实验报告：利用 LeNet 卷积神经网络进行手写数字识别

王瑞哲　PB19071509

信息科学技术学院

摘　要：利用卷积神经网络中的 LeNet 网络设计方法，结合 Matlab 环境下 Matconvnet 工具箱，采用 mnist 数据库设计并训练一个深度神经网络，用于识别 0~9 共十个手写阿拉伯数字；并在此基础上测试训练好的网络结构的识别精度，并利用训练好的网络结构识别指定的手写数字。

关键词：卷积神经网络；LeNet 网络；Matlab；Matconvnet；手写数字识别

卷积神经网络（Convolutional Neural Networks, CNN）是一类包含卷积计算且具有深度结构的前馈神经网络，是深度学习的代表算法之一。卷积神经网络具有表征学习能力，能够按其阶层结构对输入信息进行平移不变分类，因此也被称为“平移不变人工神经网络。

LeNet 神经网络由深度学习三巨头之一的 Yan LeCun 提出，他同时也是卷积神经网络之父。LeNet 主要用来进行手写字符的识别与分类。LeNet 的实现确立了 CNN 的结构，现在神经网络中的许多内容在 LeNet 的网络结构中都能看到，例如卷积层，池化层，全连接层等。

1 实验目的与实验原理

1.1 实验目的

本次实验需要通过 Matlab 中的 MatConvNet 工具箱以及其中的 mnist 数据集，建立 LeNet 网络，实现 0-9 十个手写阿拉伯数字的识别，同时掌握卷积神经网络中的 LeNet 网络结构和训练过程。

1.2 LeNet 神经网络结构

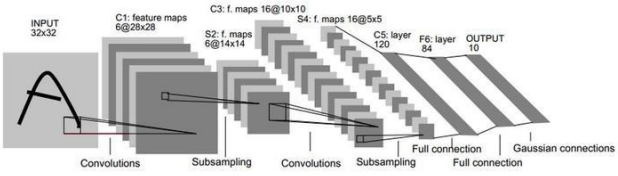

图 1　LeNet 网络结构简图

LeNet-5 这个网络虽然很小，但是它包含了深度学习的基本模块：卷积层，池化层，全连接层。是其他深度学习模型的基础。 LeNet-5 共有 7 层，不包含输入，每层都包含可训练参数；每个层有多个 Feature Map，每个 FeatureMap 通过一种卷积滤波器提取输入的一种特征，然后每个 FeatureMap 有多个神经元。

LeNet 由 C1 层（卷积层）、S2 层（池化层）、C3 层（卷积层）、S4 层（池化层）、C5 层（卷积层）、F6 层（全连接层）和输出层组成。

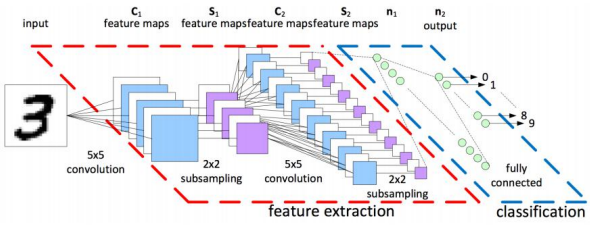

图 2　LeNet-5 网络识别数字过程

1.3 Matlab 中的 Mnist 数据集

Matlab 中的 MatConvNet 工具箱中内置了 mnist 数据集专门用于 0-9 手写数字识别。数据集共包含四个 IDX 格式文件，IDX 格式是一种用来存储向量与多维度矩阵的文件格式:

train-images-idx3-ubyte.gz:训练集图片

train-labels-idx1-ubyte.gz:对应的数字标签

t10k-images-idx3-ubyte.gz:测试集图片样本

t10k-labels-idx1-ubyte.gz:对应的数字标签

每个集合包含图片和标签两部分内容，图片为 28*28 点阵图；标签为 0-9 之间数字。这些文件本身并没有使用标准的图片格式储存。使用时需要进行解压和重构。

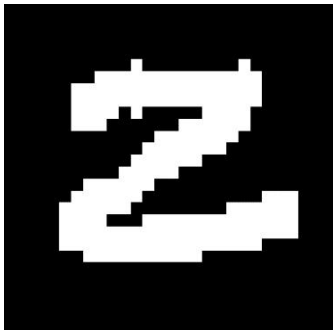

图 3　Mnist 数据集样本图片示例

每个样本图片是 28 像素*28 像素大小的灰度图片，如图 3 所示；这里定空白部分全部为 0，有笔迹部分根据颜色深浅可以在（0，1]之间取值，不过这个是一个简单的神经网络，有笔迹部分全部点为 1。图片是二维图片，把它转变成一维向量；28*28=784，图片有 784 个位置，每个位置或是 0 或是 1；这里我们不考虑数据上下左右等二维空间结构信息，从最左上角开始到最右接着下一行排列，这就把一张图片转换成了一个一串 0，1 组成一维向量(x)。所有图片都是这个方式转换展开。

根据训练模型的需要，工具箱中已备有配套函数 cnn_mnist.m 先对 imdb.mat 中的图片进行处理。通过打开 data 文件夹下的 mnist-baseline-simplenn 文件夹，看到一个含有 70000 张书写数字图片的数据集合 imdb.mat 文件，这就是手写数字图片的数据集合。本实验中，选用 imdb.mat 中的前 60000 张图片对搭建好的 LeNet 模型网络进行参数训练；选用 imdb.mat 中的后 10000 张图片进行验证测试。

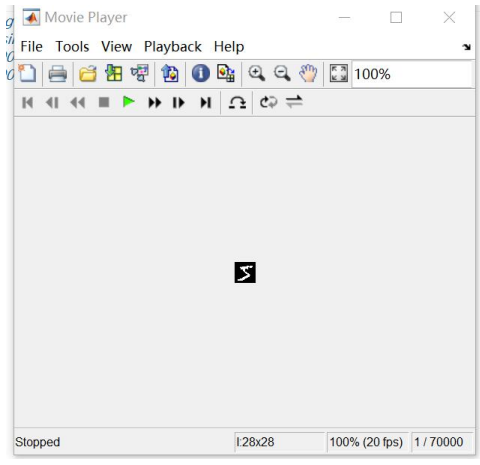

图 4　数据集 imdb.mat 里的手写数字图片

2 实验环境配置

2.1 matconvnet 工具箱路径配置

将所提供的压缩包解压，置入 matlab 文件目录中，为进一步实验做好准备。其中，查看 matlab 当前设置文件路径可以在命令行窗口输入指令”path”实现；在设置中，可以找到 matconvnet 文件夹所在位置，并将其添入 matlab 查找路径中即可。

2.2 Visual Studio 2015 安装及配置

安装 Visual Studio 2015 主要是为了配置随后 matlab 中的 mex 命令编译环境。因为 matconvnet 工具箱中编译函数 vl_compilenn.m 需要使用 VS 下的编译环境，这样才能进行下一步实验。尤其需要注意的还有，所选用的 Visual Studio 版本不能超过 matlab 的版本，否则 matlab 无法支持。当前 matlab 版本所支持的 Visual Studio 版本可以在 matlab 安装目录下...\bin\win64\mexopts 文件夹中查询到。如图 5，msvcpp2015.xml 文件即表示本机 matlab（2020b 版本）支持 2015 版本的 VS 下的 C++编译语法。

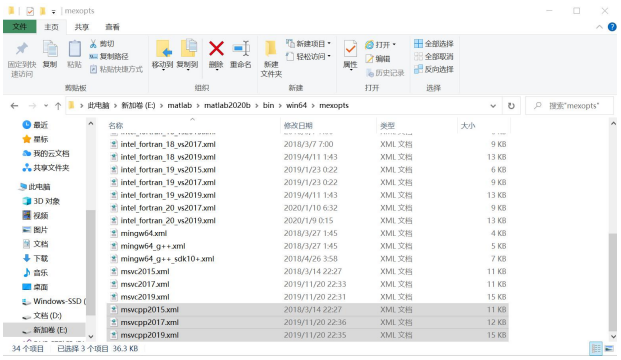

图 5　查询本机 matlab 适配 VS 版本

VS 安装完成后，回到 matlab 界面，在命令行中输入”mex -setup”指令，配置 mex 编译路径，其结果如图 6 所示。

图 6　met -setup 配置

2.3 matconvnet 工具箱编译与测试

将 Matlab 当前路径设置为 matconvnet 下文件路径，找到编译函数 vl_compilenn.m 并编译运行。其中，第 646 行位置涉及到 Microsoft C/C++编译器 cl.exe 的路径问题，由于 VS 不同版本的路径配置问题，例程所给出的路径：

cc.Location, 'VC', 'bin', 'amd64'

已经不再适用。此时编译，会产生如图 7 所示的以下错误信息：

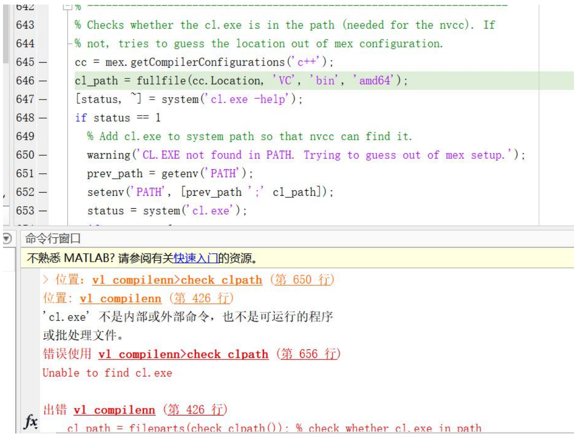

图 7 cl.exe 无法查找报错信息

此时查阅资料得知新版本下 VS 的 cl.exe 路径已经变成了：

“...\VC\Tools\MSVC\<版本>\bin\Hostx64\x64”

然后将对应位置修改即可。配置完成后，运行函数 vl_compilenn.m，得到结果如图 8 所示：

图 8 vl_compilenn.m 运行结果

最后输入 vl_testnn 测试程序，训练过程耗时 228.0708 秒，其过程如图 9 所示：

图 9 vl_testnn.m 测试过程

出现如图 10 界面说明 CPU 编译测试成功：

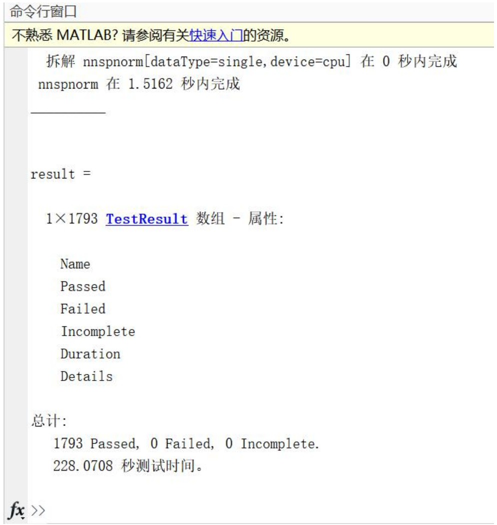

图 10 vl_testnn.m 测试结果

3 实验内容

3.1 训练网络

在 matconvnet 框架中提供了 mnist 数据库识别的例子，相关的文件主要在框架工程目录->examples->mnist 文件夹下，主要代码文件有：

.cnn_mnist.m——训练网络准备数据

.cnn_mnist_init.m——搭建网络的函数，初始化网络，设置网络参数

.cnn_mnist_experiments.m——网络实验

运行 cnn_mnist.m，等待训练结束。此时命令行窗口的显示如图 11 所示：

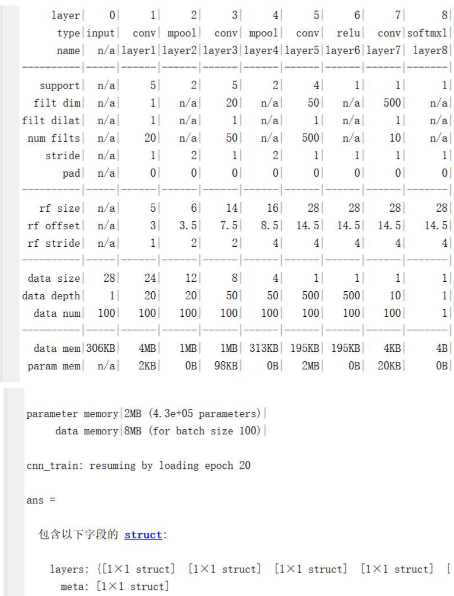

图 11 cnn_mnist.m 训练结果

在这个过程中，会生成一张如图 12 所示的一张图片。其中 objective 是总的损失函数；top1 是预测的 label 取最后概率向量里面最大的那一个作为预测结果，如果你的预测结果中概率最大的那个分类正确，则预测正确，否则预测错误。；top5 是最后概率向量最大的前五名中，只要出现了正确概率即为预测正确。否则预测错误。图 10 表明随着训练代数的增加，top1err 减小，错误减小，意味着精度增加。

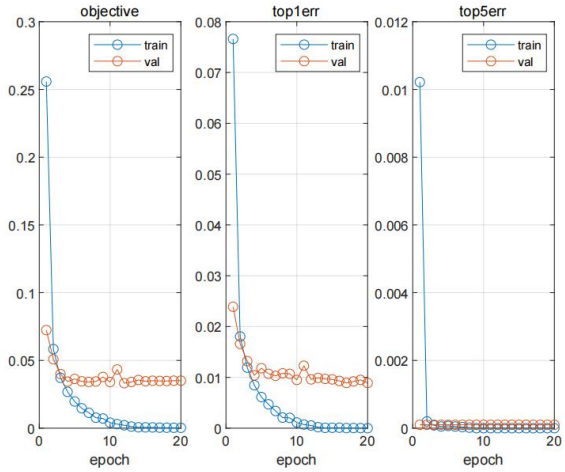

图 12 mnist 数据集训练过程损失函数结果

当 cnn_mnist.m 运行结束后，再打开 data 文件夹里的 mnist-baseline-simplenn 文件夹，会发现里面增加了 20 个 net-epoch-(1~20).mat，这些就是经过每一代训练后，获得的训练好的分类器。这里，我们共获得 20 个训练好的分类器，如图 13 所示：

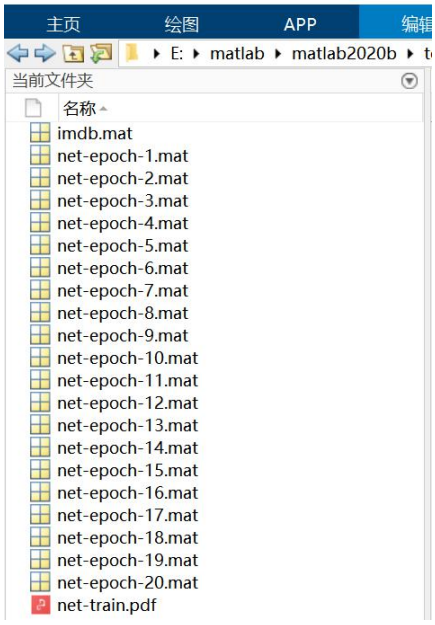

图 13 运行 cnn_mnist.m 获得的所有分类器

有了这些分类器，就可以用这些分类器来进行下一步的分类测试。

3.2 测试网络——比较测试精度大小

用若干个手写的数字图片，来测试一下刚刚训练出来的分类器的分类效果。从图 12 可以看出，训练的代数越多，错误越低，精度越高。这里为满足实验要求，**分别选取第 5 代、第 15 代和第 20 代**训练出的 net-epoch-xx.mat 分类器来进行测试。其中测试代码以实验指导书中提供的为准，截图展示如图 14 所示：

```
% 导入全体数据
load('E:\matlab\matlab2020b\toolbox\matconvnet-1.0-beta25\data\mnist-baseline-simplenn\imdb.mat');
% 挑选出测试集
test_index = find(images.set==3);
% 挑选出样本以及真实类别
test_data = images.data(:,:,:,test_index);
test_label = images.labels(test_index);
%导入模型文件
load('E:\matlab\matlab2020b\toolbox\matconvnet-1.0-beta25\data\mnist-baseline-simplenn\net-epoch-5.mat');
% 将最后一层改为 softmax （原始为softmaxloss，这是训练用）
net.layers{1, end}.type = 'softmax';
% net = vl_simplenn_tidy(net) ;
for i = 1:length(test_label)
    i
    im_ = test_data(:,:,:,i);
    im_ = im_ - images.data_mean;
    res = vl_simplenn(net, im_) ;
    scores = squeeze(gather(res(end).x)) ;
    [bestScore, best] = max(scores) ;
    pre(i) = best;
end
% 计算准确率
accurcy = length(find(pre==test_label))/length(test_label);
disp(['accurcy = ',num2str(accurcy*100),'%']);
```

图 14 test1.m 代码部分

例如第 5 代测试的结果展示如图 15 所示：

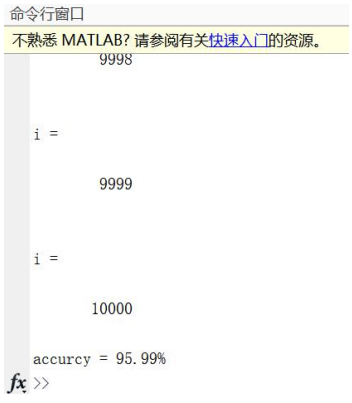

图 15 net-epoch-5.mat 精度测试结果

据此可列出训练对比表格如表 1 所示。据此可以看出，随着分类器代数的增加，测试精度会逐渐增加，但 5 次迭代和 20 次迭代的精度没有非常明显的差别。

表 1 精度测试结果

文件名	训练代数	测试精度
net-epoch-5.mat	5	95.99%
net-epoch-15.mat	15	96.67%
net-epoch-20.mat	20	97.05%

3.3 测试网络——手写数字实例识别结果

利用实验指导书附录中的 test2.m 文件，测试的样本是某一个指定的图片，目的是识别具体手写数字是几并显示其识别结果。其中 test2.m 代码截图展示如图 13 所示：

```
vl_compilenn.m | cnn_mnist.m | test1.m | test2.m
net.layers{1, end}.type = 'softmax';
% 设置默认参数
% net = vl_simplenn_tidy(net) ;
% 获取指定的图像，0-9手写数字的图片文件中已经给出，也可自行在EMNIST数据集中获取
im = imread('E:\matlab\matlab2020b\toolbox\matconvnet-master\data\mnist-baseline-simplenn\sample1.bmp');
% 灰度化
im = rgb2gray(im);
% 反转图像，训练图像为黑底白字，测试图像为白底黑字，所以需要反转一下
% im = 255-im;
im_ = single(im) ; % note: 255 range
% 输入默认大小为28*28
im_ = imresize(im_, [28 28]) ;
% 减去均值
im_ = im_ - images.data_mean;
% Run the CNN.
%返回一个res结构的输出网络层
res = vl_simplenn(net, im_);%vl_simplenn函数返回的是一个多维的数组，最后一个维度保存的是分别识别为0~9的概率值，最大
%展示分类结果
scores = squeeze(gather(res(end).x)) ;%得到图像属于各个分类的概率
[bestScore, best] = max(scores); %得到最大概率值，及其索引值
figure(1) ; clf ;
%imagesc(im) ;
imshow(im,[]);%显示图像
title(sprintf('the identified result: %d, score: %.3f',...
    best-1, bestScore)) ;%添加标题——第几,所属此的概率
```

图 15 test2.m 代码部分

最终得到的手写数字训练结果为：

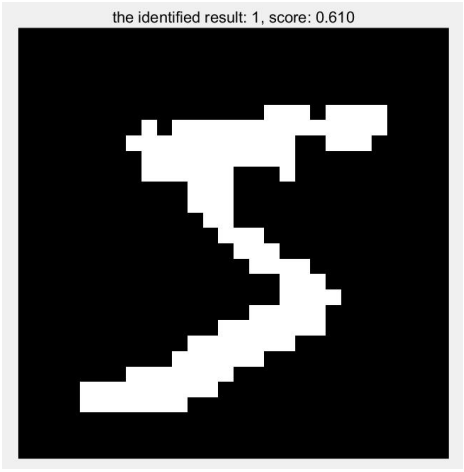

图 16 识别结果:1　识别准度:0.610

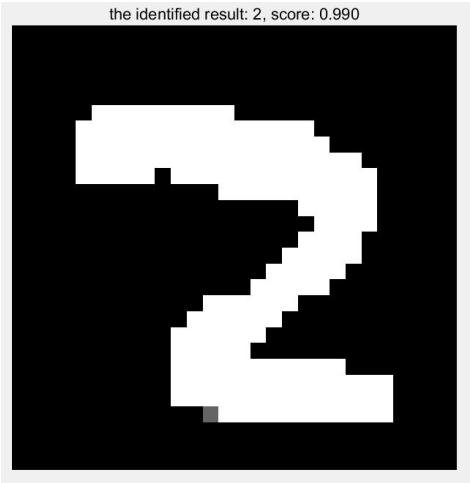

图 17 识别结果:2　识别准度:0.990

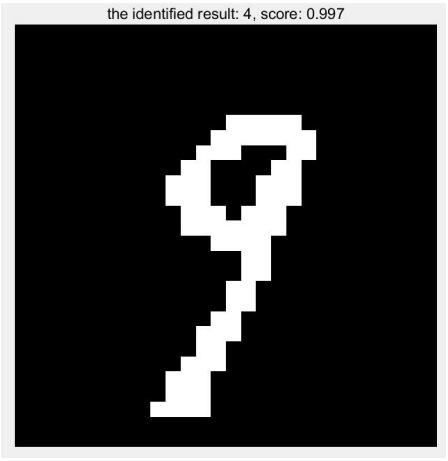

图 18 识别结果:4　识别准度:0.997

根据训练结果可以看出：第一张图片（手写 5）被识别成了 1，但网络给出的识别准度只有 0.610，说明网络自身也无法保证识别准度；第二张图片（手写 2）被识别成了 2，网络给出的识别准度有 0.990，说明网络对于较正的数字具有较高的识别精度；第三张图片（手写 9）被识别成了 4，但网络给出的识别准度仍有 0.997。观察图片，发现这个数字 9 确实与数字 4 有些许相似，而本次实验下的网络在这样的情况下识别出错也情有可原。

上面选取分析的三种典型情况初步表明，本次实验所训练的神经网络对于形状较为规正、较易辨别的手写数字有着较好的辨识能力；在一定的误差范围内，网络对于较为歪斜不规整的手写数字的识别精度仍可以被我们接受；而当手写数字中出现一些容易被误判的细节或笔触时，网络的识别效果就有较大概率出错。

4 实验总结及心得体会

本次实验借助 Matlab 中的 MatConvNet 工具箱以及其中的 mnist 数据集，建立 LeNet 网络，实现 0-9 十个手写阿拉伯数字的识别，并测试了该网络的识别精度，选取典型样本作为示例输入观察结果；同时也掌握了卷积神经网络中的 LeNet 网络结构和训练过程。

通过本次实验，我对于 CNN 卷积神经网络的基本原理有了进一步的了解，尤其是其中专门用于灰度图像识别的 LeNet 网络，并通过该网络的结构和具体实现过程，对卷积神经网络的实现过程有了更加清晰的认识。在软件实现方面，本次通过 Matlab 工具箱实现该实验，再次让我体会到 Matlab 工具箱的强大之处。但本次实验还有一较大困难在于环境

配置。由于本次实验所用到的 Matconvnet 工具箱并不是 matlab 自带的工具箱，需要通过外接接口配置编译路径；而在下载 Visual Studio 和 matlab 内配置编译路径的过程中，我又遇到了很多棘手的问题和许多莫名其妙的 Bug,这也锻炼了我检索网络信息，查阅数据手册，或采取其他解决方法的能力。在成功调试好编译环境之后，通过工具箱内系列函数的编译与调用，我又体会到了以项目化体系进行程序编写的重要之处——各文件夹相互依存，相互调用，每一步实验操作都或多或少地要依赖于上一步的实验结果完成等等。

总之，本次手写数字识别实验使我感触颇深，也引起了我更多的对于卷积神经网络和深度学习的兴趣。希望在之后的学习中，能够以本次实验为参考和指引，接触和了解到更多此方面的知识。

参考文献:

[1] 丛爽. 面向 MATLAB 工具箱的神经网络理论与应用（第三版）[M]. 合肥：中国科学技术大学出版社, 2009.4: 293-305.

[2] 神经网络与深度学习课程. 实验 4 指导书 [E]. 1-12.

[3] 神经网络与深度学习课程. MatConvNet 深度学习工具箱安装教程 [E]. 1-8.

[4] Matlab 工具箱 matconvnet 安装教程[J]. 2019.4: https://blog.csdn.net/weixin_43568722/article/details/89531953.

[5] matlab 编译 Matconvnet 的环境配置(CPU 和 GPU 版本)[J].2016.12:https://blog.csdn.net/jiejiaodebeiying/article/details/53954019.