|  |  |
| --- | --- |
| **题 目** |  |
|  |  |
| **英 文** |  |
| **题 目Temporal Streams for** |  |
|  |  |
| **院 系** |  |
| **姓 名** |  |
| **导 师** |  |
| **日 期** | **二〇一六年五月** |



本科毕业论文



封面、扉页、致谢、目录、中文内容摘要、英文内容摘要、正文章节、参考文献或资料注释、附录

致谢

首先要感谢我的指导老师张信明老师，在张老师的指导下，我完成了毕业论文的选题、修改，也非常感谢张老师能够支持我做自己非常感兴趣的课题。在此期间，他时常关注我们的进展，并给出指导意见。

在此，我还要感谢好朋友王悦同学，首先是他邀请我一起做深度学习方面的课题，而且在课题研究过程中遇到问题的情况下，他给了我很大的帮助和建议。

感谢大学四年所有的老师。在这里老师们不仅传授给了我们知识、开阔了我们的视野，更多的是让我们拥有了更强的自学能力、更多的挑战自我的自信、以及坚持下去的毅力。感谢大学期间陪伴过我的每一个同学，正因为你们，大学生活才多姿多彩、欢声笑语。

最后还要感谢我的父母，我的女友，感谢他们对我的鼓励和支持，让我能够顺利完成学业。

感谢所有曾经支持过我、陪伴过我的每一个人！

目录

中文内容摘要

随着科学技术的发展，深度学习再次获得极大的关注，深度学习已经在很多方面表现出优异的性能，已成为当下十分火热的研究课题。

Abstract

第一章 绪论及背景

1.1背景

随着互联网的发展以及各种便携式终端的普及，人们越来越多的在互联网上分享自己的生活，图像作为一种非常常见的信息载体，也越来越受到广大网民的欢迎。人类所获取外界信息中视觉信息占据了80%，触觉、嗅觉等等占据了剩余的20%，所以图像相对于文字更能够表达当下心情以及感受。

如今，数字图像已经广泛应用在生活的每一个角落，几乎每一个人均有可以采集数字图像的终端设备，手机、相机等等各种产品的普及已经极大的改变了我们的生活。人们希望能够从图像中获得更多信息，然而由于不可避免的外界干扰，图像在采集、传输等过程中受到了噪声的干扰。

噪声影响了图像的可观性，使得图像所携带的原始信息丢失，甚至不可辨认。而在自动识别、遥感、测绘等方面，噪声所造成的影响更大。所以从图像中去除噪声干扰也是一个很重要的图像预处理操作。图像去噪的目的在于，从一个被噪声干扰的图像中，尽可能还原其原始信息。对于图像处理以及其他应用，图像的去噪颇有意义。

1.2 图像的噪声模型

信号获取以及传输过程中，会受到外在能量所产生信号的干扰，频率、强弱变化无规律，杂乱无章。我们可以认为一个受噪声干扰的图像为原始信息与噪声信息的叠加。图像去噪的任务就是去除图像中的干扰信息。由于干扰源众多且不固定，噪声特性复杂且具有非常强的随机性。图像的噪声信息我们可以将其认为一个随机过程，利用概率统计的方法来分析噪声。那么绝大多数图像噪声服从绝对值为0的高斯分布：

(1-1)

其中表示噪声中的灰度值，为噪声的标准差。那么一个受到噪声干扰的图像可以表示为：

(1-2)

其中Y为观察到的包含噪声的图像，R为不包含噪声的图像，N为噪声。从该模型上看，该加性噪声仅与外界因素有关，与原始图像无关。如图1.1所示

图1.1 图像的噪声模型

因此，图像的去噪过程可以描述为：给定包含噪声的图像Y，设计适合的算法从中计算出原始图像X。

1.4 图像去噪的发展状况

图像去噪的目的是通过一个包含噪声的图像获取无噪声的图像或者使其尽量接近原始图像。这同时还隐含着额外的目标：图像中原有的纹理、边缘等等细节信息需要尽可能的保留，只有如此才能真正发挥图像去噪的作用。然而图像中部分细节信息和噪声信息是极为相似的，噪声信息和细节信息均属于高频信息，将其完全分离是很难实现的，故在图片去噪过程中会不可避免的丢失部分细节信息。因此去噪的同时追求细节的保留是当前图像去噪算法的目标。

针对不同噪声的特点，现已出现很多图像降噪方法。图像去噪方法大体可以分为空间域降噪以及频域去噪两类。空间域降噪即直接对图像的灰度值进行处理以达到目的，均值滤波降噪、中值滤波降噪、纳维滤波降噪等等；频域去噪亦称变换域去噪，它首先将图像进行变换处理，将图像从原始空间域转换到频域，再对其进行处理，最后进行逆变换操作，至此完成降噪过程，常见的域变换方法有傅立叶变换、拉普拉斯变换等。

近年来，一些新颖的去噪模型被提出，包括非局部自相似性模型(nonlocal self-similarity models，NSS)、梯度模型(gradient models)、马尔科夫随机场模型(Markov random field models, MRF)等。特别是NSS模型如BM3D、LSSC等方法目前很受欢迎。

尽管以上模型具有很高的去噪质量，然而他们尚存一些缺陷。这些方法在测试阶段通常面临着复杂的优化问题，绝大多数不得不牺牲计算效率以达到预想的效果而难以实现高性能。

1.5 深度卷积神经网络

1.5.1 深度学习的背景

深度学习(deep learning)是机器学习的分支，是机器学习中一种基于对数据进行表征学习的方法。人工神经网络的历史已经较为久远。

机器学习从诞生到如今经历过两个发展阶段：浅层学习（shallow learning）以及深度学习（deep learning）。上世纪80年代后期，反向传播算法（Back Propagation，BP）的出现以及应用极大的推进了机器学习的发展，基于统计的机器学习模型引发了新的浪潮。反向传播算法在神经网络算法中可以自动修正网络中的参数，使得网络的输出更大程度符合训练数据。到如今，反向传播算法的应仍然十分广泛。90年代以后，支持向量机（Support Vector Machine， SVM）最大熵法等更多的浅层机器学习模型出现在人们的视线中。这些模型框架大多包含一层隐藏结点或者不包含隐藏节点。1989年，扬·勒丘恩（Yann LeCun）等人开始将1974年提出的标准反向传播算法应用于深度神经网络，这一网络被用于手写邮政编码识别。尽管算法可以成功执行，但计算代价非常巨大，神经网路的训练时间达到了3天，因而无法投入实际使用。而今，随着科技的进步以及计算性能的提升，神经网络以及深度学习再次引发研究的热潮。

1.5.2 深度神经网络

深度神经网络是指一系列由多个网络层堆叠起来的网络，每一层包含多个节点。神经网络的运算在节点中完成。每个节点对于输入的每个数据，使用一组系数（亦称权重）与之结合，作为该节点的数据或者经过激活函数后作为该点的数据。如下图所示



图1.2 深度神经网络节点

 (1-3)

其中 为第n层网络第i个节点的输出值，m为第n层网络的节点的数量，fun为激活函数。激活函数能够给神经网络加入一些非线性因素，通常拥有非线性、可微性、单调性等特点，它能够使得神经网络可以更好的解决问题。

常用的激活函数有ReLU、Sigmoid、tanh等。

（1）ReLU：近期ReLU激活函数越来越受欢迎，它的计算方式如下：

 (1-4)



图1.3 ReLU激活函数的函数曲线

ReLU激活函数的优点较为明显，它拥有较快的处理速度，而且在x>0情况下呈线性关系，故在一定情况下拥有更快的收敛速度。缺点在于：当传入一个极大的梯度，那么当更新参数以后，该节点对其他数据将失去作用。

（2）Sigmoid：Sigmoid激活函数是一个很常用的激活函数，它的数学表达式为：

 (1-5)

Sigmoid能够将实数集映射到0到1之间，函数图像如图1.4所示



图1.4 Sigmoid激活函数的函数曲线

Sigmoid拥有两个明显的优点：首先，Sigmoid的输出范围为(0,1)，连续稳定，范围有限，适合用作输出层。其次，Sigmoid求导较为容易，更加方便运算。不过Sigmoid也存在缺点，它容易出现梯度消失的现象，使得训练结果无法继续优化。

（3）tanh：tanh的数学表达式如下：

 (1-6)



图1.5 tanh激活函数的函数曲线

tanh函数位于[-1,1]区间，相对于Sigmoid激活函数，拥有更快的收敛速度。不过同样没有解决Sigmoid易出现的梯度消失问题仍是一个缺憾。

对于神经网络中，从数据输入的第一层开始，每一层的输出即为下一层的输入，直到最后一层输出层。



输入层 隐含层 输出层

图1.6 神经网络结构

如图1.6即为一个全链接网络。如上图，该网络一共有三层网络，分别是输入层，隐含层以及输出层。在该全连接中，每一个节点均与上一层所有节点保持连接，同时也与下一层所有节点保持连接。对于每一层连接，由节点连接到节点的权重记为，我们另有偏执项b，假设我们在隐含层使用ReLU激活函数，那么对于节点x(2,1)，我们有



 (1-7)

同样还有



 (1-8)

那么我们可以令

 (1-9)

 (1-10)

 (1-11)

 (1-12)

 (1-13)

 (1-14)

 (1-15)

那么，我们就有

 (1-16)

 (1-17)

即得：

 (1-18)

以上为一个深度神经网络的示例。深度学习相对于传统机器学习的区别在于**深度**。我们已知传统机器学习方法不包含或者仅包含一个隐含层。包含一层以及一层以上隐含层的神经网络即可成为深度神经网络，该网络系统可以被称为深度学习系统。

在深度神经网络中，每一层网络利用上一层网络的输出来学习识别特征。网络层次越深，整合、重组的特征便越多，且越复杂，网络所能够识别的特征越多，处理大规模高维度数据更具优势，但同时也会带来更多的计算任务。







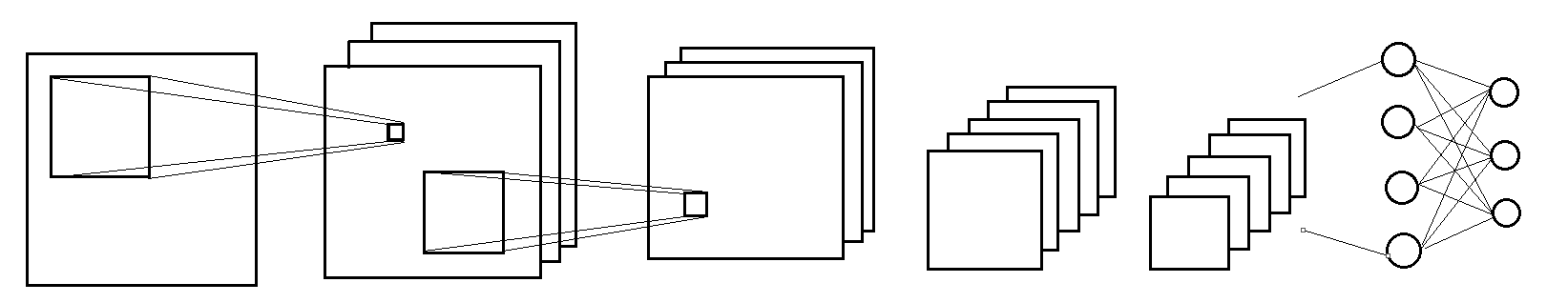


1.5.3深度卷积神经网络

以上说明是的是一个全连接神经网络。在本文所述方法中，我们使用了卷积神经网络。火热一时的AlphaGo、以及Google的GoogleNet和微软的ResNet等框架，均用到了卷积神经网络。卷积网络拥有**局部连接**、**权值共享**以及**下连接**的特征。相对于全连接神经网络，卷积神经网络更适合对图像、语音等方面的处理，主要是有以下几个原因：

1. 图像中包含较多的位置信息。对于图像来说，每一个像素与周围邻近像素会拥有较大的联系，而对于距离较远的像素，往往联系极小。全连接网络对于每一个像素，都会平等看待，这在一定程度上不符合我们的预期。而卷积网络拥有的**局部连接**的特征，相对于全连接网络，能够更好地利用像素之间的位置关系，提高信息的利用效率。
2. 图像处理需要较多的节点数量。比如一个512\*512像素的图像，输入层节点数量为512\*512=262144个结点，我们假设其下一层网络有100个节点，那么所需要的参数数量为262144\*100=26214400，多余两千万个参数。若图像尺寸扩大或者隐含层节点数量增加，在本文所述需求下，输出节点数与输入节点数相同，那么所带来的计算压力以及内存占用需求将非常庞大。卷积网络拥有**局部连接**的特点，无需连接上一层每一个节点，这样就减少了大量的参数。同时，卷积网络的**权值共享**，对于一组连接共同使用一个参数，再一次大大减少了参数的数量。
3. 图像的特征信息数量较多。而我们使用全连接网络增加网络层的数量来提高网络对特征信息的提取，当网络层次数量越多，训练将越加困难，因此，很深的全连接神经网络对于我们的使用并不友好。而在卷积网络中，卷积网络的**下采样**可以在一定程度上提取信息，并减少每层的样本数量，同时，可以减少参数数量，提升了其鲁棒性，并提高了性能。

接下来我们说明一下卷积神经网络的工作原理。



输入层 卷积层1 池化层1 卷积层2 池化层2 全连接层 输出层

图1.7 卷积神经网络结构

如图1.7，是一个经典的神经网络结构。通过图1.7和图1.6的对比，我们能够发现，卷积神经网络与全连接网络有很大的不同。对于每一层网络，全连接网络每层的节点排成一列，而卷积网络，会充分利用位置信息。图1.7所示网络，输入层的长和宽分别对应图像的长和宽，在卷积层1，我们使用多个Filter对输入层进行卷机操作得到多个输出结果，我们称之为Feature Map。我们在卷积操作以后，又经过池化层的**下采样**处理，得到同等数量的Feature Map，再次经过卷积处理以及池化处理后，经过全连接层，得到这个网络的最终结果。

1.5.4 神经网络的训练

目前神经网络的训练多用反向传播算法（Back Propagation）。我们可以将反向传播算法分为以下几个步骤：

1. 前向计算神经网络每一层每一个节点（即神经元）的值x；
2. 反向计算计算网络中每一个节点的误差项。记损失函数为E，节点的加权输入为p，那么误差项 ；
3. 对于每一个节点计算其连接权重的梯度，对于从i节点到j节点的连接，数学表达式为：，其中，x为节点i的输出，e为节点j的误差项。
4. 我们使用梯度下降法则更新权重。对于（3）中所述权重，我们有：

 (1-19)

然而，由于卷积网络的局部连接、下采样操作，以上所述误差项e的计算并不适用于卷积网络。同样，由于卷积网络权值共享，以上所述的权值w的更新方法并不适用于卷积网络。那么，对于卷积网络，当Filter的数量为N时（此时，输出层的深度亦为N），对于第i层的节点，第i-1层的每一个输出值p均会对第i层所有Feature map产生影响，故这里不再使用偏导，而使用全导。我们使用sensitivity map来记录该结果。若输入层深度为D，数学表达式如下：

 (1-20)

而对于的计算，数学表达式为：

 (1-21)

我们同时可以采用批量归一化（Batch Normalization，BN）操作加快我们的训练过程。对于批量归一化操作，我们可以利用以下几个数学表达式来表达：

 (1-22)

 (1-23)

 (1-24)

 (1-25)

批量归一化操作可以解决一些训练时收敛速度很慢、梯度爆炸无法继续训练的情况。对于一般的神经网络，批量归一化仍然能够加快训练速度、提高训练精度。

1.6 本文章节安排

本文的章节安排如下

本文的章节安排如下

本文的章节安排如下

本文的章节安排如下

第二章 相关工作

2.1 卷积神经网络应对图像处理

在本文中，我们将图像的去噪问题视为一种辨别学习问题。与以往的去噪方法不同,本文并不是首先去判别噪声模型,而是使用前馈式卷积神经网络(convolutional neural networks, CNN)区分将噪声信息与图像原始信息分离。相对于传统方法，使用CNN有以下三个优点：首先，CNN的深层网络结构可以提高发掘图像的特征信息的能力而且拥有较高的灵活度。其次，对于CNN的训练，目前已经出现了很多可以提升其训练效果以及加快训练速度的方法，比如归一化、线性滤波器（Rectifier Linear Unit，ReLU），这些方法均可在一定程度上提高网络的去噪性能。最后，深度卷积神经网络中包含大量的可并行运算，非常适合利用GPU来提高运算性能。

本文所使用的方法与以往的神经网络略有不同。对于输入的包含噪声的图像Y，我们并不去预测干净的图像X，而是去预测隐藏的噪声N。我们使用原始图像与包含噪声的图像的差异去训练该神经网络，以加快训练速度以及最终去噪效果。不同于典型的卷积网络模型，我们针对图像去噪设计使用了自己的神经网络。对于输入任意尺寸的图像，该神经网络将输出相同尺寸的噪声，该噪声为图像中所包含的噪声。由式(1-2)我们即可预测不包含噪声的原始图像信息。

我们首先针对特定噪声去训练该神经网络，对比当前较为优秀的去噪算法如1.4章节中所述BM3D、LSSC以及TNRD算法，我们能获得效果相当甚至更优的去噪效果，借助当前强大的GPU并行加速，使用CNN去噪还能够获得相对较高的性能。

技术的进步，使得大数据、机器学习等方面获得巨大的进步。大规模数据集的使用更加便捷，深度学习的价值进一步体现，卷积神经网络更是在图像处理等方面获得明显的进步，深度神经网应用于图像处理更加便捷。

第三章 深度卷积网络模型

本章节中，将介绍本文所提出的图像去噪的CNN模型，并扩展应用它去处理图像去噪的任务。一般来说，训练一个神经网络来达到我们预期的效果需要两个步骤：

（1）网络结构设计

（2）使用训练数据对网络模型进行训练。

对于（1）网络结构设计，本文将介绍一个卷积神经网络结构，并对其进行修改以适合图像的降噪处理。对于（2）神经网络的学习，本文利用包含噪声图像与清洁的图像的差值，对该神经网络进行训练。最后扩展该降噪网络并使其适用于更多情况。

3.1 卷积神经网络基础

在本项目开题之处，为了深入理解深度卷积网络，我们做了以下工作：分别使用caffe开源框架以及Google与2015年末推出的Tensorflow框架，构建深度卷积神经网络进行USTC验证码的识别。对于获取到的每一个验证码

http://mis.teach.ustc.edu.cn/randomImage.do?data=

该验证码图像尺寸为80\*20，我们将其从中分离为四个20\*20的部分，将其分类后进入训练网络。已知验证码中可能出现的字符为除去数字0、数字1、字母I以及字母O的所有大写字母以及阿拉伯数字共32个字符。输入数据训练为n个尺寸20\*20的，我们将其转化为单通道图像并将其转化为n\*20\*20\*1的四维数据，训练的label为长度为32的独热码。损失函数我们使用最小二乘法则，即

 (3-1)

使用最速下降法，并使用adam优化算法进行训练。

那么，该数据会进入该神经网络的训练：

1. 第一层卷积层:在这一次卷积网络中使用VALID填充类型，VALID类型不会在原有数据基础上填充数值。这里设置步长为[1,1]，设置每一个卷积核的尺寸为5\*5，我们共使用20个卷积核分别对网络进行卷机操作，提取出20个feature map。一般来说，不同的feature map对应不同的特征，对应的，我们设置了20个偏置量，并使用ReLU激活函数来完成这一层网络。
2. 第一层池化层：在这里，我们使用了尺寸为2\*2的窗口进行池化，池化操作为max pooling，即在每一个2\*2窗口中，最大值为输出值，同时，也减少了数据的规模。该层输出尺寸为5\*5\*20.
3. 第二层卷积层：在该层卷积层中，同样使用了VALID填充类型。第二层中，使用50个5\*5的卷积核对以上输出结果进行卷积操作，对应的，设置了50个偏置量，我们对以上结果使用ReLU激活函数进行激活操作，最终该层输出50个feature map。
4. 第二次池化层：继续使用2\*2的窗口对上层的输出进行max pooling池化，该层的输出尺寸为2\*2\*50。
5. 全连接层：该层拥有500个节点（即神经元）。对于上层的输出，我们首先改变其尺寸，将其reshape成2\*2\*50=200单位长度的一维向量，设置偏置量，我们对最终结果使用ReLU激活函数进行激活操作。
6. 添加Dropout操作来减少神经网络中常常出现的过拟合现象。
7. 输出层使用全连接的方式来实现，该层节点数对应分类数量。该识别中共有32种输出，故该层拥有32个节点（即神经元），我们对以上过程的输出使用softmax激活函数对其进行激活操作。

以上为该神经网络的结构。由于该数据集特征明显，在充分训练后，得到无限接近100%的正确率。

3.2 深度卷积网络应用于图像降噪

由3.1已经描述了一个经典的神经网络，考虑到图像去噪的特征，对于一定尺寸输入的原图，网络期望的输出的噪声的尺寸与输入图像相同。那么对于尺寸为256\*256的图像，若取得最终理想的输出，那么最终的全连接层所需参数数量至少为(256\*256)\*(256\*256)= 4,294,967,296，如此庞大的数量必然会带来巨大的性能的下降，且无法针对图像的多种尺寸进行灵活的改变。于是，在该卷积神经网络中，删除了全连接层。为了避免信息的丢失，本文所述方法删除了全部的池化层，全部使用卷积层、批量归一化（Batch Normalize）以及激活层处理。

那么，对于输入数据，首先对数千张图像进行预处理。便于训练，我们将所有图像压缩为尺寸为256\*256的缩略图，将其转为单通道灰度图。手动添加高斯噪声后作为训练数据，噪声为期望网络输出的结果，并保存为mat格式。

对3.1所述网络进行修改，网络结构如下：

（1）输入：首先将输入的尺寸为256\*256的灰度图reshape为尺寸为256\*256\*1的数据格式，并做归一化处理，使其数据范围在0至1之间。

（2）第一层卷积层：在这一层网络，我们使用尺寸为5\*5的卷积核对输入的图像进行卷积操作，卷积核的数量为24个，填充模式为SAME，添加偏置项，进行批量归一化操作（Batch Normalize）后使用ReLU激活函数进行激活操作，该层输出结果为24个feature map。

（3）第二层卷积层中，同样使用了尺寸为5\*5的24个卷积核对上一层网络的输出进行卷积操作，填充模式同样为SAME，添加偏置项、并进行批量归一化操作（Batch Normalize）后，进行ReLU激活操作，该层的输出结果为同样为24个feature map。

（4）第三次卷积网络中，继续使用5\*5的尺寸对上一层网络的输出结果进行卷积操作，卷积核的数量仍然为24，SAME填充模式、偏置项、批量归一化以及ReLU激活函数处理后，得到24个feature map。

（5）最后一层卷积层，亦为该网络的输出层。在该层网络中，对应单通道灰度图，仅使用一个卷积核处理，该卷积核尺寸为5\*5，填充类型为SAME，以保证输出尺寸与输入尺寸相同。该层输出为本网络的输出。

代码描述：

x\_image = tf.reshape(x, [-1, 256, 256, 1])  
  
W\_conv1 = weight\_variable([5, 5, 1, 24]) *# 第一层卷积层*b\_conv1 = bias\_variable([24]) *# 第一层卷积层的偏置量*h\_conv1 = tf.nn.relu(conv2d(x\_image, W\_conv1) + b\_conv1)  
  
W\_conv2 = weight\_variable([5, 5, 24, 24]) *# 第二次卷积层*b\_conv2 = bias\_variable([24]) *# 第二层卷积层的偏置量*h\_conv2 = tf.nn.relu(batchnormalize(conv2d(h\_conv1, W\_conv2) + b\_conv2))  
  
W\_conv3 = weight\_variable([5, 5, 24, 24]) *# 第三次卷积层*b\_conv3 = bias\_variable([24]) *# 第二层卷积层的偏置量*h\_conv3 = tf.nn.relu(batchnormalize(conv2d(h\_conv2, W\_conv3) + b\_conv3))  
  
W\_conv4 = weight\_variable([5, 5, 24, 1]) *# 第四次卷积层*b\_conv4 = bias\_variable([1]) *# 第二层卷积层的偏置量*h\_conv4 = conv2d(h\_conv3, W\_conv4) + b\_conv4  
y = tf.reshape(h\_conv4, [-1, 65536])

本文所述方法中，对于Y=X+N的加性高斯噪声模型，该网络的输出为y≈Y-X=N，对于包含噪声的输入图像，我们使用X=Y-y来获得预测的清洁图像。于是损失函数的计算，采用最小二乘方法：

 (3-2)

式中m、n分别为图像的宽度以及高度。

对于固定尺寸的图像，可以使用以下表达式来代替：

 (3-3)

即：

loss = tf.reduce\_sum((y\_ - y)\*\*2)

train\_step = tf.train.AdamOptimizer(2e-4).minimize(loss)

综上，假定网络的深度为d，本文所述卷积神经网络的结构可分为以下三种类型:

1. 对于第一层网络，使用卷积操作并使用ReLU激活。卷积核尺寸为5\*5\*c，卷积核的数量为24个，其中c为通道数。对于单通道灰度图，c=1，对于彩色图像，c=3或者c=4（RGB、HSL等情况下c=3，RGBA等情况下c=4）。
2. 对于第二层到第d-1层网络，使用卷积操作后，进行批量归一化（Batch Normalize）处理并使用ReLU激活。卷积操作的卷积核尺寸为5\*5\*24。
3. 对于最后一层网络，使用c个卷积核进行卷积操作，卷积核尺寸为5\*5\*24。该层网路的输出即为本网络的输出。

总之，该去噪方法有以下几个特点：

1. 利用噪声进行学习，并借助批量归一化（Batch Normalize）等方法对训练过程进行加速。
2. 该算法拥有较高的并行性，借助GPU加速，可以获得较高性能。
3. 网络的训练是图对图的，不需要过多的调参。神经网络中的隐含层通过充分的学习可以对噪声信息进行有效的区分。

图像的去噪往往要求输入尺寸与输出尺寸完全相同，这对于传统去噪方法较容易出现边缘去噪不净或者边缘画质损失等情况，TNRD去噪方法会在算法的每一个步骤前进行像素填充以实现输出尺寸与输入尺寸相同。而在本文所述方法中，对于图像的边缘，首先填充0，再进行卷积操作，确保输出图像与输入图像尺寸相同。最终结果证明，即使是简单的0填充，没有对输出结果产生不利影响 。

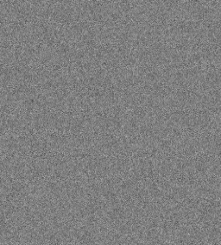
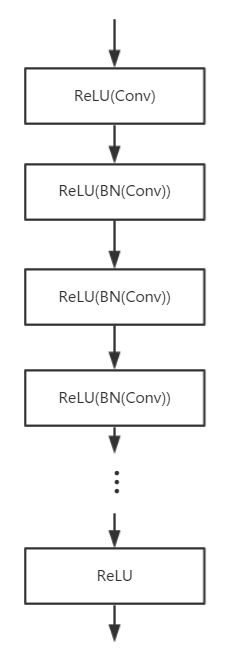


图2.1 本文所提出的去噪网络的结构

首先，我们将使用峰值信噪比（TSNR）来衡量算法的去噪效果的优劣。首先假设图像尺寸为m\*n，且该图象为单通道灰度图。在这里，我们有两个这样的图：p和p\_r，其中p为对噪声去噪处理后得到的图像，p\_r是不包含噪声的原图。那么我们首先定义他们的均方差：

 (3-4)

那么我们可以由以下公式获得PSNR值：

 (3-5)

其中，MAX为当前情况下允许的最大值。对于单色8bit图像，MAX=2^8-1=255。

同时，在该卷积网络中，批量归一化发挥了重要的作用，如图2.2所示，在本数据集测试条件下，首先固定卷积网络中基于梯度的优化算法为Adam，且使用同样的网络结构及相同的网络参数。相较于未使用批量归一化的情况下，使用批量归一化后，在训练初期略微落后，而后训练速度明显快于未使用批量归一化的情况，而且训练结果更优。

图2.2 是否使用归一化带来影响对比

图2.3 三种网络训练效果对比

如图2.2所示为在使用相同的网络结构，相同的训练参数，以及相同的梯度下降优化算法（Adam）下，训练后获得的信噪比PSNR的对比。图中我们可以看到，没有批量归一化的情况下，收敛速度在500次循环后变得十分缓慢，使得网络训练非常艰难。在没有ReLU激活操作的情况下，神经网络的训练速度同样不理想。当该卷积网络结合了ReLU激活函数的激活处理以及批量归一化处理后，它的收敛速度更快，且更稳定。

3.3 功能性拓展

以上所述卷积神经网络，仅针对特定特定尺寸且具有特定高斯噪声的图像进行噪声的区分。接下来，将对其进行拓展。

A．首先考虑尺寸问题。由3.2所述，神经网络每一层均由卷积操作、批量归一化操作、以及激活操作构成，并不包含全连接层。其中，批量归一化以及激活操作均对数据的尺寸没有要求。由1.5.3中卷积操作的原理，卷积操作对图像的尺寸没有固定的要求。且该网络不包含对尺寸敏感的全连接。这就意味着，对于同样噪声类型的图像，我们不需要重新训练不同的尺寸的网络亦可进行对不同尺寸图像的去噪处理，且避免使用图像切割等方法。

对此，修改网络如下：

x\_image = tf.reshape(x, [-1, im.size[0], im.size[1], 1])  
  
W\_conv1 = weight\_variable([5, 5, 1, 24]) *# 第一层卷积层*b\_conv1 = bias\_variable([24]) *# 第一层卷积层的偏置量*h\_conv1 = tf.nn.relu(conv2d(x\_image, W\_conv1) + b\_conv1)  
  
W\_conv2 = weight\_variable([5, 5, 24, 24]) *# 第二次卷积层*b\_conv2 = bias\_variable([24]) *# 第二层卷积层的偏置量*h\_conv2 = tf.nn.relu(batchnormalize(conv2d(h\_conv1, W\_conv2) + b\_conv2))  
  
W\_conv3 = weight\_variable([5, 5, 24, 24]) *# 第三次卷积层*b\_conv3 = bias\_variable([24]) *# 第二层卷积层的偏置量*h\_conv3 = tf.nn.relu(batchnormalize(conv2d(h\_conv2, W\_conv3) + b\_conv3))  
  
W\_conv4 = weight\_variable([5, 5, 24, 1]) *# 第四次卷积层*b\_conv4 = bias\_variable([1]) *# 第二层卷积层的偏置量*h\_conv4 = conv2d(h\_conv3, W\_conv4) + b\_conv4  
y = tf.reshape(h\_conv4, [1, -1])

其中，im.size为输入图像的尺寸，是一个二元数组。最终，实现了对不同尺寸图像的去噪处理，而不仅仅针对256\*256

图2.4 对于更大尺寸图像进行去噪

图2.4中，使用已有网络以及经过训练的模型对尺寸为512\*512的图像进行去噪处理，去噪效果与256\*256图像的去噪效果相当，去噪PSNR为26.03dB。

B．目前已进行处理的图像均为单通道灰度图

拓展2：多通道

拓展3：不同噪声

第四章 实验结果及分析

参考文献

附录