深 圳 大 学

本 科 毕 业 论 文（设计）

题　　目: 基于深度学习的无分割手写字符的识别

姓　　名: 杜雨辰

专　　业: 光电信息工程

学　　院: 光电工程学院

学　　号: 2013800290

指导教师: 雷耀虎

职　　称: 讲师

2018 年 5 月 01 日

深圳大学本科毕业论文（设计）诚信声明

本人郑重声明：所呈交的毕业论文（设计），题目《基于深度学习的无分割手写字符的识别》是本人在指导教师的指导下，独立进行研究工作所取得的成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式注明。除此之外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。本人完全意识到本声明的法律结果。

毕业论文（设计）作者签名：

日期：　　年　月　日

目录

【摘要】 序列数据的识别有着广泛的应用，比如语音识别，文本的光学识别，机器翻译等。在机器学习领域，印刷体的OCR识别已经基本解决了，单个手写体的识别率也能轻易达到９５％以上，而整段手写体的识别仍然是较为热门的研究领域。

本文探索了近年用来实现手写文本识别的一种新技术。该识别过程中不需要把文本分割为单个字符，而且对于语言没有任何限制，除了训练数据不需要额外的语言库。这种方法来自机器学习中的深度学习领域，使用迭代神经网络和一种特殊的损失函数来实现，该网络会结合上下文信息辅助识别。深度学习对于图像类型数据的处理非常有效，而图像和视频等非结构化数据是网络上数量最多的数据。模型在Google的Tensorflow框架中实现，并在最后给出了和学界近期新方法的比较和评估。

【关键词】手写字符识别，迭代神经网络，深度学习

总体问题少很多，但还有些格式问题，这次你不用急着发回给我，仔细对照模板，修改格式即可。

全文用两端对齐

1 引言

人工智能是一个比较宽泛的概念，指能够模仿人智能的某些计算机算法。现代的人工智能有悠久的历史，起始与计算机产生相同的上世纪 50 年代。标志性的事件是 1956 年 Dartmouth 大学的一个会议。会议上 John McCarthy(MIT)，Marvin Minsky 等人的演讲拉开了人工智能研究的序幕。本文研究的是人工智能中的一种实用方法--深度人工神经网络。

1.1.1 人工智能的学派和方法

错两格

人工智能没有一个统一的方法或模式。对此，不同的研究者有很大的分歧。一般的方法有以下几种：

控制论和大脑模拟：主要使用控制论，神经生物学，信息学的方法去研究。

符号学派：把人工智能问题归结为符号的操纵变化。上世纪 60 年代，符号学派在模拟特定领域的抽象推理取得很大成功，而神经网络等方法则遭到冷遇。

认知模拟：通过认知科学和心理学去模仿人来实现智能。

逻辑和知识推理：通过把知识输入计算机，结合逻辑推理实现专家系统，但是对于简单的问题，该方法都需要大量知识。

统计学：使用数学方法去使得人工智能可以被验证和测量，这为多个学派提供了统一的数学工具，也是现在人工智能成功应用的基础。

1.1.2 神经网络、机器学习和深度学习

机器学习算法又称为统计学习，是 Arthur Samuel 在 1959 年这样称呼该领域后得到广泛传播的，从模式匹配和计算学习理论发展而来。机器学习的目标是探索一系列算法，这些算法通过某种优化方法对数据集进行优化，从而避免了之前普通的“静态”计算机算法显式地固定了计算模式。机器学习一般应用在难以给出普通算法的领域，比如自然语言的处理等方面。其方法一般是运用线性或非线性统计学的方法，比如线性回归，逻辑回归，支持向量机以及神经网络等模型，针对特定数据集，运用梯度下降等优化算法，确定模型的参数，从而得到目标函数的算法。而理论主要来自统计学、数值优化和微积分等。

机器学习可以分成下面几种类别：

监督学习：给定大量标定的训练数据集（有输入 X 和输出 Y），通过训练得到一个目标函数 X→Y，或者是 P(Y|X) （即输入 X，给出 Y 的概率分布）。本文的方法和主题都属于监督学习。

无监督学习：训练集没有标注（即只有输入 X，而没有给出 Y）。由于没有给出目标，其任务是对 X 进行某些归类，比如聚类算法。

增强学习：是监督学习的变体，特点是需要的训练数据量大大减少，只需要很少数据就能得到目标函数。其中一种方法是由给定的少量数据生成大量数据再进行监督学习。

机器学习的算法：

关联分析：根据已有的数据找出规律，比如从医学数据中学习疾病之间的关系。

分类：方法有逻辑回归（线性回归的分立版本）和支持向量机。

人工神经网络：主要是深度学习。

贝叶斯方法：统计方法的一种。

表示学习：目标是学习数据的更好表示，主要是无监督学习。常用的算法有 principal components analysis 和聚类算法。

深度学习

深度学习[1]属于机器学习中的非线性统计方法，而机器学习属于统计学派。深度学习不只包括层数多的神经网络，还有深度置信网络（deep belief networks）等。深度学习的定义是：

1.使用多层的非线性处理单元用来提取和变换目标属性，每一层的输入是上层单元。

2.把任务逐级分层来学习其表示。

深度学习中比较流行的方法是人工神经网络（下面简称神经网络），属于“连接主义”学派中的一种方法，该学派相信复杂的行为会从简单单元的连接和交互中产生。人工神经网络基于“神经”单元构建，该单元一般是一个非线性变换函数，对输入的N维向量进行变换。多个单元进行组合，就可以构成深度卷积网络或迭代网络。

神经网络的理论基础是universal approximation theorem[2]，该定理证明了2层以上的神经网络有能力拟合任何函数。另一个重要的特性是收敛性，即网络是否能够收敛到目标函数，以及多快（需要多少数据）收敛。对于简单的网络有数学证明，但是复杂网络仍然靠实践确定。

深度学习的缺点

对深度学习的批评主要是其缺乏一个数学上严谨的理论模型，比如收敛速率等，所以很多性质都主要靠实践检验。另一个问题是深度学习无法归纳出逻辑推理的规则，即模型无法解释其为什么有效。

1.2 本论文的研究内容与研究意义

作为本科毕业论文，本文前面给出了少量深度学习的理论基础，后面主要侧重是在计算机上的实现和测试，并给出最终效果的评估以及和其他论文的比较。实验中会采用多种神经网络结构并给出分析。第一章讲述了人工智能领域的大致轮廓和背景，第二章给出了人工智能中神经网络模型的理论基础和代码实现，第三章开始训练优化模型的参数并分析结果，并和其他论文的模型和实现作比较。第四章总结了模型的优缺点并指出了其他研究方向。

本文的意义是利用已有框架来实现机器学习领域较新的算法，给出实际效果的评估和与其他学界先进方法的比较，形成定性定量认识，客观评估模型的优/缺点并持续改进。

2 基于神经网络模型的实现与优化

2.1 理论基础

2.1.1 多层神经网络可以模拟任何函数

最简单的 2 分类单层神经网络的表达式是：

公式居中，编号右对齐

$$f(x)_{w,b} = a(W \cdot X + b) \tag{1}$$

这是一个简单的线性模型，f(x)是要确定的预测函数，矩阵 w 和向量 b 是待确定的参数，a(x)是激活函数，当其参数超过一定值输出 1,否则输出 0。当通过训练样本确定了矩阵 W 和向量 b 的值后，该网络能够区分任何线性可分的输入向量，即确定了一个超平面。因此，单层网络的拟合能力和普通的多维线性回归是一样的。

不过对于多层（含有一个隐藏层以上的）前向（非迭代）神经网络（图 1），Universal approximation theorem[2-3] 定理表明在理论它上可以拟合任意连续函数(对函数的性质有一定限制)。其表达式是（由于太长，只给出第一项，后面项类似）：

$$f(x)_{w,b} = a(W_1(a(W_1 \cdot X + b), a(W_2 \cdot X + b)) + b) + \ldots \tag{2}$$

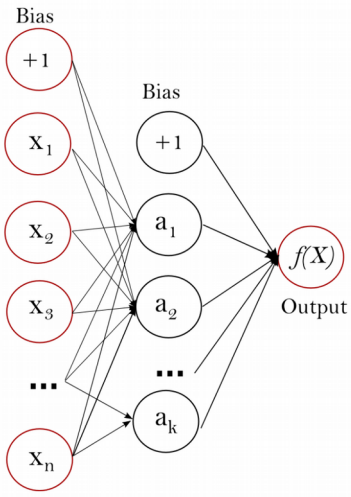


图 1. 多层前向神经网络

这也不足为奇，因为Tylor 展开式和Fourier分解同样可以拟合任何函数，不同之处后两者是展开已知函数，而神经网络是根据某些点的值（训练数据集）去尝试生成目标函数。但是，多层神经网络的训练是个难题，因此直到Back propagation算法出来[4]，再加上计算机的能力大幅度提升后，多层网络的训练才成为可能。

一般的全连接多层神经网络参数众多，能够拟合的函数过于广泛而未加限制，因此虽然能够在训练集中收敛，但是在测试集上却表现不佳。近年取得巨大成功的卷积网络和迭代网络的成功原因是限制了全连接多层网络的能力。两者都相当于是在空间上限制了神经单元之间的连接，前者是让一层的神经网络只和另一层神经网络的附近神经单元连接，而不是所有单元连接。而后者是通过迭代机制实现了神经网络的参数共享，迫使网络需要考虑序列之间的联系从而优化模型使得损失函数最小。感性地说，卷积网络主要用于非序列的识别，通过把图像的块状特征一层层提取出来，比如前面层能够识别点，线，而后面层则负责识别面，体等更抽象的特征。迭代网络是本文应用的技术，由于其对于序列（时间，或任何有序的结构，如文本等）的“记忆”能力，主要用来做序列的识别，能够把背景考虑进去。结合CTC[5]等特殊的损失函数，可以实现整体，无分割的识别。

2.1.2 优化算法：梯度下降和反向传播

反向传播（Back propagation）[4]是一种与优化算法（如梯度下降）结合使用训练人工神经网络的方法。该方法对网络中所有参数计算损失函数的梯度，然后来更新神经网络的参数（比如权值等），逐步找到损失函数的最值点，从而实现神经网络模型的训练优化。

反向传播要求对于每个输入值，给出已知的想得到的输出，来计算损失函数梯度。因此，它通常用来实现监督学习，但是有时它也用在一些无监督网络（如自动编码器）中。它是多层前馈网络的Delta规则的推广，通过链式法则的运用，对每次的迭代计算梯度。反向传播算法要求人工神经元的激励函数可微。

反向传播算法主要有两个阶段:激励传播与权重更新。

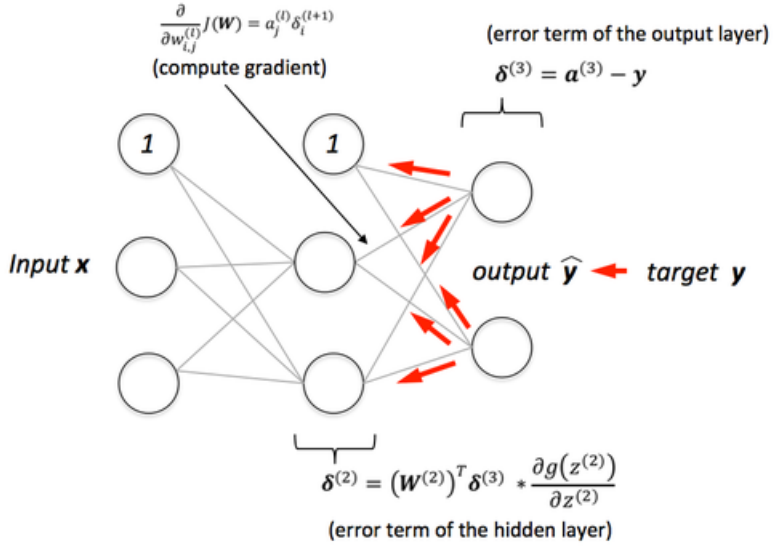


图 2. 反向传播算法示意图。网络是 3 层的，误差由后向前传播

第 1 阶段：激励传播

每次迭代中的传播环节包含两步：

前向传播：将训练输入送入网络以获得输出。（网络预先运用随机分布等方式初始化了参数，使得其能够在初始时输出）；

反向传播：将输出同训练输入对应的目标输出求差，从而获得隐层和输出层的响应误差。

第 2 阶段：权重更新

对于每个突触上的权重，按照以下步骤进行更新：

将输入激励和响应误差相乘，从而获得权重的梯度；

将这个梯度乘上一个比例并取反后加到权重上。

这个比例（百分比）将会影响到训练过程的速度和效果，称为学习速度。梯度的方向指明了误差扩大的方向，而学习速度过高会使网络无法收敛。

实际运用中，第 1 和第 2 阶段进行反复循环迭代，直到网络的对训练集的输入输出达到一定范围内，比如经验认为误差达到 0.1%内可以停止训练。

2.1.3 RNN 循环神经网络的结构和优化

迭代神经网络 RNN(recurrent neural network) 的公式并不复杂：

$$\boldsymbol{s}^{(t)} = f(\boldsymbol{s}^{(t-1)}, \boldsymbol{x}^{(t)}; \boldsymbol{\theta}) \tag{3}$$

其中 S 为状态，X 为输入向量， θ 为待确定的参数。

和普通前向神经网络不同的是，其数学形式是一个迭代函数。通过迭代，t 时刻的函数的参数实际包含了 t-1 时刻之前的所有输入和函数值。另一个不同是通过把函数展开，其输入可以是任意个输入向量 X，而其权重 W 实际上可以看作展开图（图 2）中的共享参数。

$$s^3 = f(s^2, x^2, \theta) = f(f(s^1, x^1, \theta), x^2, \theta) = f(f(f(s^0, x^0, \theta), x^1, \theta), x^2, \theta) \tag{4}$$

迭代神经网络用图表示如下：这里节点表示变换函数，箭头表示向量，x 为输入，y 为输出，之间的为隐藏单元。

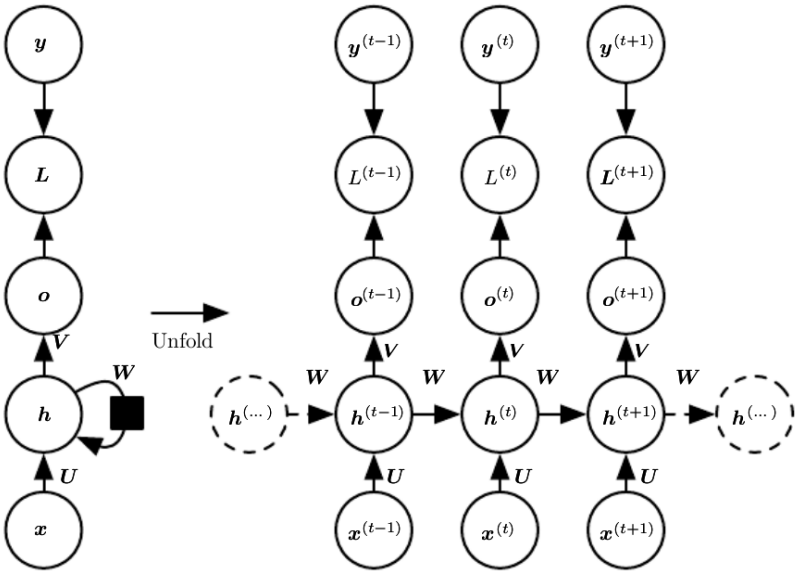


图 3. 迭代神经网络的图表示[1]

2.1.4 CTC 损失函数

对于序列数据进行识别是现实中经常遇到的需求，比如语音识别任务，是把一特定采样率的音频序列识别为文字，还有图像文本的识别，是把图像形式的文本识别为可以进行编辑的文本化数据。序列数据的识别一般需要分割，还有就是序列里边元素具有依赖关系，比如 you 后面跟着 are 的几率远远大于 you 后面跟着 is 的几率。后者的背景依赖性可以通过最近的迭代神经网络解决，但是即使使用迭代网络仍然需要序列的分割。之前的方法是使用 HMM(隐马尔可夫模型，hidden markov chain)来实现无分割识别，但是 HMM 需要把大量该语言的背景信息考虑进网络结构里边，使得整个构造非常复杂。

CTC[5]的最重要特点是不需要输入和输出的序列对齐，即不需要把文字或语音序列预先分割。对于印刷体，字符的分割比较容易，但是手写体的分割就非常困难了，而且连笔画的存在会使得分割过程损坏单个字符。再一个是空格字符的处理。预先分割使得空格的识别非常困难，因为手写字符的间距也有很大随意性。

CTC 的工作过程大致分为 2 步：

1：对于输入序列 X，建立对于所有输出路径 π 的概率分布：

$$p(\pi|\mathbf{x}) = \prod_{t=1}^{T} y_{\pi_t}^t, \ \forall \pi \in L'^T \tag{5}$$

2：对于所有输出序列 l，建立其概率分布为所有路径之和：

$$p(\mathbf{l}|\mathbf{x}) = \sum_{\pi \in \mathcal{B}^{-1}(\mathbf{l})} p(\pi|\mathbf{x}). \tag{6}$$

只要输出的序列长度小于输入序列长度，就可以使用 CTC。对于本文，输入是一列像素，输出是一句话，满足该要求，因此可以使用 CTC 。

CTC 的实际算法很复杂，上面给出了 CTC 的大概原理，目标是计算出 p(l|x)。接下来需要给出具体的算法。由于 p(l|x)是求所有路径的概率和，所以粗看似乎是在算法上不可解的（数量太多），但是利用动态规划（类似于 HMM）算法解决。

$$p(\mathbf{l}|\mathbf{x}) = \alpha_t(|\mathbf{l}'|) + \alpha_t(|\mathbf{l}'|-1) \tag{7}$$

其中 α T(s)（α T(s)=α t(s)）的含义简单的说就是 t 时刻前 s 个符号的概率，公式如下：

$$\alpha_t(s) \overset{\text{def}}{=} \sum_{\substack{\pi \in N^T: \\ \mathcal{B}(\pi_{1:t})=\mathbf{l}_{1:s}}} \prod_{t'=1}^{t} y_{\pi_{t'}}^{t'}. \tag{8}$$

解决了 p(l|x)的计算后，还需要解决的是训练问题。根据之前提到的梯度下降方法，我们需要对 p(z|x)求导数（z 是输出序列）。由于使用的机器学习框架具有自动求导能力（类似于数值求导，但是更加精确），我们不需要手动求出其导数，但是 p(y|x)的公式并没有给出，我们只知道 p(l|x)的公式。

$$O^{ML}(S,\mathcal{N}_w) = -\sum_{(\mathbf{x},\mathbf{z})\in S} ln\big(p(\mathbf{z}|\mathbf{x})\big) \tag{9}$$

省略计算步骤，结果是：

$$\frac{\partial O^{ML}(\{(\mathbf{x},\mathbf{z})\},\mathcal{N}_w)}{\partial u_k^t} = y_k^t - \frac{1}{y_k^t Z_t}\sum_{s\in lab(\mathbf{z},k)} \hat{\alpha}_t(s)\hat{\beta}_t(s)$$

where (10)

$$Z_t \overset{\text{def}}{=} \sum_{s=1}^{|\mathbf{l}'|} \frac{\hat{\alpha}_t(s)\hat{\beta}_t(s)}{y_{\mathbf{l}'_s}^t}.$$

$$\bar{\alpha}_t(s) \overset{\text{def}}{=} \alpha_{t-1}(s) + \alpha_{t-1}(s-1) \tag{11}$$

下面是更加具体的解释：图片首先经过 RNN(下图蓝色框内)，然后 CTC 的第一步是生成每一时刻各字符的分布概率，其中深色表示概率高。输入的序列还没到 H 时，输出的概率都是 H 最高，然后到了 H 和 E 之间时，H，E 的概率分布接近，而到了 l 时，变成 l 的概率最大（颜色最深）。

CTC 第二步是解码，找出概率最大的“路径”，相当与把上一步生成的“矩阵”中深色（概率最大）部分连接起来。这一步也是给出了路径的概率分布（图中有给出了 3 种情况）。再下一步是 C T C 能够识别空格和重复字符的关键：先去除重复的连续的字符，这时样本中的连续字符得到了保留，因为被 ϵ 隔开了，而识别过程中的连续字符则得到了清除。接着把 ϵ 去掉，就得到了最终结果。

CTC 的总体流程见下面图 3

图 4. RNN+CTC 神经网络的识别过程

2.2 目标和评估标准

本文的目标是：对于灰度的手写文本段落或句子图片，能够以尽可能高的正确率把图片识别为文本。这里对于句子的正确率定义为 label error rate (LER) [5]：

$$LER(h,S') = \frac{1}{|S'|}\sum_{(\mathbf{x},\mathbf{z})\in S'}\frac{ED(h(\mathbf{x}),\mathbf{z})}{|\mathbf{z}|} \tag{12}$$

其中 *ED(p, q)* 是两句话的编辑距离，定义为把其中一句话变成另外一句话最少的插入字符数，替换字符数和删除字符数之和。（x, z）是一对输入/输出对，即输入图片和输出的

句子。h 就是能够把输入图片识别为句子的目标函数，即本文的目标。S 指的是训练集。因此这里的 LER 是对于一整个训练集来说的。对于一句话，则其 LER 为：

$$LER = \frac{ED(h(x), y)}{|y|} \tag{13}$$

其中 h 仍然为识别函数，y 为已知句子文本。

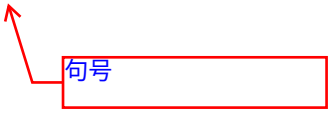
2.3 技术实现

整个模型由基本单元 GRU[6]，损失函数 CTC，还有为了防止过拟合的 Dropout 层组成。模型的基本神经元 GRU 是迭代神经网络的一种。关于一般的迭代神经网络已经在之前介绍。GRU 的优点是能够“记忆”比较长时间段内的函数（因果）关系，比如对于语音识别，在预测语音“Hello”时，先预测到了 H，则 GRU 能够结合"e"的发音，以及刚刚已经预测到的“H”，来更好的识别出字符“e”，而且在预测更后面的字符时仍然能够结合预测到“H”这一背景信息。GRU 的长时间记忆特性在文章 Andrej Karpathy 博客中的文章 Unresonable effectness of RNN 中得到了很好的展示。文章中介绍了输入某个作家的文字作品到 LSTM（和 GRU 类似，但是训练速度比 GRU 慢）神经网络进行训练，然后该网络竟然可以生成“文风”和内容都相近的文章。下面来介绍一下 GRU 的数学描述，以及其为什么能够实现长时间依赖的“记忆”：

CTC 是一种特别的损失函数。比如线性回归算法中最常用的损失函数最小二乘法，求出该损失函数的值最小时对应的线性模型的参数就确定了一个线性模型（对于一定的数据集）。同样的，对于 CTC 损失函数来说， CTC 的最小值对应的神经网络权重 W 等参数（也可以是非神经网络的其他模型）就是本文中训练过程的所求目标。CTC 和其他损失函数的区别是，给定长度为 m 的输出序列[y]，和长度为 n 的输入序列[x]，只要 m<n，CTC 函数就可以计算出这两个序列的差值，且 CTC 是可以求导的，因此应用在神经网络最后一层后，可以通过自动求导算法优化该模型。

Dropout 是一种流行的正规化方法，即让神经网络避免过拟合。这个方法的实现非常简单：随机在训练中直接忽略一些神经元不用，而在非训练时则保持不变。其原理是这样会强迫多个神经元去拟合同一个特征，就可以使得结果更加稳定，更少的依靠某些不稳定的特征，这样就忽略了训练数据集中的噪音，从而使得神经元在处理新的数据时能忽略其噪音从而得到更好的结果。

最后一层是 CTC 损失函数，其输出的数量（神经网络执行一次输出的已经识别的字符）可以小于等于输入（一个输入是一列像素，宽度为 1 像素，高度为字符图片高度）的个数，这是无分割识别和训练最关键的地方。这里的输入个数远远大于输出的个数，即输入很多列宽度为 1 的像素，因此一句话的输入可能有上百个，而输出的字符数量只有几十。在 CTC 之前主要应用的是 HMM 损失函数。下面重点介绍一下 CTC 损失函数



介绍完基本单元后，这里开始描述由基本单元组成神经网络的前面部分。模型采用了 5 层 GRU，每层的隐藏神经数依次递减，这样的设置是因为底层（更接近输入层）的像素组合非常多，需要更多神经元来提供足够的容量去拟合，而接近输出层的网络参数较少（英文大小写字母加上空格一共 53 类），所以用较少层数即可。

具体编程实现

模型的实际代码实现是在 google 的深度学习框架 tensorflow 中，该模型采用了计算图的抽象模型，以及 Python 作为编程语言。计算图模型提供了很好的抽象，可以直观地实现迭代神经网络。本模型调用了迭代神经网络，而本身的数据结构仍然属于树状。先在下图核心代码部分构建计算图，把各个输入，输出参数都作为图的节点输出，然后在后面调用该函数，获得计算图的输入输出节点。最后调用 sess.run，进行实际计算和训练。下面是核心部分代码：

```
    sink_x = tf.placeholder(tf.float32, [None, None, num_width])  # num feature is input length?
    sink_lenth_x = tf.placeholder(tf.int32, [None])
    sink_y = tf.sparse_placeholder(tf.int32)  # targets
    def nncell(hu):
        return tf.nn.rnn_cell.GRUCell(num_units=hu)

    stackTrain = tf.nn.rnn_cell.MultiRNNCell([
        tf.nn.rnn_cell.DropoutWrapper(cell=nncell(prob_numHidden[1]),
                                      output_keep_prob=prob_numHidden[0])
        for prob_numHidden in [[0.6, 400], [0.7, 300], [0.8, 200], [0.9, 200]]
    ])
    stackValid = tf.nn.rnn_cell.MultiRNNCell([
        nncell(prob_numHidden[1])
        for prob_numHidden in [[0.5, 400], [0.6, 300], [0.8, 200], [0.8, 200]]
    ])
    stack = stackValid if is_validating else stackTrain
    outputs, _ = tf.nn.dynamic_rnn(stack, sink_x, sink_lenth_x, dtype=tf.float32)

    sink_x_shape = tf.shape(sink_x)
    batch_s, max_timesteps = sink_x_shape[0], sink_x_shape[1]
    outputs = tf.reshape(outputs, [-1, num_hidden])

    W = tf.Variable(tf.truncated_normal([num_hidden, num_classes], stddev=0.2))
    b = tf.Variable(tf.constant(0.1, shape=[num_classes]))
    logits = tf.transpose(tf.reshape(tf.matmul(outputs, W) + b, [batch_s, -1, num_classes]), (1, 0, 2))
    cost = tf.reduce_mean(tf.nn.ctc_loss(sink_y, logits, sink_lenth_x))
    # optimizer = tf.train.MomentumOptimizer(initial_learning_rate, 0.9).minimize(cost)
    optimizer = tf.train.AdamOptimizer(initial_learning_rate, 0.9).minimize(cost)
    # decoded, log_prob = tf.nn.ctc_greedy_decoder(logits, sink_lenth_x)
    source_y_decoded, log_prob = tf.nn.ctc_beam_search_decoder(logits, sink_lenth_x)

    ler = tf.reduce_mean(tf.edit_distance(tf.cast(source_y_decoded[0], tf.int32), sink_y))
    # tf.summary.scalar('ler', ler)
    saver = tf.train.Saver()
    return sink_x, sink_lenth_x, sink_y, source_y_decoded, cost, optimizer, ler, graph, saver
```

图 5. 网络的 tensorflow 代码实现

以及 tensorboard 提供的可视化图：

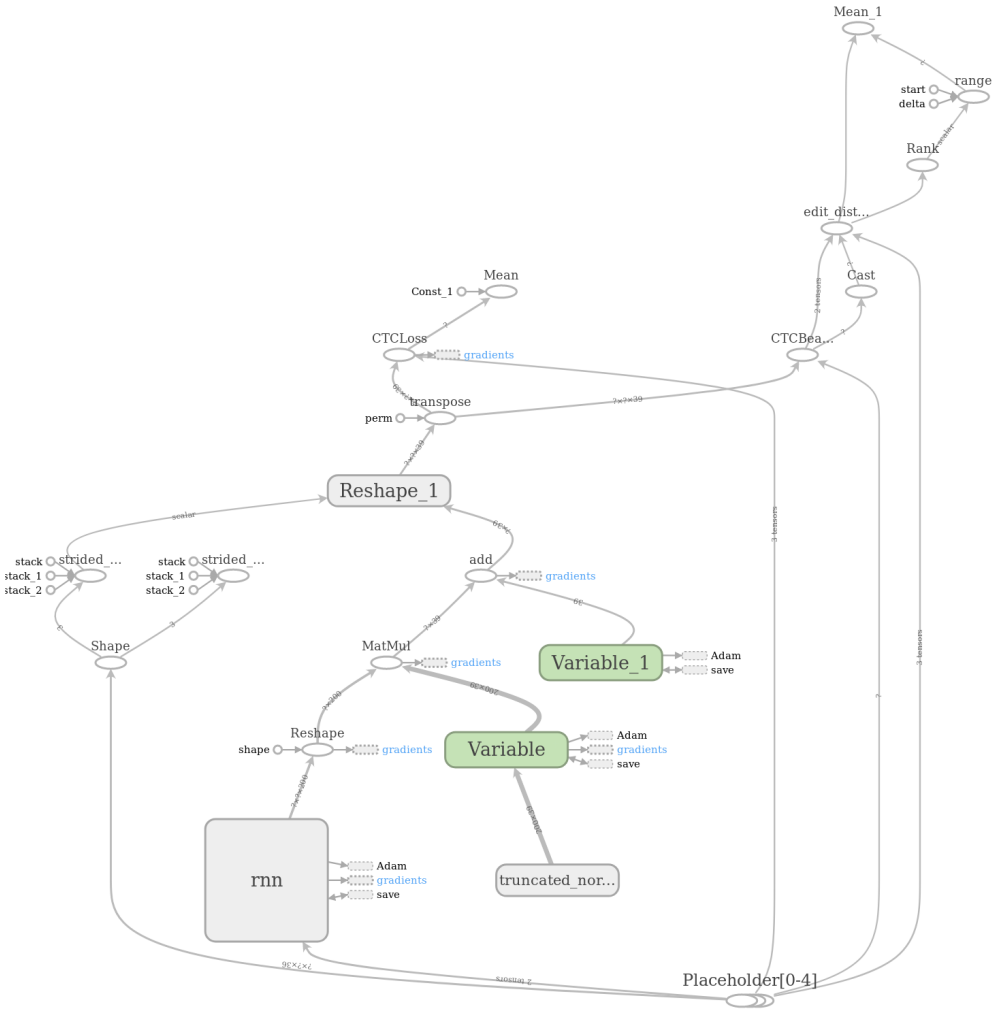


图 6. 网络的图表示（tensorflow 可视化）

图中的 Variable 和 Variable_B 分别对应上图代码中的 W，b， 是需要通过训练确定的参数，CTC_Loss 为之前提到的 CTC 损失函数。Placeholder 是输入，rnn 是 GRU(Gated recurrent cell)层。Matmul,add 是矩阵乘法。Reshape，transpose 等是数据张量的变换。

前面的图展示了完整的模型，但是难以辨识。简化的模型示意图如下：

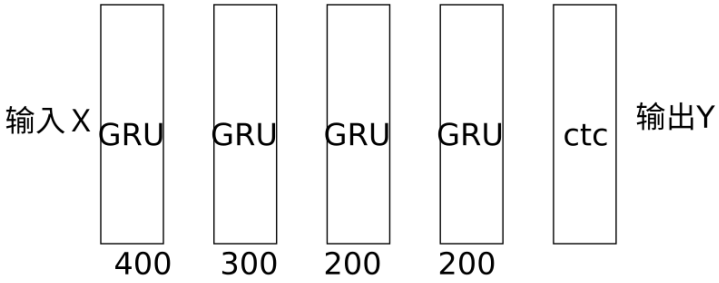
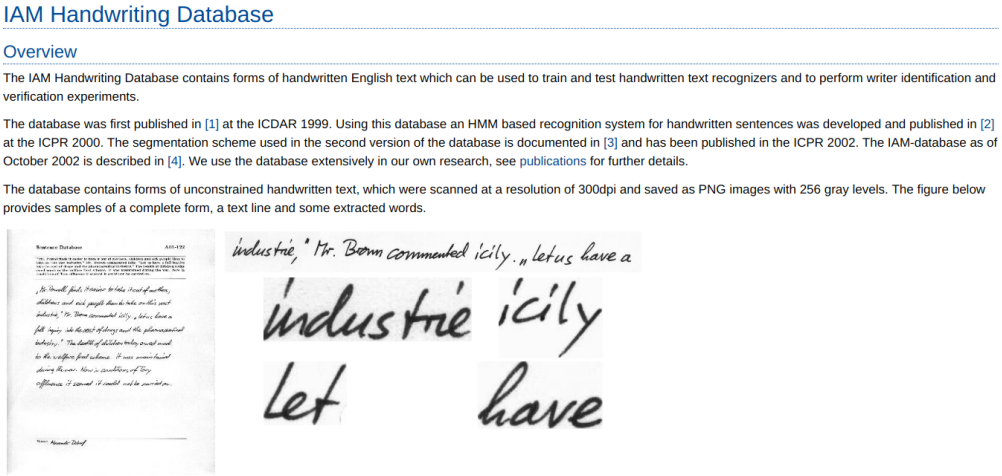


图 7. 简化的网络示意图。网络由 4 层 GRU 构成。

3 实验和结果分析

3.1 模型的训练优化过程

训练数据集包括 16432 张 PNG 格式的句子图片，来自 IAM 数据库[7]。


IAM Handwriting Database

Overview

The IAM Handwriting Database contains forms of handwritten English text which can be used to train and test handwritten text recognizers and to perform writer identification and verification experiments.

The database was first published in [1] at the ICDAR 1999. Using this database an HMM based recognition system for handwritten sentences was developed and published in [2] at the ICPR 2000. The segmentation scheme used in the second version of the database is documented in [3] and has been published in the ICPR 2002. The IAM-database as of October 2002 is described in [4]. We use the database extensively in our own research, see publications for further details.

The database contains forms of unconstrained handwritten text, which were scanned at a resolution of 300dpi and saved as PNG images with 256 gray levels. The figure below provides samples of a complete form, a text line and some extracted words.

图 8. IAM 数据库[7]

数据预处理：

运用本学科光学信息处理课程的知识，先把原来的灰度图片二值化，再进行中值滤波去除噪音。这两部的预处理是可选的，如果不进行的话，推测由于灰度图片和噪音带来的干扰，可能需要更大容量的神经网络，而迭代神经网络在实际测试中无法利用显卡 GPU 进行并行计算加速，所以训练速度会非常慢。处理完成后，把图像的缩放为 W×H = n * 12 大小的规格。这里高度是确定的，而宽度是可变的，因为句子长度不一样。

对于单个 GRU 节点，在某一时刻，其输入的向量为 1 维，长度为 12 的向量。这里长度为 12 的向量还需要转换为稀疏表示，即[1,0,0…],[0,1,0...]这样的形式。把迭代网络按照宽度 n 展开（这里把不定长度宽度也作为时间轴看待），得到一个输入维度是 2 维，尺寸为 n*12 的神经网络。展开的网络输出的向量仍然是 2 维，只是尺寸变成 n*m，长度保持不变。然后，输入到 CTC，计算出一个变长的输出向量，再把该向量用某种算法（beam decode）[5] 进行进行解码，得到最终结果。

训练环境是 intel i5-4690 CPU(四核，3.50GHz) + 18G RAM
训练途中的结果：在 内存训练大概 1 天后（迭代神经网络需要很长的训练时间），结果如下。

```
9670/13813 of data
Epoch 3/500,train_cost 50.629, train_ler 0.400, accumLer 1187.624,time = 19563.292
2018-04-19 00:24:35 ----> 2018-04-19 21:05:18
Original:
so the idea of personal mission by
Decoded:
so the idea of arysoiol rmision b p
ler : 0.20588236,minimal:0.225
Original:
the prime minister to paris was dropped
Decoded:
the ninme tiniser to sis tos lanped
ler : 0.2820513,minimal:0.225
Original:
hand picked team under the leadership of
Decoded:
tond pica d teom wndar the eadar sh f gf
ler : 0.35,minimal:0.225
------avg ler:0.2793112148841222-------
```

图 9.神经网络训练中的效果，Decoded 为识别结果，Original 为标注。

original 表示原本的句子，decoded 表示由神经网络识别的结果，由于是正在训练过程中，所以只取了 3 个样本进行测试。可以看到第一句正确率比较高，前面的 so the idea of 都对了，可是从后面就开始出现较多错误。

这是用来测试的 3 张图片：

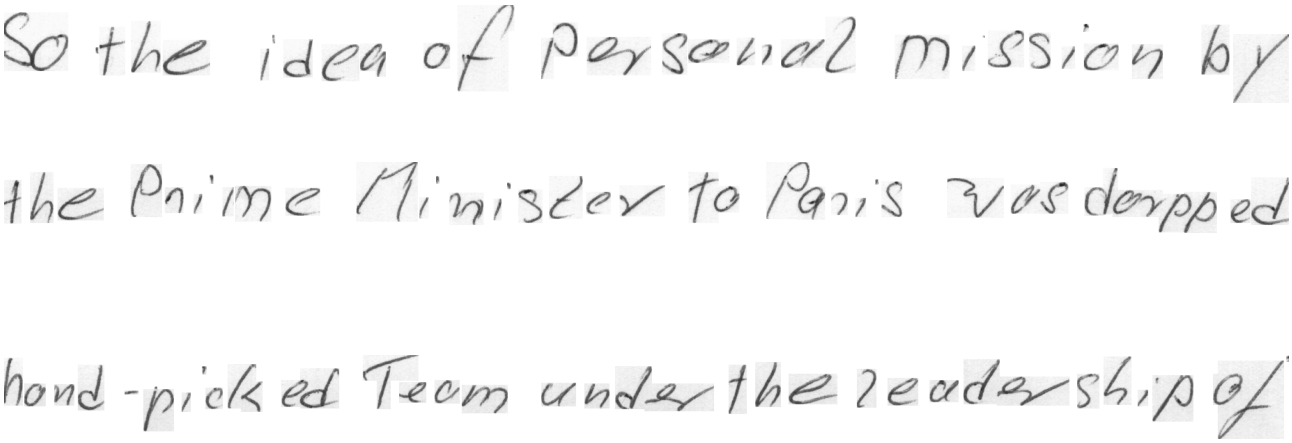


图 10. 样本数据集展示，来自于[7]

可以看到有的地方即使是人也难以辨认出某些字母。

训练优化过程

参数 1： 当输入图片高度缩放为 10 时，LER 始终无法降到 0.5 以下，通过观察缩放后的图片(图 11),推测是因为高度太小造成特征丢失，因此在后面测试中增加图片高度。

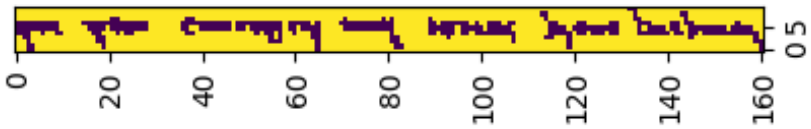


图 11. 输入图片高度缩放为 10. 可以看到细节丢失太多。

参数 2：当输入图片高度缩放为 36 时（图表未进行平滑处理）

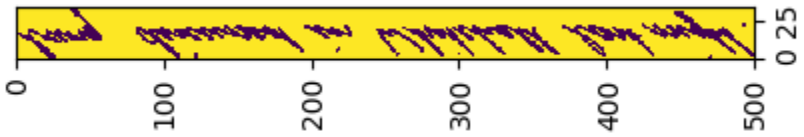


图 12. 输入图片高度缩放为 36，细节得到了一定保留。

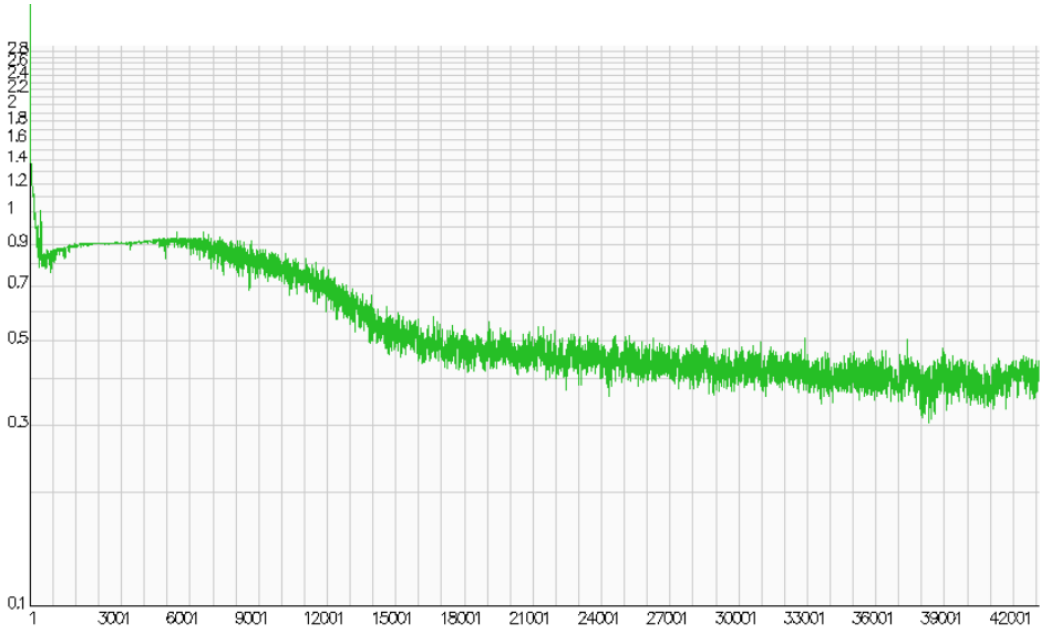


图 13. 实际 LER（表示真实正确率）

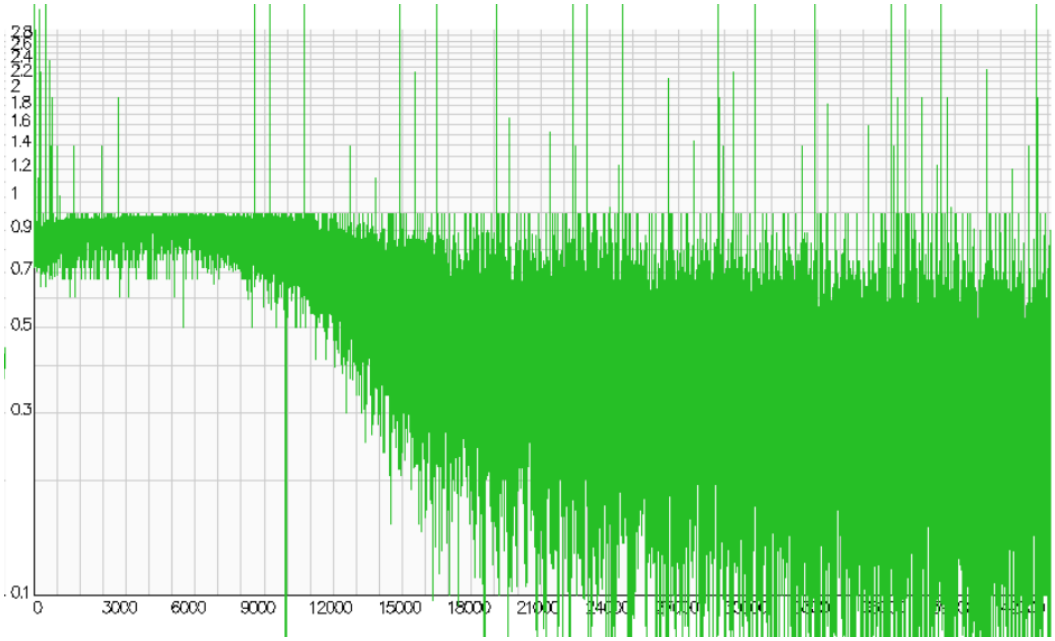


图 14. 训练 LER（表示神经网络的收敛）情况。后面曲线超出边界是由于 Y 轴是对数的。

图 13 是实际（测试） LER（表示真实正确率），图 14 是训练 LER（表示神经网络的收敛）。横轴为 42000 的点表示神经网络训练到第 42000 个数据时的 LER。可以看到左边的 LER 随着训练逐渐变小，但是到 0.4 以后不再下降，右边的训练 LER 表示大概在第 30000 时近似收敛完成，但收敛效果不好。可能的原因是训练数据量不够，只有 1 万多张图片，而目标函数过于复杂。

3.2 结果分析与对比

在尝试了改变图片高度和网络层数后，最佳结果是：平均 LER 大致为 0.2453 （57 个测试数据集），此时的网络结构为 4 层，第一层包含 400 个隐藏神经元，第二层 300，其他层 200。下图是训练完成后的识别结果。

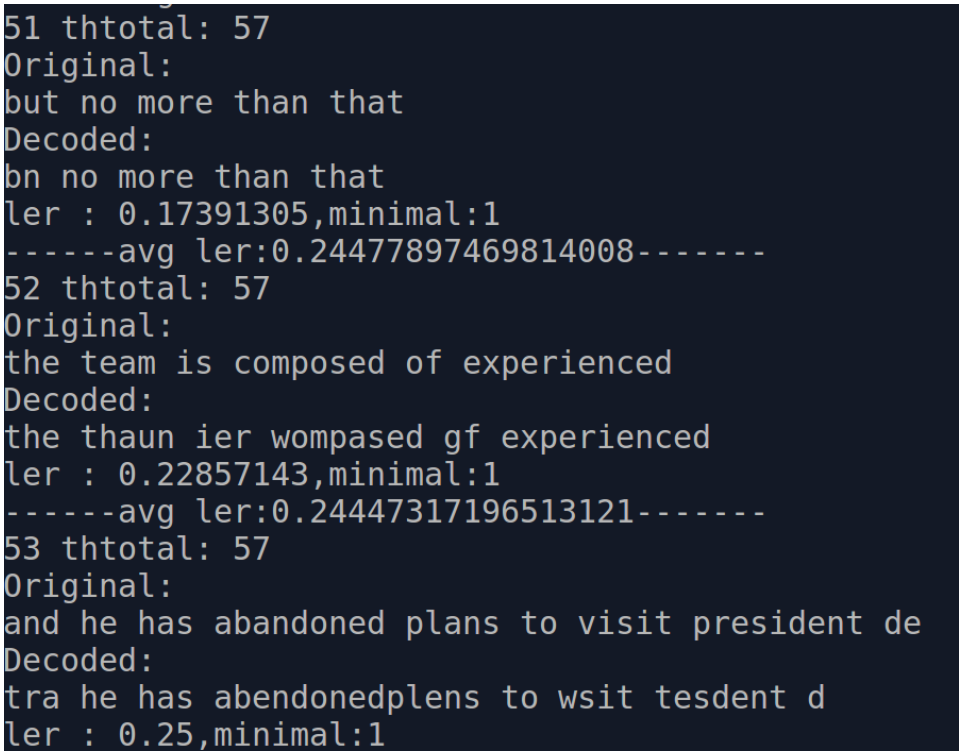


图 15.识别结果。Original 表示原来的图片，decoded 表示识别出的字符。

错两格，全文类似的地方还有一些，检查修改

整个训练过程的曲线如下：整个训练过程非常漫长，大概用了 70 多个小时才达到目前的效果。

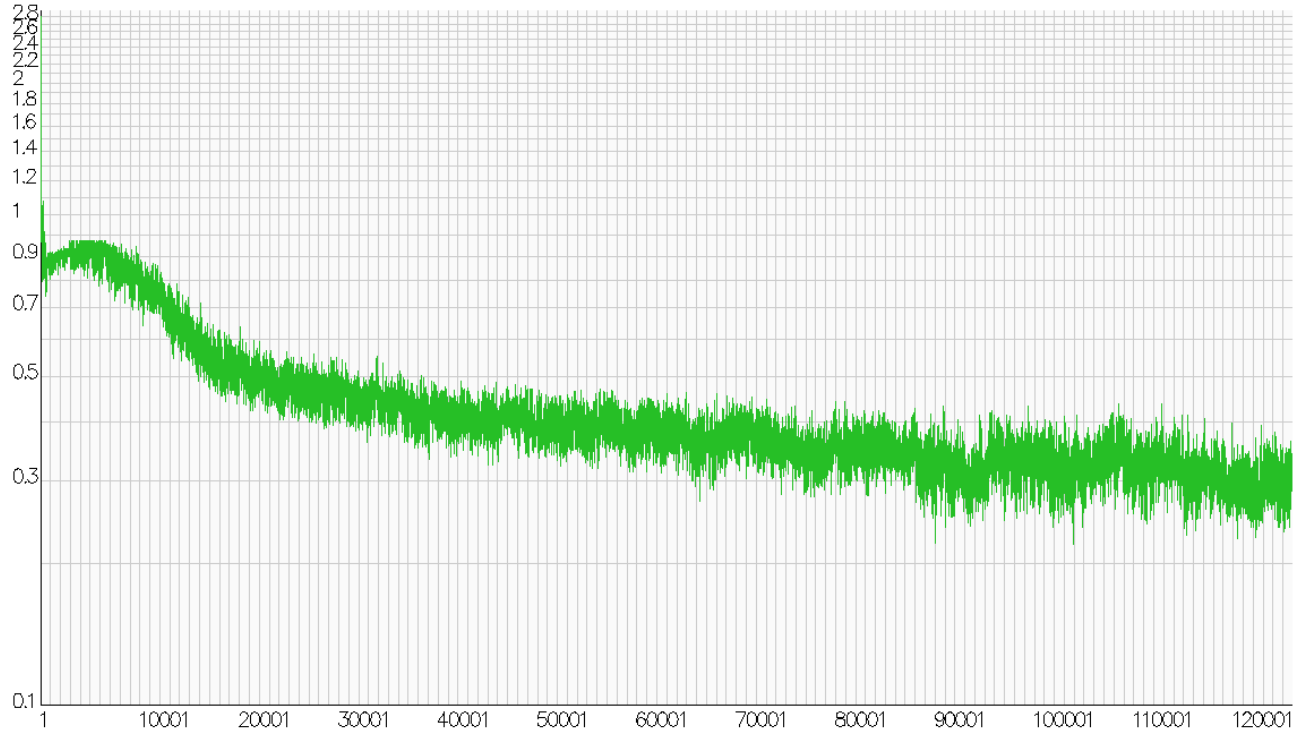


图 16. 训练过程（耗时>70h）。可以看到神经网络的 LER 在训练到 12000 个数据时 LER 到了 0.3 左右。

纵轴是 LER, 横轴是通过训练集的数量（整个数据集有 16432 张标注了的图片，120001 表示整个数据集过了 7 遍）

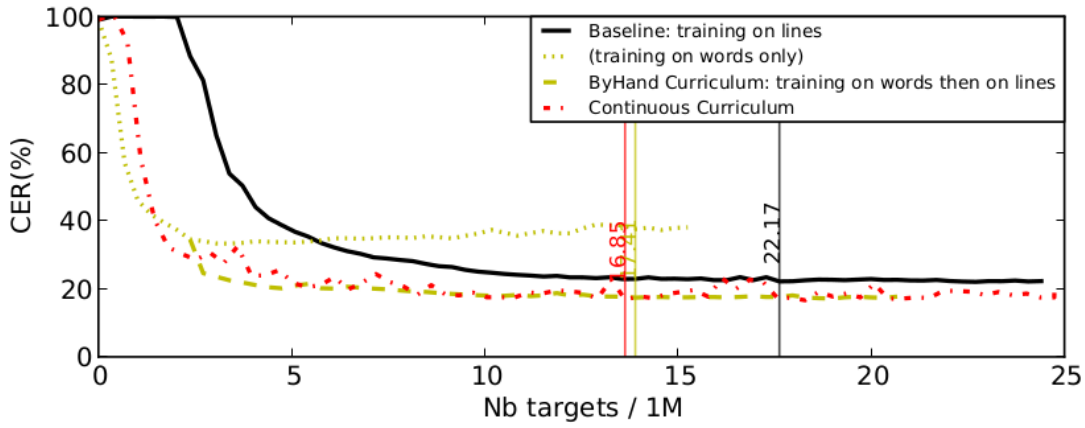


图 17. 另一篇[8]使用多维 LSTM 和卷积神经网络的论文给出的结果，只查看其中的 Baseline 即可，其表示的是用句子去训练。点状曲线表示的是用单词去训练，和本文不同。

上图是使用同样数据集的其他论文的结果[8]。该论文也实现了端到端的 OCR 识别网络，其最后一层同样是 CTC 损失，只是其网络结构中的前面层综合了最新的多维 LSTM 和卷积神经网络，远比本文的模型复杂：

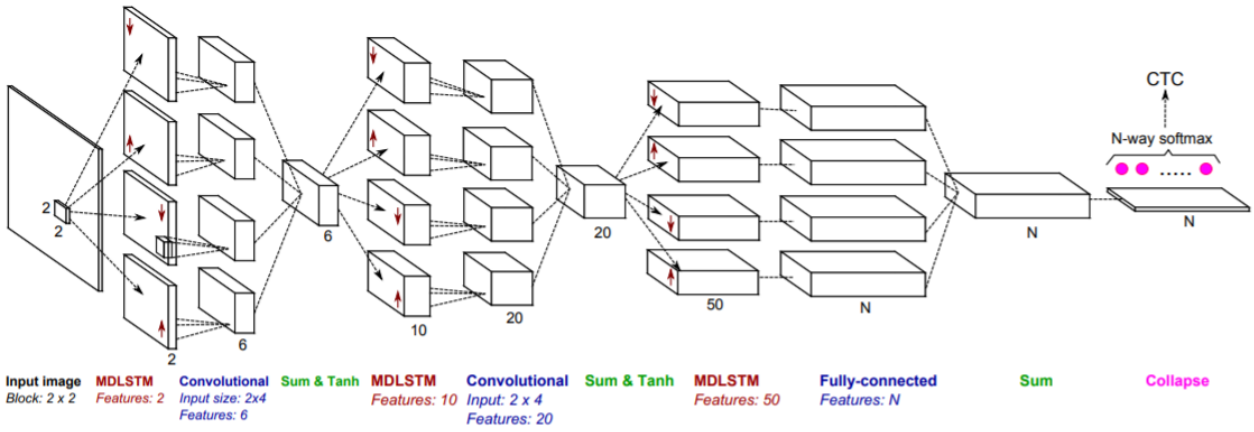


图 18. 论文[8]的网络结构。由 3 层 MDLSTM(multi-dimensional lstm)和 3 层 CNN 加上 CTC 构成。

中间空隙太大

学界 2017 年发表的另一种方法是结合 CNN 和 LSTM 进行联合训练[10] ． CNN 卷积神经网络也属于深度学习的方法，其主要用于固定大小的图片的识别，如果图片大小不一样需要预处理到同一尺寸，无法用于变长的序列识别。CNN 的优点是能够识别图片中的特征，并进行组合。CNN 识别单个手写字符的正确率非常高，能达到 98%以上，但是对于序列识别基本无能为力。所以如果能把 CNN 和 RNN 结合应该能提高准确率。作为拓展，下面简要介绍一下该文的思想。这篇文章使用 CNN 先把序列按照一定步长切片为子序列，然后把子序列输入 CNN，CNN 输出序列特征向量到迭代神经网络，之后的步骤就和本文的方法一样了。

4 总结与展望

本论文应用机器学习领域较新的迭代神经网络方法，去尝试端到端解决手写英文字符的ＯＣＲ识别问题，即整个过程中没有字符分割这一过程，训练时也不需要标注了分割的数据，只需要整句标注的数据集进行训练，而无需进行对齐。这极大简化了训练数据集的产生。无分割识别对于手写字符也是必要的，因为连笔非常多，较好的分割是不太可能的。

文中给出了在 tensorflow 中的代码实现和实际测试数据，由于计算机能力有限，以及迭代神经网络相对于前向神经网络无法利用 GPU 加速，其训练速度慢很多，所以只尝试了以较低分辨率（句子图片宽度为 36 像素）进行实验。

最后把测试结果进行了定性评估以及定量地和近年学界其他更先进神经网络模型的比较[8]。最终的结果离学界最好结果仍然有一定差距，但是基本符合预期，证明模型的实现和训练都没有问题。

接下来我会继续试验文中提到的其他先进神经网络，特别是卷积－迭代神经网络（CRNN），因为卷积网络适合特征的提取，而 RNN 能够处理变长序列

错两格

句号

另外，除了使用评估中提到的结构更加复杂，先进的神经网络，还有一个提高准确率的方向是使用语言模型，即针对该语言输入相关背景信息。比如可以引入单词库，然后根据其他算法自动修正识别中的错误。该方法的缺点是需要该语言相关的信息。另一个研究的方向是完全不使用 CTC 损失函数，而是引入所谓注意力模型（attention）[10]的机制来替代 CNN，这样可以实现整篇文章的端到端识别，甚至不需要进行文本行列的分割。

【参考文献】

字体对照模板再看看，
我记得不是这个字体

[1] Ian Goodfellow , Yoshua Bengio , Aaron Courvil ,Deep Learning,MIT Press,2016

[2] Kurt Hornik, Approximation Capabilities of Multilayer Feedforward Networks. Neural Networks, vol. 4, 1991.

[3] Balázs Csanád Csáji . Approximation with Artificial Neural Networks . MSc thesis.EÖtvÖs Loránd University.2001

[4] Lecun, Y. A theoretical framework for back-propagation. Proceedings of the 1988 Connectionist Models Summer School, CMU, Pittsburg, PA.

[5] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. ICML 2006, Pittsburgh, USA, 2006.

[6] J. Chung, C. Gulcehre, et al."Gated feedback recurrent neural networks", ICML-15, 2015.

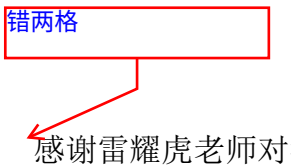
[7] U. Marti and H. Bunke. The IAM-database: An English Sentence Database for Off-line Handwriting Recognition. Int. Journal on Document Analysis and Recognition, Volume 5, pages 39 - 46, 2002.

[8] Graves, A. , Schmidhuber, J. Offline handwriting recognition with multidimensional recurrent neural networks.In Advances in Neural Information Processing Systems, volume 21, 2008.

[9] B. Shi, X. Bai and C. Yao, "An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 11, pp. 2298-2304, Nov. 1 2017

[10] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In Proceedings of the 2015 Conference on EMNLP.

致谢



感谢雷耀虎老师对我的鼓励和支持！这次自己的选题由于涉及的技术较新，虽然自己在课外学习了很多机器学习的相关知识，而且已经开始计算机方面的工作，并且之前已经使用过神经网络，但是对于较新的深度学习方法其实把握并不大，还好得到雷老师的一直鼓励。

感谢室友袁嘉言同学之前和我一起参加图像识别比赛，在参赛过程我慢慢开始对机器学习产生兴趣，并且接下来持续阅读相关书籍。

感谢牛丽红老师教我数字图像处理课程，其中讲到的图像滤波等预处理提升了论文的最终效果。

感谢郭金川老师和李冀老师教授我电磁场与电磁波和量子力学课程，这些课程的学习使我的抽象思维得到了很大锻炼，从而在阅读论文时更有信心。

【Abstract】The recognition of sequence data has lots of applications, such as speech recognition, text recognition and machine translation. In the field of machine learning, because of the optical recognition rate for the printed characters is generally above 95%, it has been considered as a common job. However, recognition of handwriting character in unsegmented paragraph is still under the focus.

This thesis explored a novel way to recognize handwriting English sentences, without any prior segmentation to individual characters. The method came from machine learning and deep learning. More specifically, the model was implemented with recurrent neural networks and CTC (connectionist temporal classification), which are in charge of the context sensitive recognition and automatic arrangement between input and output. Then, the model was implemented in Google’s Tensor flow framework. Finally, the test result and comparison with other new methods was provided.

【Key words】 handwriting character recognition , recurrent neural network , deep learning