经知道的问题就是训练的不稳定性，经常的结果的就是导致产生无意义的输出。2014年GAN被Ian Goodfellow等人提出以后，很多研究者在其之上提出了各种各样不同的改进型，现在我们能看到完全无监督的（GAN、Wasserstein GAN、BE-GAN、改进型Wasserstein GAN），和有条件的（Conditional GAN、Auxiliary Classifier GAN）。

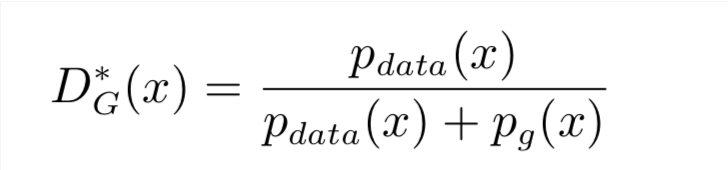
3.1.3 GAN的学习方法

优化判别器D：



x 采样于真实数据分布 pdata(x), z 采样于先 验分布 pz (z) (例如高斯噪声分布), E(·) 表示计算期 望值. 这里实际训练时和常规二值分类模型不同, 判 别器的训练数据集来源于真实数据集分布 pdata(x) (标注为 1) 和生成器的数据分布 pg (x) (标注为 0) 两部分.

当给定生成器的情况下，目标函数在



处得到最小值，即为判别器的最优解。D 的目标是正确判断数据来源, 使得 D(G(z)) 趋近于 0, 同时 G 的目标是使得其趋近于 1. 生成器 G 的损失函数为

ObjG(θG) = −ObjD(θD, θG)

对于 GAN 的学习过程, 我们需要训练模 型 D 来最大化判别数据来源于真实数据或者伪数 据分布 G(z) 的准确率, 同时, 我们需要训练模型 G 来最小化 log(1 − D(G(z))). 这里可以采用交替优 化的方法: 先固定生成器 G, 优化判别器 D, 使得 D 的判别准确率最大化; 然后固定判别器 D, 优化 生成器 G, 使得 D 的判别准确率最小化. 当且仅当 pdata = pg 时达到全局最优解. 训练 GAN 时, 同一 轮参数更新中, 一般对 D 的参数更新 k 次再对 G 的参数更新 1 次.

3.1.4 **GAN的实际应用**

GAN作为一个有力的生成模型，可以用来建模，生成还原度极高的图像，视频等。同时利用GAN还可以修复图像中的细节，使得局部信息更加丰富。在语音领域，利用SeqGAN可以生成逼真自然的诗词，音乐等。GAN与强化学习相结合，可以帮助检测恶意代码，用GAN生成具有对抗性的病毒样本。GAN 作为一种生成式方法, 有效解决了可建立自然 性解释的数据的生成难题. 尤其对于生成高维数据, 所采用的神经网络结构不限制生成维度, 大大拓宽 了生成数据样本的范围. 所采用的神经网络结构能 够整合各类损失函数, 增加了设计的自由度.

3.2 隐私攻击框架设计

攻击者A参与协同学习协议。所有参与者事先同意一个共同的学习目标，这意味着他们就神经网络架构的类型和训练发生的标签达成一致。让V成为另一个声明标签[a，b]的参与者（受害者）。攻击者A声明标签[b，c]。因此，虽然b是共同的，但A没有关于a类的信息。对手的目标是尽可能多地推断出有关a中元素的有用信息。攻击者将这些假冒的样品从c类注入到协同学习程序中。通过这种方式，受害者需要更努力地区分a和c类，因此会揭示更多关于a类的信息。因此，内部人员模仿来自a的样本，并利用受害者提高他在训练前忽略关于类别[a]的知识。在这种情况下，我们使用这个属性欺骗受害者，提供更多关于内部人员不知道的类别信息。

为了简单起见，我们考虑只有两位用户（攻击者和受害者），然后将我们的攻击策略扩展到多个用户。每个用户可以声明任意数量的标签，而且不需要重叠。

3.3实际攻击场景设计

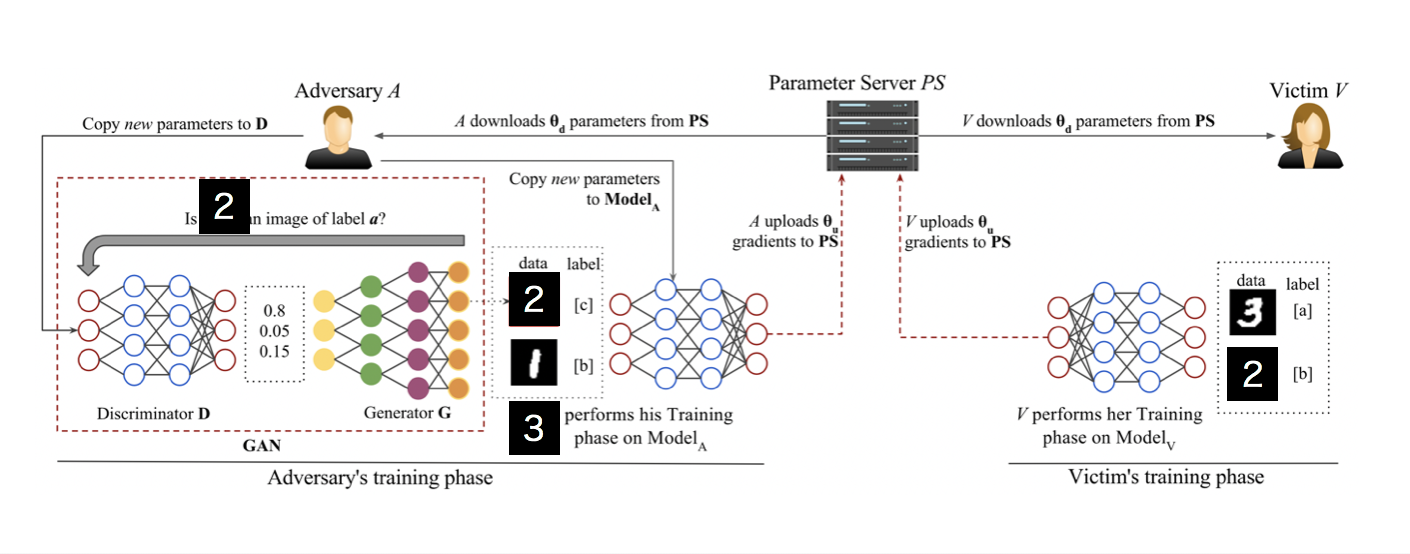
 这里假设上文提到的协同学习场景中只有两个参与者（A，V），其中A为攻击者，V为受害者。假设受害者V有两类图片“1”，“2”，分别申明标签 [b, c]，攻击者A有数字“0”，“1”，标签为[a, b]。同时在攻击者一侧有GAN的网络模型，用以从中央服务器下载下来的参数中生成和受害者V的图片“2”以假乱真的图片，并将假冒的图片“2”申明为标签 [3]。攻击者的最终目的是获取受害者 V 设备中的有关于标签[c],数字”2”的信息。

图3.2 协同学习场景下利用 GAN进行隐私攻击

3.4 隐私攻击算法与实现

3.3.1 具体算法

- 训练前的准备

参加者对模型的参数事前进行确认，确保训练学习过程中采用一致的参数，具体有：

1.相同的学习模型，标签。

2.学习速率

3.上传比例参数（实验中采用全部上传）

4.下载比例参数（实验中采用全部下载）

- 学习过程

对于每一次迭代，采取以下步骤：

1.用户从中央服务器下载参数到本地。

2.更新本地模型

3.如果用户是攻击者：

3.1创建一个本地模型的复制模型，作为GAN的识别器D

3.2运行生成器G和识别器D，目标生成尽可能真实的标记为a的受害者数据。

3.3根据生成器G基于识别器D的结果

3.4生成器G生成n个目标是标签a的样本图片

3.5将生成的图片标记上标签c（虚假标签）

3.6将生成的图片和攻击者的本地数据库中的图片混合

4.如果是受害者：

4.1用新下载的参数更新本地模型

4.2训练本地模型

4.3上传模型参数到服务器

5.将最后一次GAN的生成器生成的图片进行展示。在训练的最后攻击者方可能会生成一些只有受害者方才知道的图片信息。

3.3.2 实验

（1）硬件

本研究选择编程语言python,利用tensorflow的library进行模型的搭建。操作平台我们这里选用了Anaconda。硬件 macOS，3.1GHZ ， 内存8G。

（2）实验数据集

考虑到MNIST数据集中的数据采用了随机排放，实验中需将攻击者和受害者的数据区分开，用以实现数据的窃取。本研究采用一个程序[1]对MNIST数据集进行了从二进制数据到图像的转化，然后我们得到一个由数字集[0 : 9]组成的文件夹。诚如上文所述，我们将选取少量的数据进行模型的训练：Adversary[0，1] Victim[1，2]。在这里我们选择相应的图像文件夹进行数据集的分类。随后通过对每个像素点的遍历来保存图像的二进制信息，对遍历完的图像给予相应的标签。其余的每个数据文件夹都采取此种方法。到此数据集的制作完成。

3.3.4 系统搭建

按照上文中所述的算法，我们构建了以下的几个函数：

* build\_inputdata函数用于将文件夹中的数字图像（.jpg）转化为二进制格式，并对图像标上相应的数字标签，保存在tensor中输出。

def build\_inputdata(input\_path,user\_type):

input\_count = 0

if user\_type == 'ADVERSARY\_MAIN':

if to\_restore\_local\_dataset:

print ("[INFO][OS]Restoring the dataset of {%s}" % user\_type)

input\_images = np.loadtxt('adversary\_main\_images.txt', dtype=int)

input\_labels = np.loadtxt('adversary\_main\_labels.txt', dtype=int)

input\_count = np.loadtxt('adversary\_main\_count.txt', dtype=int)

return input\_images,input\_labels,input\_count

print ("[INFO][OS]Writing in the datasets of {%s} from {./%s/}" % (user\_type,input\_path))

for i in range(0,2):

dir = './'+input\_path+'/%s/' % i

for rt, dirs, files in os.walk(dir):

for filename in files:

input\_count += 1

np.savetxt('adversary\_main\_count.txt', [input\_count], fmt='%d')

if user\_type == 'ADVERSARY\_MAIN':

for i in range(0,2):

dir = './'+input\_path+'/%s/' % i

for rt, dirs, files in os.walk(dir):

for filename in files:

filename = dir + filename

img = Image.open(filename)

width = img.size[0]

height = img.size[1]

for h in range(0, height):

for w in range(0, width):

# 通过这样的处理，使数字的线条变细，有利于提高识别准确率

if img.getpixel((w, h)) > 230:

input\_images[index][w+h\*width] = 1

else:

input\_images[index][w+h\*width] = 0

if i == 0:

input\_labels[index][0] = 0

else:

input\_labels[index][0] = 1

index += 1

np.savetxt('adversary\_main\_images.txt', input\_images, fmt='%d')

np.savetxt('adversary\_main\_labels.txt', input\_labels, fmt='%d')

print ("[INFO][OS]Writed in {%s} images {%s} labels of {%s}" % (user\_type,input\_count, input\_count))

return input\_images,input\_labels,input\_count

* build\_generater和build\_discriminater用来生成生成对抗网络模型。每次迭代训练大小为一个batch\_size，256个数据。
* build\_discriminater，这里我们采用三层网络模型，第一层为将图像28\*28，即784个元素的输入转化为300个元素的输出；第二层网络将上一层输出的300元素作为输入转为150个元素的输出；最后一层采用1个输出，作为判别新生成的图像是否足以以假乱真。输出我们采用西格蒙德函数作为激活函数。

def build\_discriminator(x\_generated, keep\_prob):

# tf.concat

#x\_in = tf.concat([x\_data, x\_generated],0)#connect the tensor of data original with a size of [batch\_size,img\_size] and the generated data with the same size of [batches\_size,img\_size]

w1 = tf.Variable(tf.truncated\_normal([img\_size, h2\_size], stddev=0.1), name="d\_w1", dtype=tf.float32)

b1 = tf.Variable(tf.zeros([h2\_size]), name="d\_b1", dtype=tf.float32)

h1 = tf.nn.dropout(tf.nn.relu(tf.matmul(x\_generated, w1) + b1), keep\_prob)

w2 = tf.Variable(tf.truncated\_normal([h2\_size, h1\_size], stddev=0.1), name="d\_w2", dtype=tf.float32)

b2 = tf.Variable(tf.zeros([h1\_size]), name="d\_b2", dtype=tf.float32)

h2 = tf.nn.dropout(tf.nn.relu(tf.matmul(h1, w2) + b2), keep\_prob)

w3 = tf.Variable(tf.truncated\_normal([h1\_size,1], stddev=0.1), name="d\_w3", dtype=tf.float32)

b3 = tf.Variable(tf.zeros([1]), name="d\_b3", dtype=tf.float32)

h3 = tf.matmul(h2, w3) + b3

#the former output is the result of the original data1\*batch\_size]

y\_generated = tf.nn.sigmoid(h3)#the latter output is the result of the generated data[1\*batch\_size]

d\_params = [w1, b1, w2, b2, w3, b3]

return y\_generated, d\_params

* build\_generater，同样采取三层网络模型，第一层输入为随机的大小为100的噪声参数，输出转化为 150大小的输出；第二层采用300大小的网络模型；最后一层将上一层输入的300大小的网络转化为784大小的张量输出。这里我们利用tanh函数作为激活函数。

def build\_generator(z\_prior):#input is the tensor of images with the number of batches(28\*28\*batch\_size)

w1 = tf.Variable(tf.truncated\_normal([z\_size, h1\_size], stddev=0.1), name="g\_w1", dtype=tf.float32)#w1:z\_size 100 \*h1\_size 150 vertor;

b1 = tf.Variable(tf.zeros([h1\_size]), name="g\_b1", dtype=tf.float32)#b1:list of h1\_size 150 in 0;

h1 = tf.nn.relu(tf.matmul(z\_prior, w1) + b1)

w2 = tf.Variable(tf.truncated\_normal([h1\_size, h2\_size], stddev=0.1), name="g\_w2", dtype=tf.float32)#w2: h1\_size 150 \*h2\_size 300;

b2 = tf.Variable(tf.zeros([h2\_size]), name="g\_b2", dtype=tf.float32)#b2:list of h2\_size 300 in 0;

h2 = tf.nn.relu(tf.matmul(h1, w2) + b2)

w3 = tf.Variable(tf.truncated\_normal([h2\_size, img\_size], stddev=0.1), name="g\_w3", dtype=tf.float32)#w3: h2\_size 300 \*img\_size 28\*28;

b3 = tf.Variable(tf.zeros([img\_size]), name="g\_b3", dtype=tf.float32)#b3:list of img\_size 28\*28 in 0;

h3 = tf.matmul(h2, w3) + b3

x\_generate = tf.nn.tanh(h3)#output is a tensor of (28\*28\*batch\_size)

g\_params = [w1, b1, w2, b2, w3, b3]

return x\_generate, g\_params

* 受害者本地的神经网络

def

run\_victim(ps\_params,keep\_prob,i,input\_count\_victim,victim\_value,victim\_label,batches\_count\_victim,remainder\_victim):

v\_data = tf.placeholder(tf.float32, [batch\_size,img\_size], name="v\_data")

t = tf.placeholder(tf.float32, [batch\_size,1], name="t")

w1 = tf.Variable(tf.truncated\_normal([img\_size, h2\_size], stddev=0.1), name="d\_w1", dtype=tf.float32)

b1 = tf.Variable(tf.zeros([h2\_size]), name="d\_b1", dtype=tf.float32)

h1 = tf.nn.dropout(tf.nn.relu(tf.matmul(v\_data, w1) + b1),keep\_prob)

w2 = tf.Variable(tf.truncated\_normal([h2\_size, h1\_size], stddev=0.1), name="d\_w2", dtype=tf.float32)

b2 = tf.Variable(tf.zeros([h1\_size]), name="d\_b2", dtype=tf.float32)

h2 = tf.nn.dropout(tf.nn.relu(tf.matmul(h1, w2) + b2),keep\_prob)

w3 = tf.Variable(tf.truncated\_normal([h1\_size, 1], stddev=0.1), name="d\_w3", dtype=tf.float32)

b3 = tf.Variable(tf.zeros([1]), name="d\_b3", dtype=tf.float32)

h3 = tf.matmul(h2, w3) + b3

y\_data = tf.nn.sigmoid(h3)

v\_params = [w1, b1, w2, b2, w3, b3]

optimizer = tf.train.AdamOptimizer(learning\_rate)

victim\_loss = - tf.reduce\_sum(t \*tf.log(y\_data)+(1-t)\*tf.log(1-y\_data))

* 攻击者本地的主神经网络（区分于GAN模型结构）

def run\_adversary\_main(ps\_params,keep\_prob,i,batches\_count\_adversary\_main,remainder\_adversary\_main,adversary\_main\_value,adversary\_main\_label):

m\_data = tf.placeholder(tf.float32, [batch\_size,img\_size], name="m\_data")

t = tf.placeholder(tf.float32, [batch\_size,1], name="t")

w1 = tf.Variable(tf.truncated\_normal([img\_size, h2\_size], stddev=0.1), name="m\_w1", dtype=tf.float32)

b1 = tf.Variable(tf.zeros([h2\_size]), name="m\_b1", dtype=tf.float32)

h1 = tf.nn.dropout(tf.nn.relu(tf.matmul(m\_data, w1) + b1),keep\_prob)

w2 = tf.Variable(tf.truncated\_normal([h2\_size, h1\_size], stddev=0.1), name="m\_w2", dtype=tf.float32)

b2 = tf.Variable(tf.zeros([h1\_size]), name="m\_b2", dtype=tf.float32)

h2 = tf.nn.dropout(tf.nn.relu(tf.matmul(h1, w2) + b2),keep\_prob)

w3 = tf.Variable(tf.truncated\_normal([h1\_size, 1], stddev=0.1), name="m\_w3", dtype=tf.float32)

b3 = tf.Variable(tf.zeros([1]), name="m\_b3", dtype=tf.float32)

h3 = tf.matmul(h2, w3) + b3

y\_data = tf.nn.sigmoid(h3)

m\_params = [w1, b1, w2, b2, w3, b3]

optimizer = tf.train.AdamOptimizer(learning\_rate)

victim\_loss = - tf.reduce\_sum(t \*tf.log(y\_data)+(1-t)\*tf.log(1-y\_data))

victim\_trainer = optimizer.minimize(victim\_loss, var\_list=m\_params)

* 每次执行GAN随机生成一个batch的图像，共256张。保存到本地。

def show\_result(batch\_res, fname):

img = batch\_res \* 255

img = img.reshape(28,28)

new\_img = Image.fromarray(img.astype(np.uint8))

imsave(fname, new\_img)

3.4 结果分析

程序设定执行协同学习算法时共迭代200次，每次迭代过程按上文中所述算法执行攻击者，受害者上传与下载各步骤内容，其中受害者端，攻击者端每次分别执行batches数量步的本地迭代（50次）。学习速率我们采用0.0001.为使程序的效果便于查看，在程序执行过程中，不断输出相应的信息来表明目前程序处于哪个阶段。

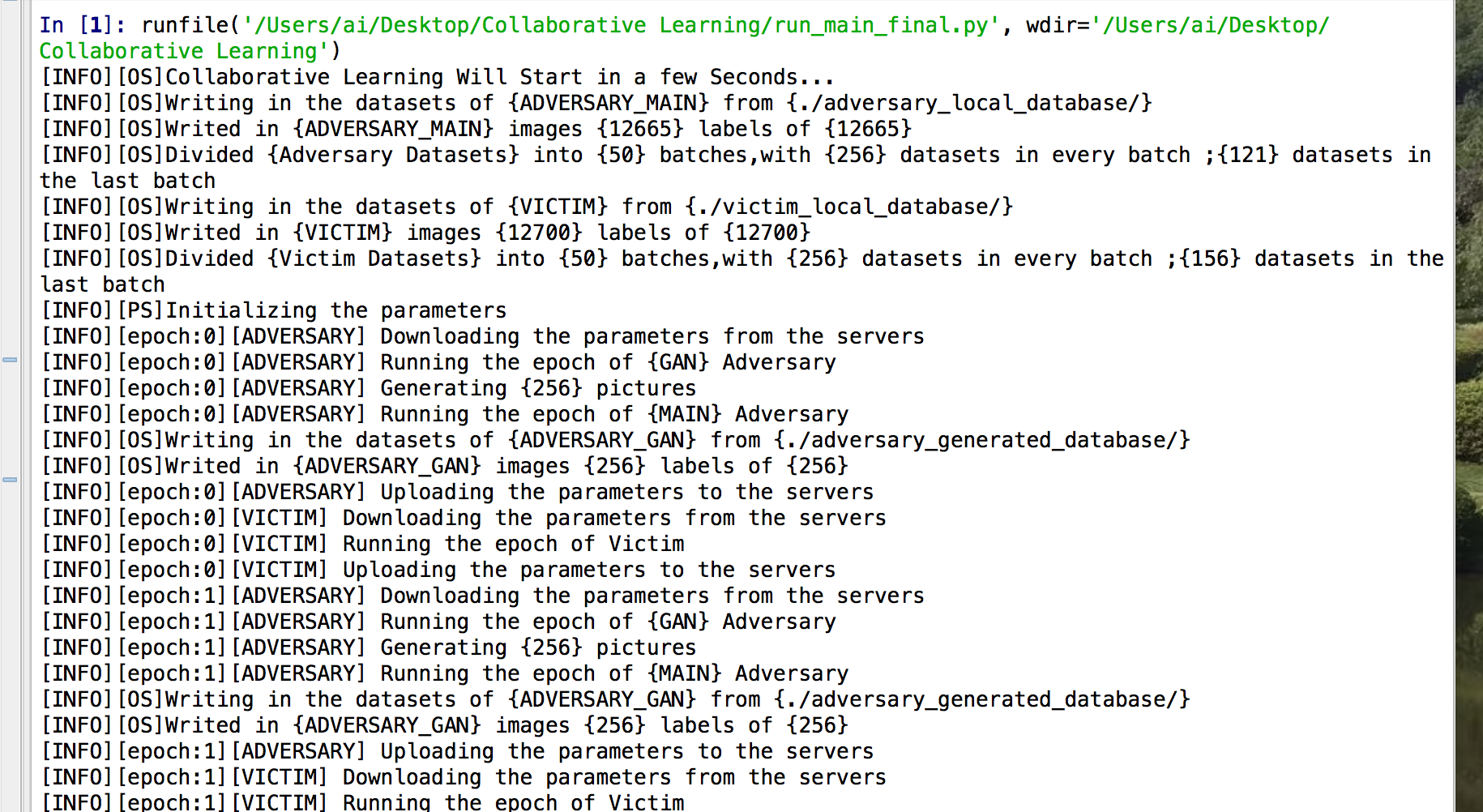


图3.3算法执行时的信息输出

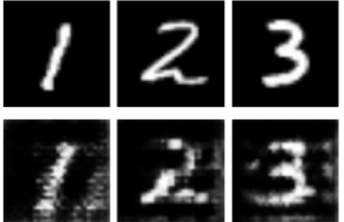


图3.5算法迭代200次后生成的图片

图3.4算法迭代100次后生成的图片

由生成图像可知，当迭代到100次时，GAN生成的图像已与原图像十分相似。当迭代200此时，我们得到了和受害者端基本一致的图像。

研究意义与结论

近几年来机器学习在大规模图像分类上取得了重大突破，有效解决图像识别中的问题。从一开始的感知器到后来的浅层网络，再到现在的深度学习网络，这种效仿人类大脑机制的模型算法展现出了极强的问题解决的能力。尤其是在图像识别领域，具有深度学习模型的图像识别系统得到了广泛的应用。基于选择性随机梯度下降的新型分布式训练技术的方法正在彻底改变着人工智能的领域。它可以帮助将深度学习的好处带到数据所有者不愿意通过机密性问题分享数据的域。GAN的出现加速了机器学习领域的发展同时也揭示了一些问题，本研究中讨论了协同学习场景下，利用GAN模型来窃取其他用户本地数据的场景，目的在于揭示协同学习场景中的数据安全的脆弱性，并对此考虑可行的解决方案。实验实现了协同式学习场景的搭建，并完成了协同学习任务系统的编写过程。在对生成对抗模型GAN的充分理解的基础上，实现了一方用户（攻击者）对另一方用户（受害者）数据的还原与再现。

在这项工作中，我们提出并实施了一种新型的在深度学习神经网络的协同学习环境中的攻击模型。我们的方法依赖于生成对抗网络（GAN），并且比现有的信息提取机制更加有效和更加全面。相信我们的工作将在现实世界中产生重大影响，因为各大公司正在考虑采用分布式，联合式或分散式深度学习方法来保护用户隐私。我们研究的主要观点是合作学习比它本应取代的集中学习方法更不理想。最后，我们无法制定有效的对付我们袭击的对策。解决方案可能依赖于安全的多方计算或（完全）同态加密。然而：（1）引入隐私保护协作学习是为了避免这些昂贵的密码学原型并且（2）我们基于它们探索的解决方案仍然容易受到某些形式的攻击。另一种方法是考虑不同粒度下的差别隐私。用户或设备级DP可以防止本文中设计的攻击。然而，目前还不清楚如何构建一个真正的系统，用于设备，类或用户级别的协作学习（例如，用户以不可预知的方式行为和共享数据）。

今后的课题，将进一步考虑能否采用一定的算法实现参数上传及下载筛选的部分。以提高GAN模型还原其他用户数据的效率及其精确度。再有，本次实验中D（识别器），G（生成器）采用了普通的3层结构模型，今后的研究将试着采用学习效果更好的CNN（卷积神经网络）和DP（深度学习网络）。基于试验结果给出对于深度学习的安全隐私问题的具体的解决策略。

参考文献

[1]原田達也：画像認識と画像復元；情報処理Vol.56 No.7(2015)

[2]松尾豊著（2016）『人工知能は人間を超えるか』KADOKAWA

[3]松尾豊:人工知能の未来とデータ共有の新しい可能性;情報管理vol. 58 no. 8 (2015)

[4]小山　俊隆、鈴木　崇啓、別所　史浩、原田　達也、國吉　康夫：移動体の自己位置情報を利用した画像からの異常領域検出；情報処理学会創立５０周年記念（第７２回）全国大会（２０１０）

[5]Ray Kurzuweil著、盛杨燕訳 (2016)『How to create a mind』浙江人民出版社

[6]Ray Kurzuweil 著、胡晓姣张温卓玛吴纯洁訳(2016)『The age of spiritual machines』中信出版集团

[7]Jeff Hawkins著、贺俊杰 李若子 杨倩訳（2006）『人工智能的未来』陕西科学技术出版社

[8]李承昊 ，蔡晨阳 ，李 锐 :基于神经网络的图像识别系统的研究;《科技传播》(2015)

[9]李义 :RESEARCH ON IMAGE RECOGNITION METHOD BASED ON CORRELATION LEARNING NEURAL NETWORK ;哈尔滨工业大学 (2015)

[10]河嵜光毅 吉川大弘 古橋武 :時系列データの異常検出を目的とした深層学習における再構築誤差の利用可能性に関する検討;The 29th Annual Conference of the Japanese Society for Arti cial Intelligence(2015)

[11] CS 224N / Ling 284, Neural Networks for Named Entity Recognition

[12]LeCun et al. Gradient-Based Learning Applied to Document Recognition 1998

[13]Tom M. Mitchell, "机器学习", 曾华军等译, 机械工业出版社

[14]Jake Bouvrie, Notes on Convolutional Neural Networks, 2006

[15]Ian Goodfellow, Yoshua Bengio, Aaron Courville, Deep Learning, MIT Press, 2016

[16]YuLT,ZhangWN,WangJ,YuY.SeqGAN:sequencegen- erative adversarial nets with policy gradient. arXiv preprint arXiv: 1609.05473, 2016

[17]Rezende D J, Mohamed S, Wierstra D. Stochastic back- propagation and approximate inference in deep generative models. arXiv preprint arXiv: 1401.4082, 2014

[18]HintonG,DengL,YuD,DahlGE,MohamedAR,JaitlyN, Senior A, Vanhoucke V, Nguyen P, Sainath T N, Kingsbury B. Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Processing Magazine, 2012, 29(6): 82−97

[19]Qi G J. Loss-sensitive generative adversarial networks on Lipschitz densities. arXiv preprint arXiv: 1701.06264, 2017

[20]Hinton, G. E. and Salakhutdinov, R. R. : Reducing the Dimensionality of, Science, Vol.313, No.July, pp.504-507 (2006).

[21]Bengio, Y., Courville, A. and Vincent, P. : Representation Learning : A Review and New Perspectives, No.1993, pp.1-34 (2012)

[22]Le, Q. V., Ranzato, M. A., Devin, M., Corrado, G. S. and Ng, A. Y. : Building High-level Features Using Large Scale Unsupervised Learning, Proc. of International Conference on Machine L e a r n i n g ( I C M L )( 2 0 1 2 ) .

[23]森永淳史，原健二，井上光平，浦浜喜一: “一般化ガウス分布 のパラメータ推定と画像識別への応用”， 映情メ年大， 22–7， (2014)

[24]渡部 啓吾, Bollegala, D., 松尾 豊, 石塚 満:Web からの

人物の属性情報抽出, 第 23 回人工知能学会全国大会 (JSAI2009) 論文集 (2009)

[25]王坤峰 ,苟超 , 段艳杰,林懿伦, 郑心湖,王飞跃:生成式对抗网络GAN的研究进展与展望;自 动 化 学 报 Vol. 43, No. 3 (2017)

[26]  K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. arXiv:1502.01852, 2015.   
[27]  G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed,  
N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. Signal Processing Magazine, 29(6):82–97, 2012.   
[28]  G. Jagannathan and R. Wright. Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. In KDD, 2005.   
[29]  P. Jain, V. Kulkarni, A. Thakurta, and O. Williams. To drop or not to drop: Robustness, consistency and differential privacy properties of dropout. arXiv:1503.02031, 2015.

[30]  A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, 2012.

[31]  Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proc. of the IEEE, 86(11):2278–2324, 1998.

[32]  M. Liang, Z. Li, T. Chen, and J. Zeng. Integrative data analysis of multi-platform cancer data with a multimodal deep learning approach. Trans. Comput. Biology and Bioinformatics, 12(4):928 – 937, 2015.

致谢

本课题在选题及研究过程中得到王洪涛老师的亲切关怀和悉心指导。定期与我就关于课题进行讨论与分析。激发了我对深度学习研究的动力，给我的科研道路树立了优秀的榜样，促使我打下了坚实的理论基础，提高了相关领域的研究水平，受益匪浅。 最终指导我完成了毕业论文的撰写。在此谨向王洪涛老师致以诚挚的谢意和崇高的敬意。

还有，要感谢日本Value Bridge公司在我毕业设计期间给予的帮助，感谢前田俊博,下地辉史两位工程师给予我的指导与帮助，对我的系统设计以及代码的编写与测试提供了宝贵的意见。

最后，我要感谢四年的大学生活，感谢所有老师同学以及我的家人，他们的支持与帮助，是我人生永远的财富。再次向所有关心过我，帮助过我的人表达最诚挚的谢意！