

**本科课程设计报告**

设 计 名 称： 基于软注意力机制的STN模块在图像识别领域的应用研究

课 程 名 称： 机器学习与神经网络

课 程 编 号： 08060152

小 组 成 员： 郑晓鹏 温钊迪 罗晓峰 陈栋

学 院： 信息科学技术学院

专 业： 计算机科学与技术

指 导 教 师： 李 展

教 师 单 位： 信息科学技术学院

答 辩 日 期： 2019 年 6 月 11 日

**目录**

[摘要 1](#_Toc14205)

[一、 选题背景 2](#_Toc10032)

[二、 实验环境 4](#_Toc19250)

[三、 分工 4](#_Toc1360)

[四、 设计思路 4](#_Toc9783)

[五、 网络结构 5](#_Toc20654)

[六、 损失函数 7](#_Toc30970)

[七、 优化器 8](#_Toc23407)

[八、 训练过程 8](#_Toc27312)

[九、 结果分析 12](#_Toc32508)

[十、 实验总结 13](#_Toc2591)

[参考文献 13](#_Toc9745)

## 摘要

基于注意力机制而产生的**Spatial Transformer Network（STN）**能让网络模型显示地学习图像的各种不变性，从而能灵活的处理图像的仿射变换（如，平移、旋转、缩放等），借此网络模型可以更好地对图像进行特征提取。

本文尝试将**Spatial Transformer Network（STN）** 应用到图像识别中，通过在原有的神经网络中添加STN模块以提升图像识别的准确性，同时验证带STN的模型是否能够提取图像的关键特征。为了获得更加准确的结果，在本文中选取了两个训练集，通过对两个训练集分别训练两个模型来进行对照，以比较在添加了STN模型之后，模型的识别准确率是否能够提升。

实验结果表明，STN模块在某些数据集上的确能够提取出图片中的关键区域，并提高模型识别的准确率。

## 选题背景

近年来，卷积神经网络被广泛应用于图像识别领域，是计算机视觉的核心技术之一。而普通的CNN能够较为局限地学习平移不变性，以及隐式的学习旋转不变性，这样训练出的神经网络，对图片的全部特征其实是等价处理的。虽然神经网络学习到了图片的特征来进行分类，但是这些特征在神经网络“眼里”没有差异，神经网络并不会过多关注某个“区域”，但是人类注意力是会集中在图片的某个区域内，而其他的信息受关注度会相应降低。

视觉注意力机制是人类视觉所特有的大脑信号处理机制。人类视觉通过快速扫描全局图像，获得需要重点关注的目标区域，然后对这一区域投入更多的注意力资源，以获取更多所需要关注目标的细节信息，从而抑制其它无用信息。下图展示了人类在看到一副图像时是如何高效分配有限的注意力资源的，其中红色区域表明视觉系统更关注的目标。



图 1 注意力机制原理图

近几年来，深度学习与视觉注意力机制结合的研究工作越来越多，目前主要有两种注意力机制，分别是软注意力和强注意力，软注意力的关键点在于，这种注意力更关注区域或者通道，而且软注意力是可微的，因此可以通过神经网络算出梯度并且前向传播和后向反馈来学习得到注意力的权重；而强注意力是一个不可微的注意力，训练过程往往是通过增强学习来完成。

2015年NIPS上的一篇文章《Spatial Transformer Networks》提出对池化的改进改进方法，即通过添加一个STN模块让网络线性的学习图像转换不变性，甚至将其范围扩展到所有仿射变换乃至非放射变换；添加了STN模块之后，使得传统的卷积带有了[裁剪]、[平移]、[缩放]、[旋转]等特性，突出图像中的重点区域，提高模型的识别准确率。

在《Spatial Transformer Networks》提出的STN模块中，STN网络分为三个部分：Localisation Network-局部网络、Parameterised Sampling Grid-参数化网格采样、Differentiable Image Sampling-差分图像采样；其结构如下图所示：

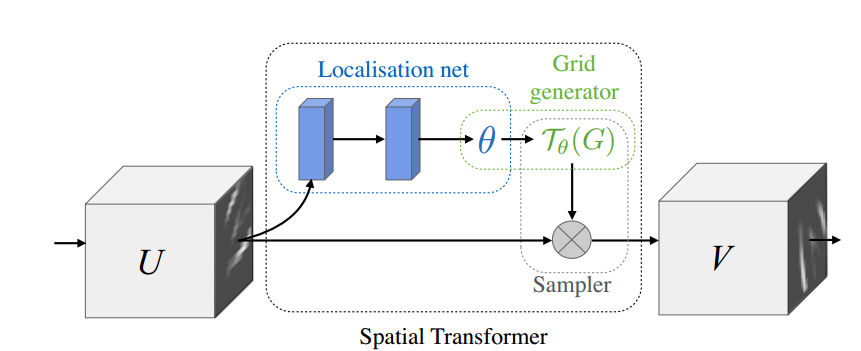


图 2 STN模块结构图

其中参数预测部分根据输入的图像经过损失函数不断更正定位参数，最终得到期望的仿射变换矩阵，用于下一步计算；在论文中提出：对图像的裁剪、平移、缩放、旋转等所有的这些操作，只需要六个参数[2 \* 3]控制就可以了；而坐标映射部分用于根据得到输出特征图的坐标点计算出对应的输入特征图的坐标点的位置，其计算公式如下：

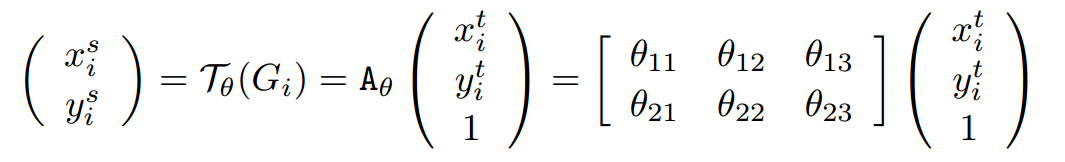


图 3 映射转换公式

对于计算出来的位置，还需要解决输出坐标为小数的问题，如果单纯用四舍五入是不能进行梯度下降来回传梯度的，因此还要在STN中坐标求解的可微性。

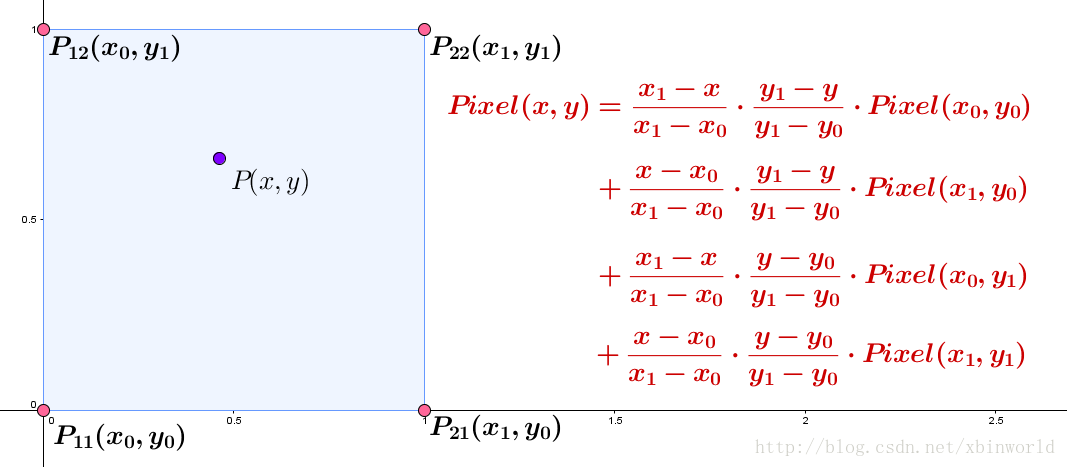


图 4 二次线性插值

基于上述理论基础，我们的研究目的为将STN模块应用到图像识别中，以测试该模块是否能够提升图像识别的准确率；即我们的工作在于将论文中的基于注意力机制的STN模块进行复现，将其添加到普通的神经网络中，并对其结果进行比较。

## 实验环境

设备配置：Ubuntu 18.04.2 LTS

开发语言：Anaconda3-5，Jupyter notebook

深度学习架构：keras,2.2.4,tensorflow

GPU：GeForce GTX 1080 Ti\*4

## 分工

郑晓鹏：模型构建、调参

温钊迪：模型构建、调参

罗晓峰：论文阅读、文献收集、文档编写

陈栋：PPT制作、文献收集、文档编写

## 设计思路

为了对STN模块进行检验，以测试其是否能够对模型的识别准确率有所提高，我们设计了一个对照试验，对于同一个数据集建立了两个模型，分别是原始的神经网络模型和添加了STN模块的模型，后一个模型是在原有的模型的基础上在最前面添加的STN模块，由于时间关系和技术限制，STN模块我们参考了网上的源代码，而原始的神经网络结构则自己搭建并在之后根据实验效果进行调参。对两个模型采用相同的数据集和相同迭代次数进行训练，再根据训练的结果的准确率进行比较，比较他们在测试集和训练集上的准确率的差别，以检测STN模块是否对模型的识别有帮助。为了能够更好的对STN模块进行测试，我们分别采取了两个测试集进行测试，第一个测试集是SVHN，该数据集中的图像来自真实世界的门牌号数字，每张图片中包含一组 '0-9' 的阿拉伯数字；第二个数据集是mnist手写体数字数据集的变形，即在原有基础上加入了噪声，并且图像像素为60\*60。以下为两个数据集的图像：

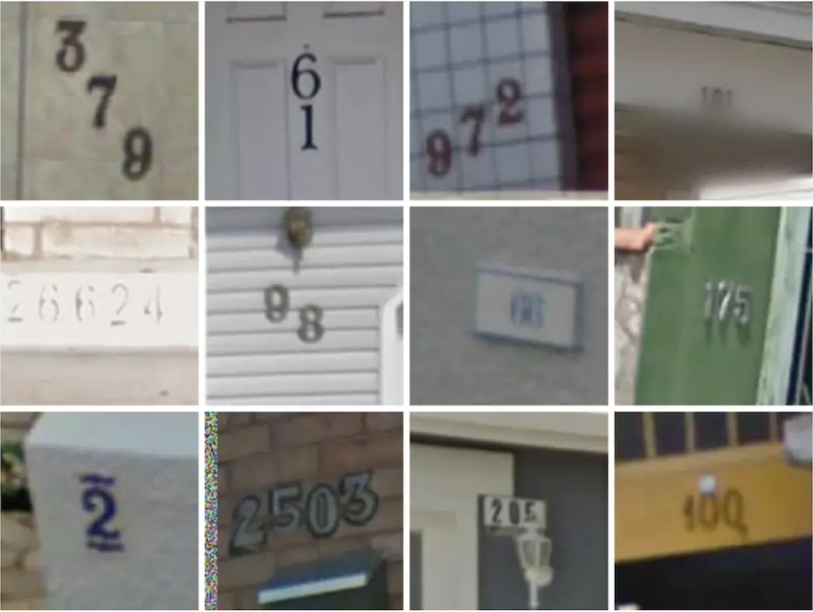


图 5 SVHN数据集

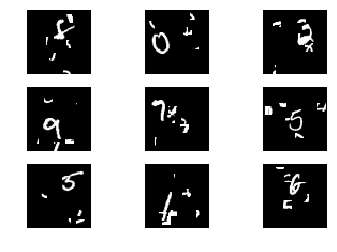


图 6 mnist手写体数据集

## 网络结构

在我们设计的网络结构中，添加了STN模块的网络结构分为两个：STN模块与卷积网络结构，dense\_2层及以上的都属于STN 模块，以下的为普通卷积网络，其网络结构具体如下：

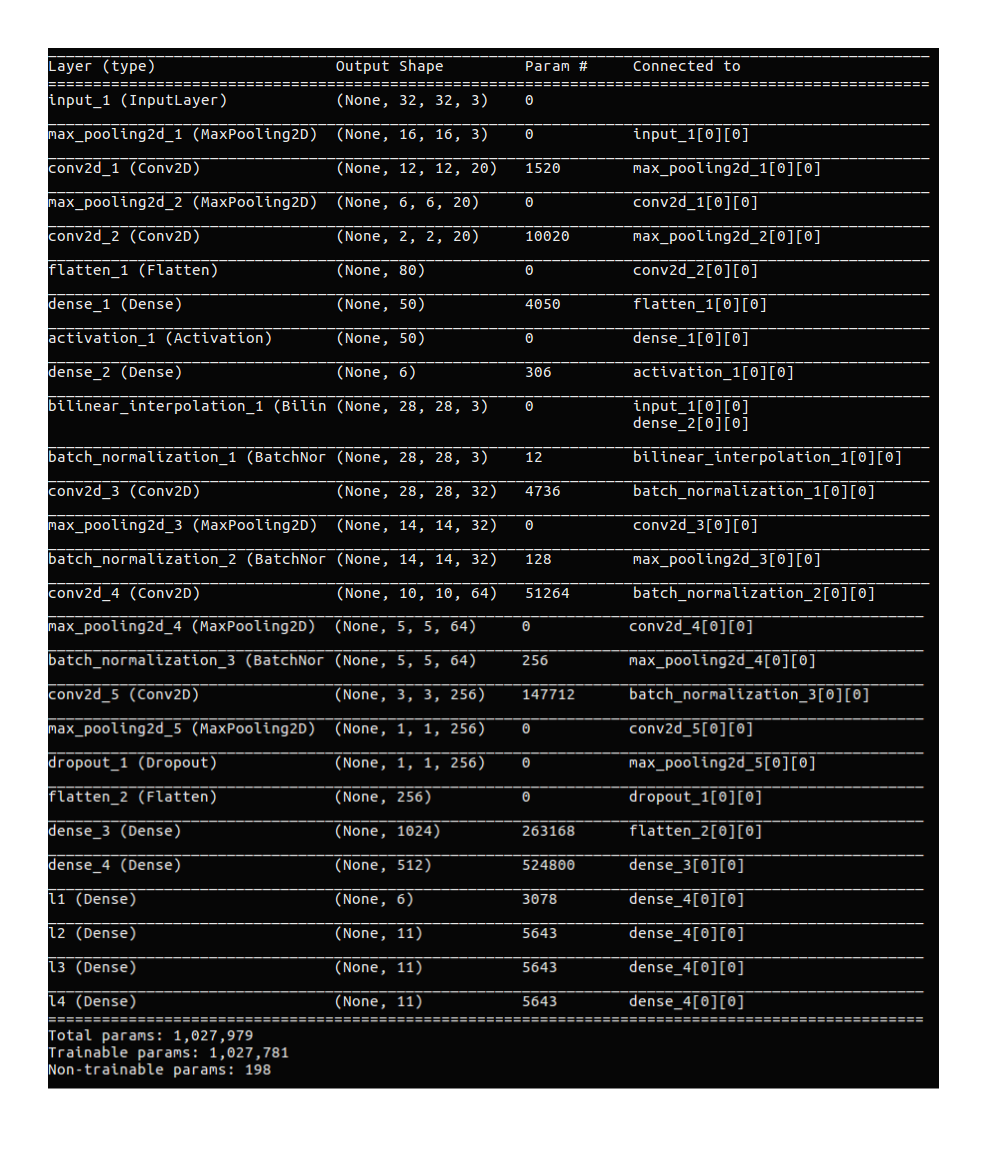


图 7 网络结构图

以下为一个更直观的网络结构，为了展示的协调化，图形的实际大小经过log处，还需要注意的是STN 模块输出的6x1 向量会和原来的Input 32x32x3 作为新的输入送入卷积网络中进行训练，具体如下：

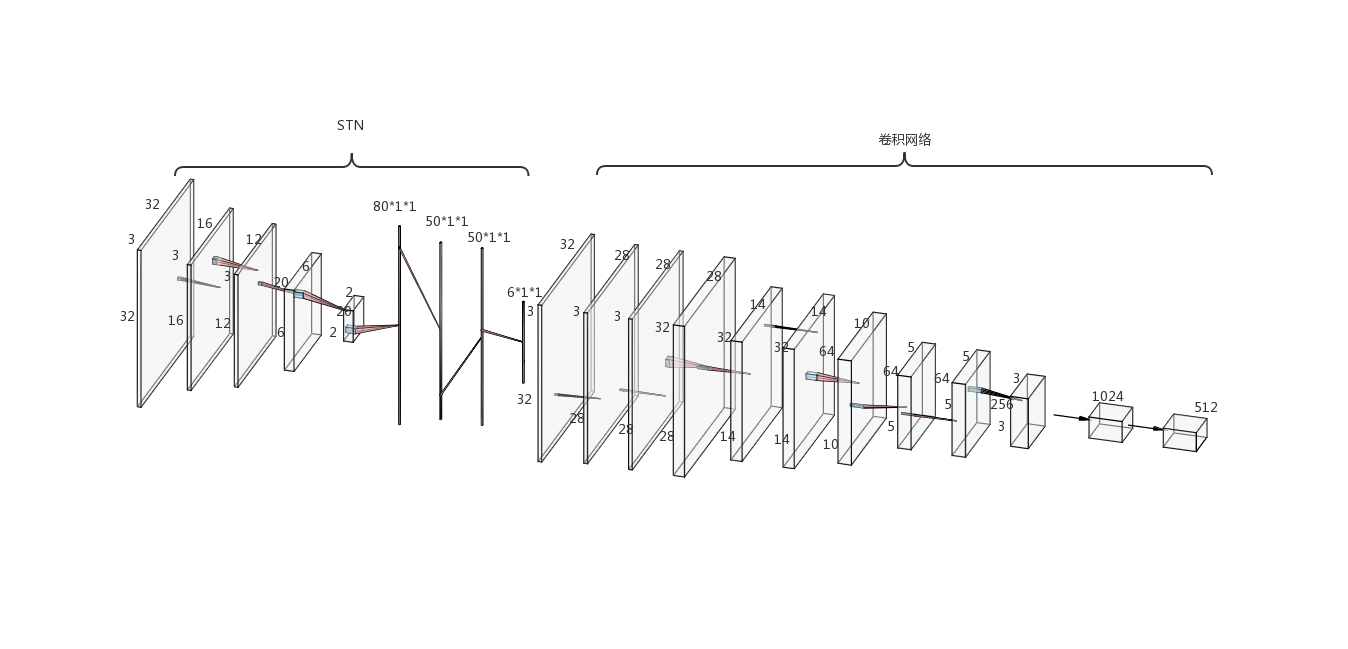


图 8 网络结构示意图

## 损失函数

Keras的categorical cross entropy函数求交叉熵。改交叉熵函数是在均方误差(MSE)的基础上进行改进的。

其中MSE的公式为

MSE == ①

=2 ②

但是因为sigmoid函数的性质，导致σ′(z)在z取大部分值时会很小，这样会使得w和b更新非常慢（因为η \* a \* σ′(z)这一项接近于0）。

为了克服这个不足，引入了categorical\_crossentropy（交叉熵损失函数）

①

=- ②

=- ③

….

=- ④

一般来说，如果最后一层接上softmax作为分类概率输出时，都会用categorical\_crossentropy作为损失函数，所以框架中会进行优化，对这两条公式的梯度合起来计算，发现合起来以后，loss计算就会超简单。

该函数对标签labels和卷积神经网络最后一层的输出logits计算softmax和一个包含交叉熵损失的张量(Tensor)。其中，函数的参数logits是指未归一化的对数概率。

经过reduce\_mean()求出张量的均值，平均值即为交叉熵。

输入：

输出：

## 优化器

本卷积神经网络选用梯度下降法更新权值，选用Adam优化算法。Adam算法属于自适应学习率优化算法，有利于提高训练效率。对比其他优化算法，Adam算法具有良好的稳定性，优化的效果好，而且超参数具有很好的解释性，通常无需调整或仅需微调。

Adam更新规则：

计算t时间步的梯度：

首先，计算梯度的指数移动平均数，初始化为0。类似于Momentum算法，综合考虑之前时间步的梯度动量。系数为指数衰减率，控制权重分配（动量与当前梯度），通常取接近于1的值，默认为0.9。

如果 mt 和 vt 被初始化为 0 向量，那它们就会向 0 偏置，所以做了偏差校正，通过计算偏差校正后的 mt 和 vt 来抵消这些偏差：

## 训练过程

首先对SVHN训练集进行训练，但在训练过程中意外发现，对于SVHN训练集，不带STN模块的神经网络模型的训练结果会比加入了STN模块的神经网络的训练结果要好，从下图可以看出：不带STN模块的神经网络模型的训练集准确率在几次迭代之后已经上升到了90%，而加入了STN模块之后的模型训练集准确率在多次迭代训练之后仍然停留在30%到40%，没有上升。

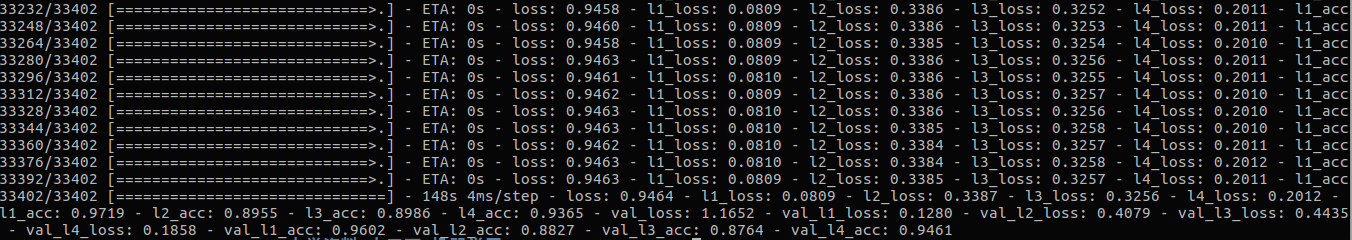


图 9 不带STN的CNN关于数据集svhn的训练结果

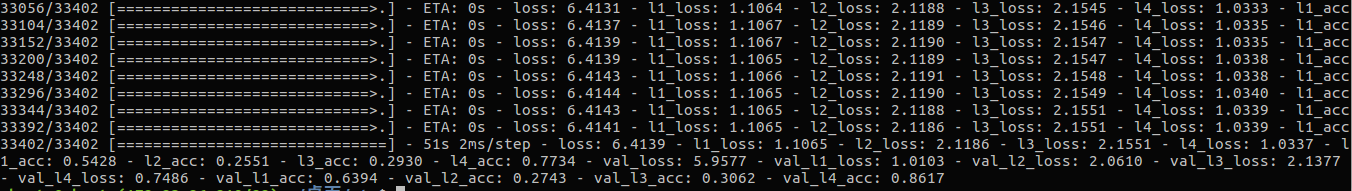


图 10 带STN的CNN关于数据集svhn的训练结果

对于该实验效果，我们后来进行了总结分析，发现了可能的原因是：在SVHN数据集中有每张图片上每个数字的定位点，由于每张图片的尺寸不同，原始图片输入到神经网络时会自动根据每个数字的定位点提取出数字的图片，因此不需要STN模块再对图像进行处理；并且，在添加了STN模块之后，由于STN模块会对图像进行转换，在这个过程中可能会改变图像的定位点，造成在输入神经网络的时候不能很好地根据定位点提取出数字区域的图像，因此在添加了STN模块之后模型的准确率才会有明显的下降。

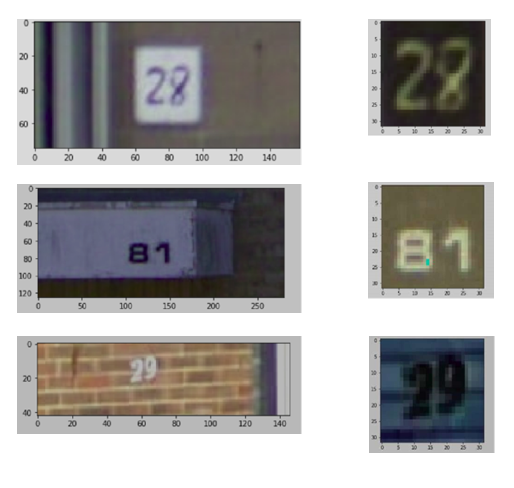


图 11 左图是原数据集中图像，右图是输入到神经网络的图像

之后，我们对下一个训练集mnist的神经网络模型进行训练，重新训练两个模型（一个带STN，一个不带STN）进行比对。

训练过程中，同样采取对照组实验，一组是带有STN的卷积网络，一组是不带STN的卷积网络。在控制其他变量一致的情况下，通过对模型训练过程中对训练集数据的准确度acc 和对验证集数据的准确度val\_acc 进行观察，对比哪个模型的识别效果更好。

训练过程如下：

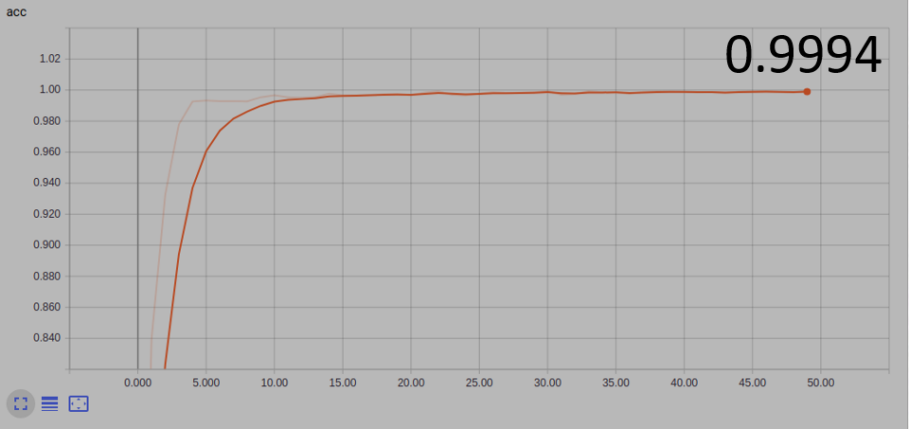


图 12 不带STN的模型的准确度acc随epoch的变化

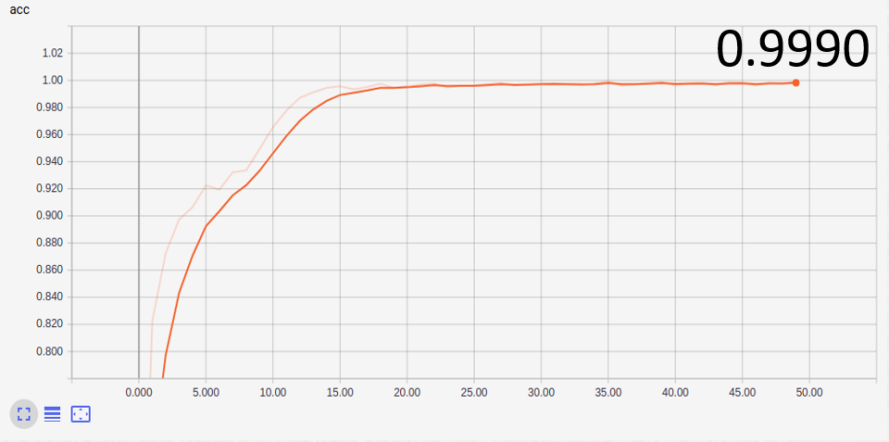


图 13 带STN的模型的准确度acc随epoch的变化

从训练数据来看，不带STN的模型和带STN的模型二者的精确度几乎没有什么差别。但这并不意味着，他们性能上没有差。因为可能会存在过拟合的情况，因此还需要对比，两个模型对于验证集的精确度如何。

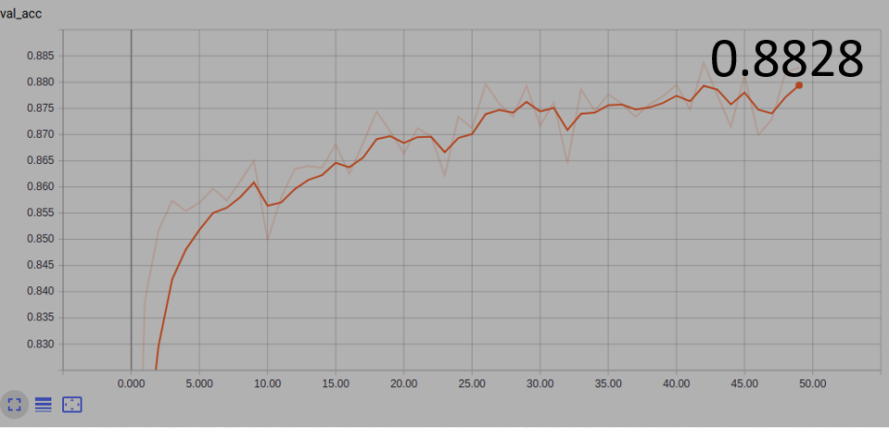


图 14 不带STN的模型对于验证集的准确度val\_acc随epoch的变化

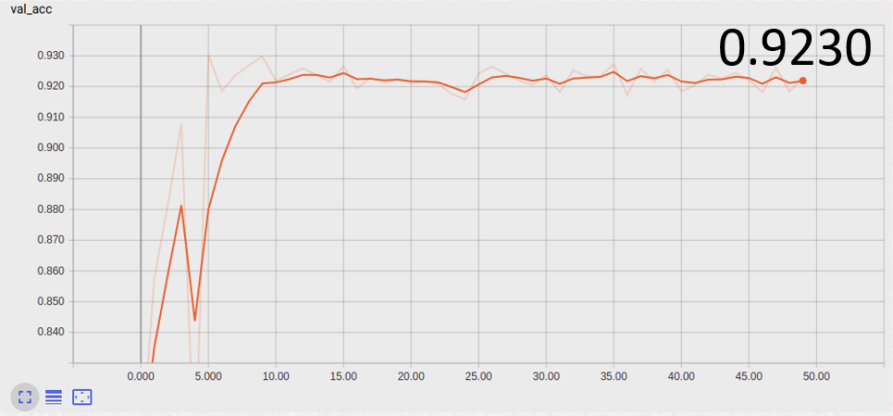


图 15 带STN的模型对于验证集的准确度val\_acc随epoch的变化

从验证集上来看，不添加STN模块的神经网络模型的在测试集上的准确率只有88%，而在添加了STN模块之后的模型在测试集上的识别准确率达到了92%，明显要比原始的网络要更佳。为了能更方便地比较，我们把对同一变量的不同模型的观测情况放到同一个图进行对比，两个模型在训练集上的准确率如下图所示：

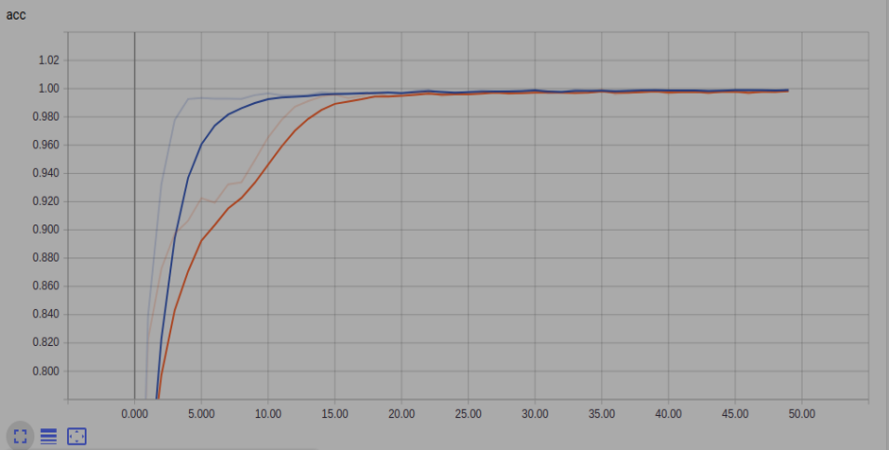


图 16 STN与非STN关于acc的变化，黄色的为STN,蓝色的为非STN

从上图可以看到，两个模型在训练集上的准确率基本相同，但添加了STN模块之后，模型的训练时间明显要加长，这是因为在添加了STN模块之后，模型的结构变得更加复杂，因此需要的训练时间也变得更长。

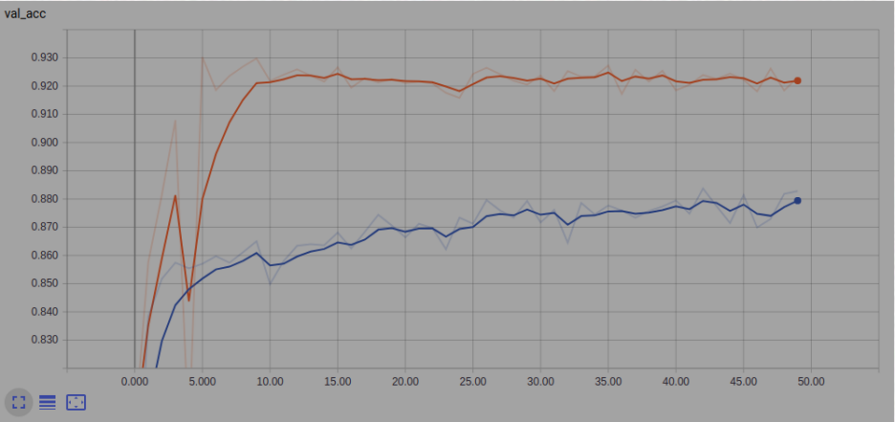


图 17 STN与非STN关于val\_acc的变化，黄色的为STN,蓝色的为非STN

通过上图对比来看，带STN的模型在验证集的准确率最终保持在0.9230,而不带STN的模型则在验证集上的准确率则最终停在了0.8828，由此可见排除过拟合的情况，带STN的模型在识别准确度上是优于不带STN的模型的，这也说明了在添加了STN模块之后，模型的泛化能力提高了。

## 结果分析

根据我们的实验结果，基于软注意力机制的STN模块能让模型对图形的仿射变换如平移、缩放、旋转等变换进行学习，让模型能更好地把握图像变换不变性，由上述的训练过程中就可以观察得知，STN模块通过加强模型对图像关键特征的提取能力，从而提升模型对验证集数据的识别准确度，为了能更好的展示STN模块帮助模型提取图像关键特征的效果，我们单独地进行了STN对图像关键特征的提取，如下图：

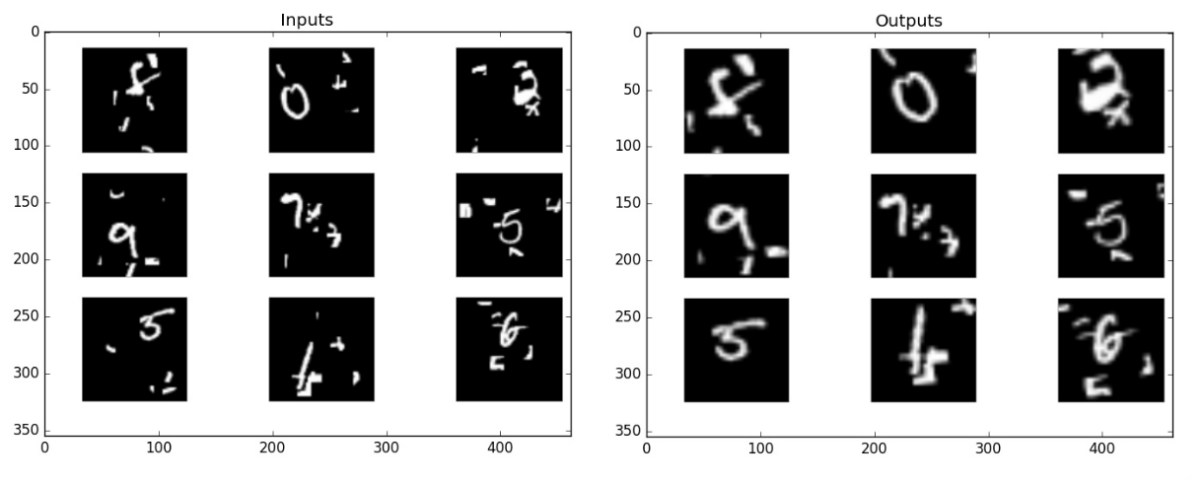


图 18 STN模块处理结果

左图为手写字体变形数字集的输入，右图为对STN模块进行了50次迭代之后对输入图像的关键特征的提取结果，我们可以直观的看到，STN模块对有数字那一块的图像进行了在视野中放大和居中的提取处理。

## 实验总结

先来谈谈收获，最大的收获是学习到了课上没有提及过的新知识——注意力机制，相较于单纯池化的特征抽取，注意力机制对关键特征有着更高的敏感度，而且能够应对一些复杂的变换，例如旋转、平移、缩放等。

其次是实验过程中的一些感想:

（1）调参也要讲究方法，可以通过交叉验证的方法来判断模型是属于过拟合还是属于欠拟合，从而针对性地堆超参数进行调整。

（2）文献查阅和多方讨论非常重要，有时候一个人看文献或者博客的时候，会很容易钻牛角尖，将问题拿出来讨论，借助他人思考问题的角度，之前的问题或许就会迎刃而解。

（3）现在的机器学习的框架对新手的很友好，认真阅读理解使用文档能够帮助初学者快速的完成模型的搭建、运行、调试。当然这一切也都是要建立在能理解框架底层运行的原理的基础之上的，如果只是一味的调用框架的API，养成依赖性，反而不利于我们把握要研究的问题的本质。

（4）学会训练过程的可视化技巧有利于帮助我们在训练模型的过程中，及时地针对可能出现的问题进行预防和完善。

（5）有条件的话，可以尝试利用GPU来加速模型的训练，以节约模型的开发成本，或者可以利用网络上一些开放的平台进行模型的训练，这样能够节约我们学习时间的成本。

值得改进的地方是：STN 还有对于经过变换（例如平移、旋转、缩放）的图像的特征提取相较于传统的池化层具有更好的效果，因此还可以设置一组对照实验，用经过变换数据集去测试有STN和无STN的网络，比较哪个效果更好。

# 参考文献

[1]Max Jaderberg, Karen Simonyan, Andrew Zisserman, Koray Kavukcuoglu. Spatial Transformer Networks

[2]https://blog.csdn.net/qq\_39422642/article/details/78870629

[3]https://zhuanlan.zhihu.com/p/37110107

[4]<https://blog.csdn.net/xbinworld/article/details/69049680>

[5]<https://blog.csdn.net/zb1165048017/article/details/90083552>

[6]https://github.com/oarriaga/STN.keras

[7]https://github.com/kevinzakka/spatial-transformer-network

[8]https://zhuanlan.zhihu.com/p/42692080

[9]https://blog.csdn.net/u011961856/article/details/77920970