2018-12-25，**主要内容：**

深度学习初稿撰写要求：

1. 不能直接粘贴拷贝网上或者其他书籍内容，需按照自己的理解撰写。
2. 图表不能用现有图表，需重新制作，图用Vision绘图，表用Word制表。
3. 每章后需整理本章参考文献
4. 不局限于提纲内容，有修改或者新加内容随时和我讨论
5. 图文并茂，尽量通俗易懂 。

# 第七章 其他典型深度学习方法

**7.1 生成对抗网络**

## 7.1.1概述

生成对抗式网络（Generative Adversarial Networks，GANs）是一种以无监督方式学习目标分布的深度生成模型。这种目标分布的学习能广泛的应用于很多应用，包括：图像合成、图像风格转换、图像重构与分类等。GANs最早在2014年提出，是一种半监督与无监督的学习方法，它实现通过隐性数据的高维分布。GANs是训练一对相互竞争的网络，基于博弈论的思想。这里我们做一个形象地比喻，可以将其中一个网络视为造假者（生成器），另一个看作专家。学习的过程就是造假者尽最大努力仿造出接近真实的物品，专家（判别器）的职责就是将仿造品与真品进行比较，鉴别真伪。重复以上过程，专家和伪造者同时训练并相互竞争，直到两者达到一个均衡和谐的状态后，最终伪造者能产出更为真实的的物品，专家则一个判断能力较强的分类器。前者可以用于机器创作（自动画出“猫”“狗”），而后者则可以用来机器分类（自动判断“猫”“狗”）。

## 7.1.2算法原理

GANs基于博弈论的思想，其中生成器网络（generator network）与其对手判别器网络（discriminator network）相互竞争。首先生成器网络从训练数据中学习到统计分布，然后根据分布合成样本 x = g(z ; θ(g)) 。其对手，判别器网络会尝试区分生成器生成的样本和训练数据中抽取的样本。生成器由 d(x  ; θ(g)) 生成一个概率值来判别样本x是从训练数据中抽取的样本还是由生成器生成的 ‘赝品’ 。

生成对抗网络是一种生成模型，GANs的结构和我们之前见到的神经网络略为不同。大体上来说，GANs有生成器Generator和辨别器Discriminator组成，基本的结构图如下：



我们通常使用两个优化算法来训练GANs。判别器是一个普通的神经网络分类器，训练的过程中，辨别器 (discriminator) 学习引导生成器。

　　判别器：

　　在训练的过程中，我们向辨别器discriminator输入的数据一半来自于真实的训练数据，另一半来自于生成器生成的假图像。在训练的过程中，对于真实数据，判别器尝试向其分配一个接近1的概率（为更好泛化，一般会使用smooth参数将labels设为略小于1的值，如0.9）；而对于生成器生成的‘赝品’，判别器尝试向其分配一个接近0的概率。

也就是说，对于真实数据，我们使用label=1计算代价函数来训练判别器，其代价函数的计算方法为：

　　对于生成器，我们使用label=0计算代价函数来训练判别器，其代价函数的计算方法为：

　　　　　　　　　　　　d\_loss\_fake = tf.reduce\_mean(tf.nn.sigmoid\_cross\_entropy\_with\_logits(logits=d\_logits\_fake, labels=tf.zeros\_like(d\_logits\_fake)))

　　所以判别器的代价函数为：

　　　　　　　　　　　　d\_loss = d\_loss\_real + d\_loss\_fake

　　生成器：

　　与此同时，生成器尝试做相反的事情，它经训练尝试输出能使辨别器分配接近概率1的样本。生成器的代价函数为

　　　　　　　　　　　　g\_loss = tf.reduce\_mean(tf.nn.sigmoid\_cross\_entropy\_with\_logits(logits=d\_logits\_fake, labels=tf.ones\_like(d\_logits\_fake)))

　　随着以上训练的进行，判别器‘被迫’增强自身的判别能力，而生成器‘被迫’生成越来越逼真的输出，以欺骗判别器。理论上，最终生成器和判别器会达到一种均衡“纳什均衡”。

Discriminator和Generator损失计算

　　GANs和很多其他模型不同，GANs在训练时需要同时运行两个优化算法，我们需要为discriminator和generator分别定义一个优化器，一个用来来最小化discriminator的损失，另一个用来最小化generator的损失。即loss = d\_loss + g\_loss

　　d\_loss计算方法：

　　对于辨别器discriminator，其损失等于真实图片和生成图片的损失之和，即 d\_loss = d\_loss\_real + d\_loss\_fake , losses 均由交叉熵计算而得。在 tensorflow 中可使用以下函数：

tf.nn.sigmoid\_cross\_entropy\_with\_logits(logits=logits, labels=labels)

　　在计算真实数据产生的损失d\_loss\_real时，我们希望辨别器discriminator输出1；而在计算生成器生成的 ‘假’ 数据所产生的损失d\_loss\_fake时，我们希望discriminator输出0.

　　因此，对于真实数据，在计算其损失时，将上式中的labels全部都设为1，因为它们都是真实的。为了是增强辨别器discriminator的泛化能力，可以将labels设为0.9，而不是1.0。

　　对于生成器生成的‘假’数据，在计算其损失d\_loss\_fake时，将上式中的labels全部设为0。

　　g\_loss计算方法：

　　最后，生成器generator的损失用 '假' 数据的logits（即d\_logits\_fake），但是，现在所有的labels全部设为1（即我们希望生成器generator输出1）。这样，通过训练，生成器generator试图 ‘骗过’ 辨别器discriminator。

## 7.1.3应用场景

**参考文献**

7.6 深度森林

7.6.1概述

7.6.2 主流算法原理

7.6.3应用场景

**参考文献**

# 第十一章 深度学习在语音识别中的典型应用

11.1 声学模型

**11.1.1 概述**

**语音识别系统主要有四部分组成：信号处理、特征提取、声学模型、解码器**

**传统的声学模型：HMM-GMM 、CD-DNN-HMM**

**基于神经网络的模型：DNN-HMM、CTC算法（Connectionist temporal classification）**

**11.1.2主流算法**

**传统**

**深度学习**

**11.1.3场景应用**

**11.1.4 训练自己的声学模型：**

**需有数据集和代码**

**参考文献**