基于长短时记忆网络的验证码识别系统

汤智阳, 华中科技大学计算机学院

摘　要　验证码是一种全自动区分计算机和人类的图灵测试.验证码识别是文字识别的一个领域,研究验证码的自动识别方法对文字识别领域有很大的用处。本文分析了传统验证码识别方法并指出其中的难点，分析了基于深度学习进行验证码识别的方法，然后选择了深度学习的一种网络——长短时记忆网络，进行验证码识别，详细介绍了识别方法及实验结果。最后对未来研究工作进行了展望。

关键词　验证码识别;文字识别;深度学习;长短时记忆网络

Abstract　Captcha Recognition is a Turing test which fully automatically distinguish between computers and humans.Captcha Recognition is a field of Optical Character Recognition and has a great use to it. This paper analyzes the traditional method of Captcha Recognition and points out the difficulty.Then it analyzes the method of using deep learning to recognize captcha, and uses the long short term network to do captcha recognition. It also describes the result of experiment. Finally, the future research work is prospected.

Key words　captcha recognition; deep learning; long short term network

文字识别(optical character recognition, ocr)是利用计算机自动识别字符的技术,是模式识别应用的一个重要领域.文字识别在很多领域都有着很重要的作用.验证码识别(captcha recognition)是文字识别的一个领域,验证码识别的方法可以推广到一般的文字识别应用领域中.因此研究验证码的识别也很有意义.

传统做验证码识别的方法，一般需要经过如下三步:

1. 图像二值化.一副图像的包括目标物体,背景还有噪声,要想从多值的数

字图像中直接提取出目标物体,最常用的方法是设定一个阈值 T,用 T 将图像的数据分成两部分:大于 T 的像素群和小于 T 的像素群.这是研究灰度变换的最特殊的方法,即图像二值化.

2. 字符分割.为了识别出字符,需要对图像中的字符进行分割,把每个字符当作单独的一个图片来看待.这里涉及到图像分割的算法.
3. 字符识别.对上一步分割出来的一个个字符进行识别,最常用的方法就是模板对比,把分割出来的字符和提前训练出的模板进行对比分类,找出相似度最高的即为识别结果.这里涉及到许多分类算法.

传统验证码识别方法中最复杂的一步就是字符分割,因为现在的验证码都做的很复杂,有的字符变形严重,有的字符连在一起互相交叉,对于这种字符,很难分割的很完美.因此如果有能够不用分割,就识别出验证码字符的方法将会很方便.即实现验证码的端到端识别,所谓端到端就是不需要对原始数据做任何处理,输入原始数据,输出想要的结果.

深度学习是一种端到端的学习方式,整个学习过程中不需要中间的和显著的人类参与.直接把海量数据放到算法里,让数据自己说话,系统会自动从数据中学习.从输入到输出是一个完全自动的过程.利用深度学习进行文字识别可以避免传统验证码识别中的难点:字符分割.目前应用深度学习进行文字识别主要有以下两个套路:

1. 把文字识别当作是多标签学习的问题.例如 4 个字符的验证码就相当于有 4 个标签的图片识别问题,可以用卷积神经网络(Convolutinal Neural Networks,CNN)来实现.
2. 把文字识别问题当作一个序列识别问题来解决.可以把图片当作一个序列输入到神经网络中,最终输出文字.这和语音识别问题类似,语音识别是将语音作为序列输入到神经网络中进行学习,所以可以用语音识别常用的方法来进行文字识别,比如 lstm+ctc.

我们将以上两种方法都实现了,本文只介绍第二种方法.即使用 long short

term network(lstm)和 Connectionist temporal classification(ctc)进行验证码识别.

本文第 1 节首先介绍了循环神经网络模型(recurrent neural networks,rnn),然后介绍了循环神经网络的改进,长短时记忆网络和 ctc 分类算法,主要对理论知识方面进行了详细介绍.第 2 节介绍了我们用 lstm 和 ctc 进行验证码识别的程序框架和实验代码,同时还会介绍我们进行深度学习所使用的训练集是如何生成的.第 3 节介绍了进行验证码识别的实验环境和实验结果,为了方便的进行测试,我们还开发了一个页面方便来显示识别结果,在该节也会介绍该系统.第四节介绍了总结和对后续研究工作的展望.

1 引言

1.1 循环神经网络(RNN)

在传统的神经网络模型中,数据一般是从输入层到隐含层再到输出层,层与层之间是全连接的,每层之间的节点是无连接的.但是这种普通的神经网络对于很多问题却无能无力.例如,你要预测句子的下一个单词是什么,一般需要用到前面的单词,因为一个句子中前后单词并不是独立的.而循环神经网络模型解决了这个问题,在循环神经网络中,一个序列的输出不仅与当前的输入有关,还和前面的输入有关,具体的表现形式为网络会对前面的信息进行记忆并应用于当前输出的计算中,即隐藏层之间的节点不再是无连接而是有连接的,并且隐藏层的输入不仅包括输入层的输出还包括上一时刻隐藏层的输出.图 1 给出了循环神经网络的基本结构:

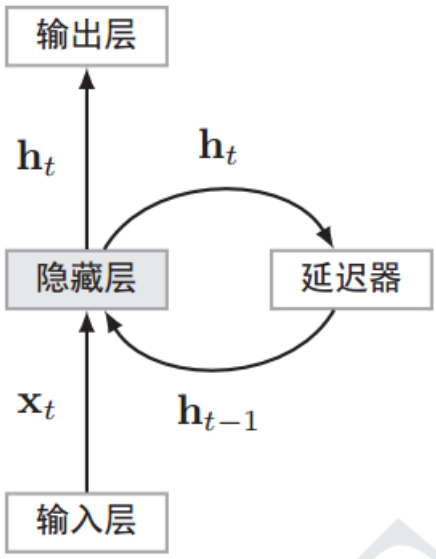

Fig.1 the architecture of RNN network

图 1:循环神经网络

从图 1 可以看出,数据x_t从输入层输入,经过隐藏层处理,产生的结果会分为两部分,一部分h_t用于输出,另一部分h_t输入给延迟器保存着,与下一时刻的输入同时进入隐藏层处理.这样就达到了记忆以前的输入的作用.图 2 是将图 1 展开后的循环神经网络:

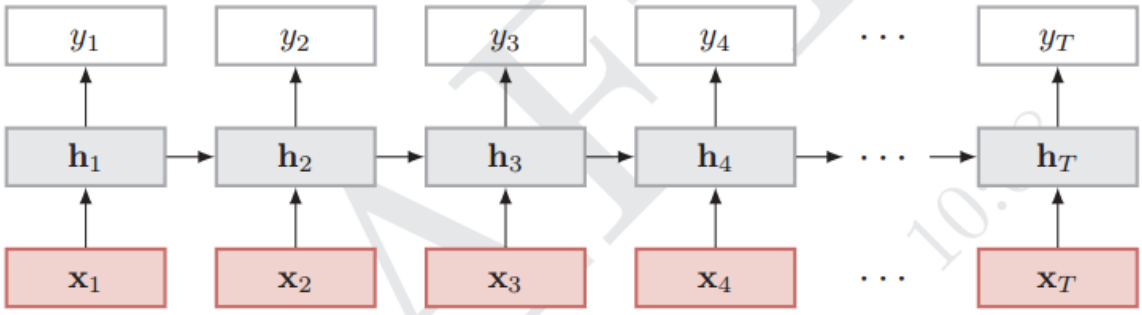

Fig.2 time expansion of RNN network

图 2. 按时间展开后的循环神经网络

RNN 在的链式特征使得它在许多基于序列的领域取得了不错的成绩,如语言识别,翻译,图片描述等.RNN 的关键点在于能连接先前的信息到当前的任务上,例如使用过去的视频段来推测对当前段的理解.有时候,我们仅仅需要知

道先前的信息来执行当前的任务.例如,我们有一个语言模型用来基于先前的词来预测下一个词.如果我们试着预测 “the clouds are in the sky” 最后的词,我们并不需要任何其他的上下文,因此下一个词很显然就应该是 sky.在这样的场景中,相关的信息和预测的词位置之间的间隔是非常小的,RNN 可以学会使用先前的信息.但是同样会有一种更复杂的场景.假设我们试着去预测“I grew up in France... I speak fluent French”最后的词.当前的信息建议下一个词可能是一种语言的名字,但是如果我们需要弄清楚是什么语言,我们是需要先前提到的离当前位置很远的 France 的上下文的.这说明相关信息和当前预测位置之间的间隔就肯定变得相当的大.不幸的是，在这个间隔不断增大时，RNN 会丧失学习到连接如此远的信息的能力.在理论上,RNN 可以处理这样的长期依赖问题.人们可以仔细挑选参数来解决这样问题,但实践起来却非常麻烦. Bengio, et al. (1994) [1]等人对该问题进行了深入的研究,他们发现一些使训练 RNN 变得非常困难的相当根本的原因.然而,LSTM 并没有这些问题.

1.2 长短时记忆网络(LSTM)

LSTM 是一种 RNN 网络的特殊类型,可以学习长期依赖信息. LSTM 由 Hochreiter & Schmidhuber (1997) [2]提出,并被 Alex Graves[3]进行了改良和推广.LSTM 通过刻意的设计来避免长期依赖问题,图 3 展示了 LSTM 的网络结构,从图中可以看出,LSTM 的重复模块包含了四个交互的层,而 RNN 中只包含了一个 tanh 层.

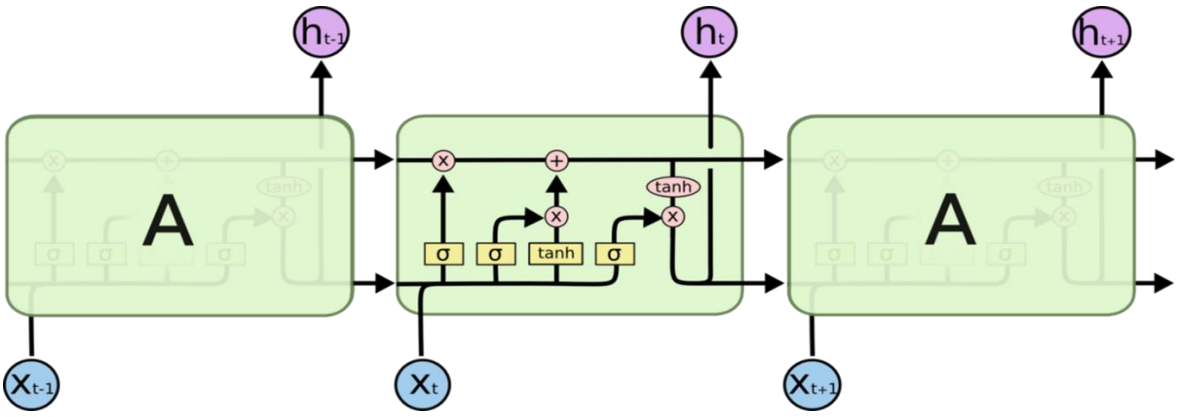

Fig.3. the architecture of LSTM network

图 3. LSTM 的网络结构

LSTM 的核心思想是细胞状态,也就是图 3 中贯穿于上方的水平线,细胞状态类似于传送带.直接在整个链上运行,只有一些少量的线性交互.信息在上面流传保持不变会很容易.LSTM 有通过精心设计的称作为"门"的结构来去除或者增加信息到细胞状态的能力.门是一种让信息选择式通过的方法.他们包含一个 sigmoid 神经网络层和一个 pointwise 乘法操作. Sigmoid 层输出 0 到 1 之间的数值,描述每个部分有多少量可以通过.0 代表"0 不许任何量通过",1 就指"允许任意量通过".LSTM 拥有三个门来保护和控制细胞状态.

(1) 遗忘门层

该门会读取上一序列的输出$h_{\{t-1\}}$和当前序列输入x_t,输出一个 0 到 1 之间的数值向量f_t,给每个在细胞状态$C_{\{t-1\}}$中的数字, 1 表示"完全保留",0 表示"完全舍弃".

(2) 输入门层

该层读取输入x_t,输入 0 到 1 之间的数值向量i_t,决定什么值我们将要更新,tanh 层根据当前输入创建一个新的候选向量$\bar{C}_t$,然后更新细胞状态,更新的方法为:将旧细胞状态$C_{\{t-1\}}$与f_t相乘,然后加上$i_t * \bar{C}_t$,这就是新的细胞状态值C_t.

(3) 输出门层

最终我们需要确定输出什么值,这个输出将会基于我们的细胞状态,但是也是一个过滤后的版本.首先,我们运行一个 sigmoid 层来确定细胞状态的哪个部分将输出出去.接着,我们把细胞状态通过 tanh 进行处理(得到一个在-1 到 1 之间的值)并将它和 sigmoid 门的输出相乘,最终我们仅仅会输出我们确定输出的那部分.

LSTM 模型可以用来对两个序列之间的关系进行建模.但是,传统的 LSTM,标注序列和输入的序列是一一对应的.而实际中识别出的字符序列或者

音素序列长度远小于输入的特征帧序列，所以在 LSTM 后加上 CTC 来计算最后的输出.

1.3 基于神经网络的时序分类(CTC)

CTC 是 Alex Graves 在 2006 年的 ICML 会议上提出的[4],它为 RNN 设计了一种新的目标函数，使得输入,输出序列不必等长,也不需要在训练前对输入,输出序列进行对齐.所以在输入特征和输出标签之间对齐关系不确定的序列分析问题上很适用.CTC 通过在标注符号集中加一个空白符号 blank，两个发音单元之间混淆或不确定的区域映射到"blank"节点.图 4 展示了 CTC 计算输出序列("THE CAT")概率的过程，先对所有可能映射成"THE CAT"的输入序列概率值的求和，然后去掉重复值和空格，得到最终序列。

Fig.4 CTC calculates the output sequence of "THE CAT"

图 4.CTC 计算输出序列 THE CAT 的过程

2 系统框架

2.1 系统设计

我们使用的深度学习框架为 mxnet[5],它是一个开源的深度学习框架,里面提供了 lstm 网络的接口,ctc 分类算法使用的是百度的开源代码库 warp-ctc[6] ，使用的数据集是通过 python 的验证码生成库 captcha[7]生成的,图 5 是生成的验证码的一个例子.从图中可以看出,生成的验证码中有字符的旋转,噪声和粘

连.

Fig.5 a example of captcha gernerated from python-captcha

图 5.captcha 生成的验证码例子

首先定义一个迭代器来输入数据,我们每次都是调用 python-captcha 这个库来根据随机生成的 label 来生成相应的验证码图片,这样训练集的个数可以设为足够大.

图 6 是网络结构模型图.具体流程如下：

(1) 输入图片，大小为 80*30 像素，每个图片包含 4 个数字组成的验证码

(2) 通过分片操作将图片划分为 80 个列向量，每一个列向量大小为 30

(3) 将每一列数据按序输入到 LSTM 网络中，这里使用了两层 LSTM 网络，第一层的输出作为第二层的输入

(4) 在 LSTM 的输出后接上一个全连接层，全连接层的神经元个数为 11，这是因为输入验证码是由数字组成的，每个数字可能取值有 10 个，再加上一个表示 blank

(5) 全连接层的输出为大小为 11 的列向量，输入 CTC 目标函数中计算损失，CTC 的计算结果包含了 blank 值，再去掉 blank 和重复元素，这样得到最终的预测字符，然后将其与 label 值比较,进行反向迭代算法.

LSTM 层数的增加可以提高网络的建模能力，但是层数过多可能引起模型泛化性能的降低，导致过拟合，使训练时间变得很长，我们选用了两层，在实验中发现效果也不错。

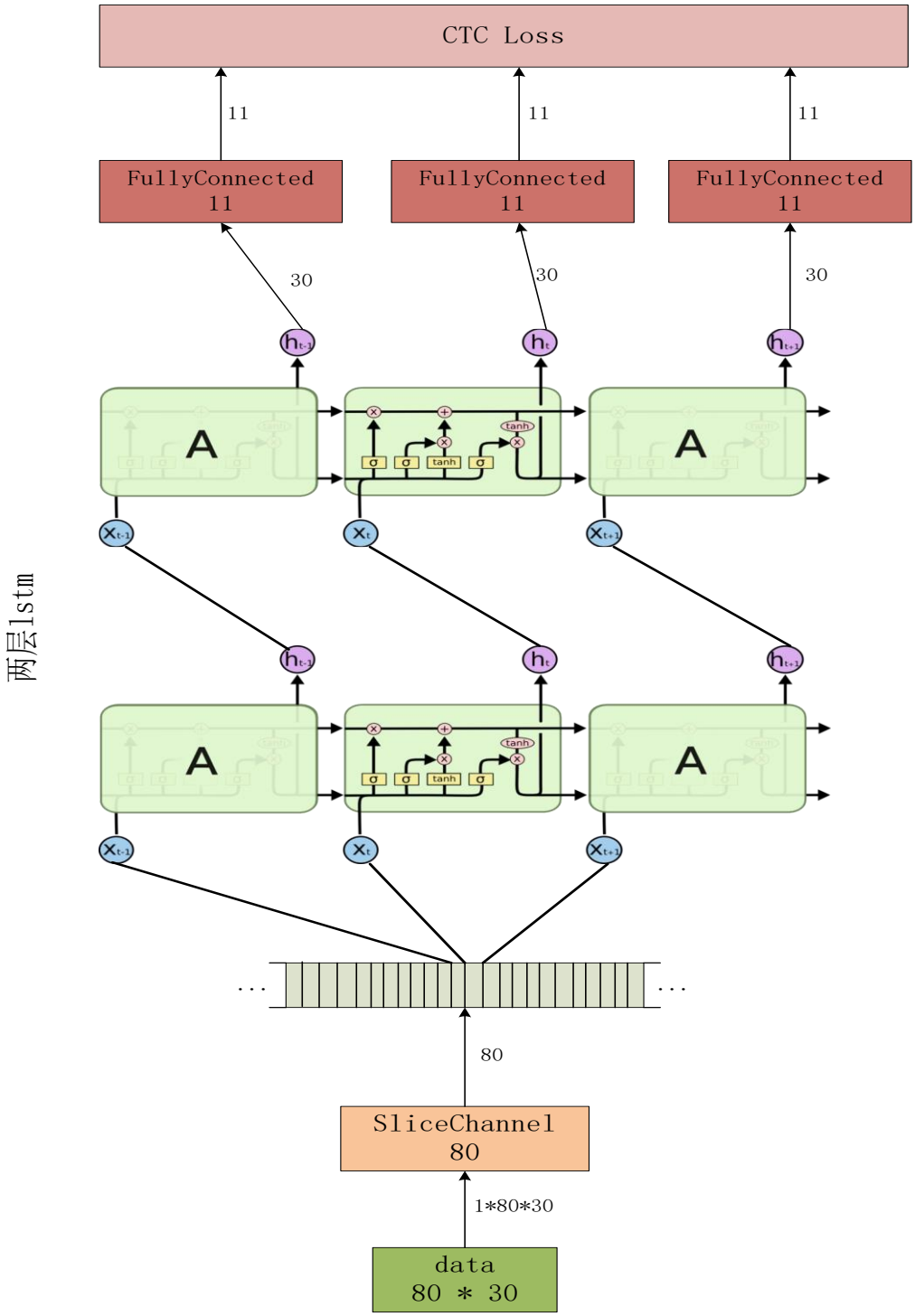

Fig.6 the model of this system

图 6.lstm+ctc 网络结构模型图

3 实验结果与分析

3.1 实验结果

我们在一台 GPU 为 NVIDIA TESLA K40M，内存为 12G 的服务器上进行了实验，使用 GPU 运行深度学习程序可以减少很多的时间，操作系统为 Ubuntu14.04，使用的编程语言为 Python。

我们在大小为 10 万的数据集上进行实验，验证码长度为 4，实验运行过程准确率变化如图 7，从图中可以看出，使用 CTC 作为损失函数来进行训练时，训练准确性收敛的速度很慢，一直进行到 7200 个 batch 后，准确性才开始明显提高，进行两个 epoch 后准确性基本收敛。

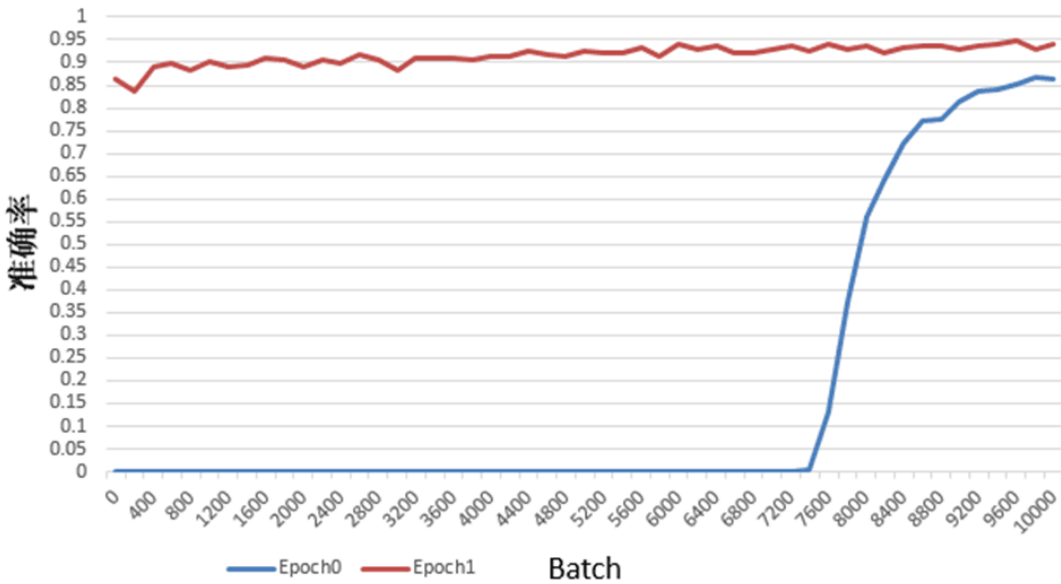

Fig.7 Learning rate during the training of this model

图 7. 训练过程中准确率变化曲线

在实验过程中，我们遇到了不少问题，也花费了很多时间，如 batch_size，learning_rate 等参数大小的设置对模型的训练能力有很大影响，在实验中发现 batch_size 过大会导致训练准确率突然降为 0，推测是发生了过拟合的现象，最终选择大小为 32 的，learning_rate 设置太小会导致训练速度过慢，我们设

置的大小为 0.001。

3.2 验证码测试系统

为了方便使用训练好的模型进行测试，我们做了一个网页，图 8 是测试结果。具体流程如下：

(1) 用户选择模型，上传验证码，点击提交后，将数据发给后台

(2) 后台将接收到的验证码图片保存在本地，同时根据用户选择的模型，读取相应模型的参数，并初始化网络，将图片输入网络，得到输出结果

(3) 将输出结果返回给用户查看

Fig.8 test system of captha recognition

图 8. 验证码识别测试系统

4 总结及展望

验证码识别是一样涉及到图像处理，模式识别等领域的任务。本文针对传统验证码识别方法中存在的难点问题进行了分析，然后提出了基于深度学

习神经网络 LSTM 的定长验证码识别方法，在实验中验证了这种方法对于具有一定复杂度的 python-captcha 库生成的验证码识别性能很不错，我们还做了一个验证码识别的 web 系统来方便进行测试。另外我们在实验中只使用了数字组成的验证码，复杂度较低，接下来打算做数字、字母、汉字组合验证码的识别。

参考文献

[1] Bengio, et al.Learning Long-Term Dependencies with Gradient Desent is Difficult. Ieee Transactions On Neural Networks, 1994,3:5-2

[2] Hochreiter S, Schmidhuber J. Long Short-Term Memory. Neural Computation.1997,9(8):1735-1780

[3] Alex Graves. Supervised Sequence Labelling with Recurrent Neural Networks. Springer Berlin Heidelberg, 2012, 385

[4] Alex Graves.Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. International Conference, 2006, 2006:369-376

[5] URL: http://mxnet.io/index.html

[6] URL: https://github.com/baidu-research/warp-ctc

[7] URL: https://pypi.python.org/pypi/captcha/0.1.1