致谢：

首先要感谢我的指导老师张信明教授，从毕业设计题目的选定，到中期检查时候的方向的微调和后期的论文完成过程中，张老师的悉心教导令我受益匪浅。

其次要感谢中国科大超级计算中心，在前期的工作中，由于自己的笔记本性能不足，在学习过程中实践难度很大，超算中心提供了平台的支持。

还要感谢我的好朋友好同学，前期的实践不仅仅依靠超算中心，也依靠他，借用他的笔记本进行了不少运算。

更要感谢的是我的父母，养育之恩终身难以回报，更是提供了资金更换笔记本电脑，让我得以在自己的计算机上继续毕业设计，免去了超算中心的排队和借用同学电脑的不便之处。

最后感谢这个城市，这所学校，还有身边的同学、朋友，提供了一个合适的平台环境，让我学习成长。

摘要：

1. 绪论
   1. 背景
   2. 对抗生成网络
   3. 本文章节安排

本文对内容做如下安排：

第一章 绪论。介绍对抗生成网络的相关背景以及本人所做毕设的方向。

第二章 对抗生成网络基础。从神经网络入手，介绍神经网络的思想和部分算法，接着是卷积神经网络的算法原理，以及生成对抗网络的基础。

第三章 Wasserstein GAN的关键技术。首先介绍的是本人学习调研的一些生成对抗网络的改进思想，其次着重介绍Wasserstein GAN的改进方法，包括Wasserstein距离的优势和Wasserstein GAN的优势。

第四章 Wasserstein GAN在图像生成上的改进。本章所述为毕设中期的工作，是在图像生成方面对Wasserstein GAN的修改。

第五章 Improved Training of Wasserstein GANs 所做改进。本章所述是提出Wasserstein GAN的作者Martin Arjovsky于三月31日提出的gradient penalty改进。以及结合本人毕业设计之后的网络设计方案。

1. 对抗生成网络基础
   1. 神经网络基础

神经网络是一种模拟生物神经元结构和功能的模型，典型的神经网络模型包含结构、激活函数和学习规则三部分，其作用是通过简单的矩阵运算和非线性的激活运算来对数据进行处理。

下面所述的是一个简单的神经网络。



图1-1 神经网络示意图

如图1-1所示，这是一个简单的神经网络，包含两个输入变量、，三个中间变量、、，一个输出变量。其中所在为输入层，所在为隐藏层，所在为输出层。和被称为权重。计算方式为：

， (1.1)

例如：

(1.2)

为了表达和计算上的方便，用矩阵来表示如上运算过程：

 (1.3)

在实际的运用中，往往还会加上一个偏置项bias，则1.1、1.2、1.3式转变为如下：

， (1.4)

(1.5)

.  (1.6)

对于图1-1，没有表现出偏置项，在实现中，偏置项有多种方式可以添加进去，一种是如式1.6加上一个向量，还可以将偏置项视作1\*w，将偏置项的加法运算放入矩阵相乘中，式1.6转化为：

 (1.7)

其中。这种方式在计算上省去了一步向量相加，但是多出了一步向量的拼接操作，将拼接为了。如图1-2：



图1-2 神经网络示意图，带偏置项bias

在实际的运算中，拼接操作比向量相加消耗更多的计算。于是出现了又一种计算方法。考虑到每次拼接的开销，不妨考虑只在输入增加一个值为1的神经元，之后的除了最后一层的每一层，都附加一个神经元，但是该神经元的值由之前的计算得到，如图1-3：



图1-3 神经网络示意图，转变后的偏置项

相对应的计算公式为：

.  (1.8)

其中，作为隐藏层的偏置项参与运算。

神经网络里面一个举足轻重的反向传播算法back propagation，其作用在于调整网络使之成为我们所需要的状态，简单的来说就是复合函数的链式求导，用梯度下降的方法更新网络，有兴趣的可以参考！！！

* 1. 卷积神经网络

在后文中用到了另一个重要的网络层是卷积层，熟悉图像处理的人知道，卷积操作在图像中十分常见，对图像的平滑、模糊、锐化、边缘检测、浮雕等操作都可以通过特定的卷积算子来实现。下面简略的介绍卷积在图像中的操作方式。

卷积在图像中的操作方式往往是一个2D卷积，对于一张图像和一个卷积核，如图1-4所示，左边是一个5x5的图像，右边是一个3x3的卷积核。



图1-4 图像卷积示意(1)

有一个很形象的说法，将卷积核视为窗口，在图像上滑动，得到卷积后的结果。举个例子，在图1-4中，将卷积核置于图像的左上角，那么运算方式为，每个像素点的值乘以卷积核上与像素位置对应的值，最后累加得到结果。



图1-5 图像卷积示意(2)

运算的式子为：



然后将卷积核向右移动一个，如图1-6：



图1-6 图像卷积示意

对应的运算为：



将卷积核在图像上全部滑动一遍后，得到一个3x3的新图像。这里图像变小了，若是想维持图像大小不变，则在原图像四周补上数据，补充的数据的方式有多种，最简单的是用0填充，卷积操作连边缘都扫过，则可以得到和原图一样大小的图像。上述的卷积的滑动向左和向下的步长都是1，实际上步长还可以为更大的值，比如，步长为2，边缘补上数据，则卷积后的图像大小的长和宽都是原图像的一半。

在实际的运算中，卷积运算会转换成矩阵相乘来实现，首先对原图像和卷积核都做展开，上述的例子则可以转变成

* 1. 生成对抗网络