CRNN复现——模型迁移与新架构探索

黄彬
2019011344
计96
huangb19@mails.tsinghua.edu.cn

孟本源
2018010527
计91
mby18@mails.tsinghua.edu.cn

夏箫
2019013300
计97
xiax19@mails.tsinghua.edu.cn

1 简介

复现论文为An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition[1]

我们复现了论文中的模型架构，训练至收敛并且在原论文使用的IC03等测试集上完成测试，所有指标均与原文相符。并且在性能测试中发现，用Jittor框架下的推断速度相比PyTorch框架有显著提升。

我们根据原论文及一些相关文献，归纳提出一种处理该任务的模型范式，即“特征提取-特征优化-序列生成”。该范式的三个部分所处理的任务相对独立，且有多种可行的实现方式，因此模型的这些部分可以独立地进行替换；这实现了模型组件的解耦，可以对模型设计起到很好的规范与指导作用。我们基于此实现了一些额外的替换模块，进行了多种组合测试，验证了该范式的合理性，并且也对这些替换模块本身的性能进行了对比。包括已经额外实现的残差网络、Encoder-Decoder结构、注意力LSTM结构等，而且支持扩展更多替换模块。

除了可用模型的增加，相比于原仓库代码，我们还增加了构建数据集、多种方式读取数据集用于IO加速、训练过程记录、用BKTree加速转录、在所有测试集上测试等功能。以上工作在GitHub中开源。[1]

我们还将CRNN架构用于乐谱识别任务，取得超过baseline的准确率，同样已经开源。[2]

2 复现论文介绍

复现论文任务为识别图片中的序列，如单词等。

Figure 1: 单词识别任务：应识别出“SHAKESHACK”

[1]https://github.com/huangb19/crnn_jittor
[2]https://github.com/Rishubi/CRNN-OMR-with-Jittor

论文提出了CRNN结构用于完成该任务，核心思想是结合CNN与RNN：使用CNN得到特征，输出feature sequence，使用RNN识别上下文信息，输出序列。

由于采用定长时间步输出，需要引入空白字符'-'来适配不定长的label。由于CNN的连续局部特征可能捕获的是同一个位置的符号，输出时的连续符号应该被移除。因此引入映射B:先移除连续重复字符，再移除空字符。例如"hh-e-l-ll-oo-"经过映射后变为"hello"。

对于模型的输出$\mathbf{y} = y_1, ..., y_T$,$y_i$为第$i$个时间步在符号集上的概率分布，序列$\pi$出现的概率：$p(\pi|\mathbf{y}) = \prod_{t=1}^{T} y_t^{(\pi_t)}$，模型输出的字符串为$I^* = B(argmax_\pi p(\pi|\mathbf{y}))$

如果限制输出的字符串在某个备选串的集合D(称为词表)中，需要计算生成其中某个串l的概率：$p(l|\mathbf{y}) = \sum_{B(\pi)=l} p(\pi|\mathbf{y})$，模型输出的字符串为$I^* = argmax_{l \in D} p(l|\mathbf{y})$

由于词表可能很大，可以排除一些可能性很小的串，具体来说先计算无词表转录的字符串I，以与I的编辑距离不超过δ的单词子集作为词表，该过程可以使用BKTree进行加速。

注意到计算$p(l|\mathbf{y})$可能是指数级别的复杂度，可以用CTC方式快速计算。[2]

其它参考:

我们复现时参考的原深度学习框架是PyTorch[3]，用Jittor复现难点主要是对于Jittor环境不熟悉，通过Jittor相关人员的指导得到了很好的解决。

我们实现额外的替换组件时也参考了一些文献。残差网络的实现参考了He等[3]的研究，注意力LSTM结构则参考了同样处理环境文本识别任务的ASTER[4]。

3 实验

3.1 实验设置

我们复现了论文原始模型架构，采用的训练集为包含8M张人工合成图片的MJSynth[5]，learning rate=1e-4，25个epoch后在验证集上准确率达到98%

测试数据集为在真实世界取景的IC03[6],IC13[7],IIIT5K[8],SVT[9]。生成的结果将通过无词表/有词表两种转录方式生成最终字符串。其中，SVT数据集每张图片自带大小为50的词表，其它数据集的词表需要随机构建。对于IC03数据集，还使用了50k-words Hunspell词典作为词表。

3.2 实验结果

数据集	IIIT5K			SVT		IC03				IC13
词表	50	1k	None	50	None	50	Full	50k	None	None
原文准确率	**97.6**	94.4	78.2	96.4	**80.8**	**98.7**	**97.6**	95.5	89.4	86.7
复现准确率	96.8	**95.4**	**79.3**	**96.9**	79.4	97.8	97.3	**96.6**	**89.7**	**87.7**

Table 1: 复现实验结果

实验结果表明我们在所有数据集上均达到原论文水平。

3.3 额外工作

3.3.1 模型范式

我们提出了处理环境文本识别任务的"特征提取-特征优化-序列生成"模型范式，如下图所示。该范式包含三个部分，分别执行相对独立的任务。

特征提取部分的任务是从原始图像中提取出一组特征向量，这一般是一组长度相同的高维稠密向量。该部分通常可以用类CNN结构实现，例如基础的CNN，增加残差连接或短路的CNN，也可以根据需要增加特殊的机制(例如Shi等[4]增加了用于修正由拍摄角度引起的

[3]https://github.com/meijieru/crnn.pytorch

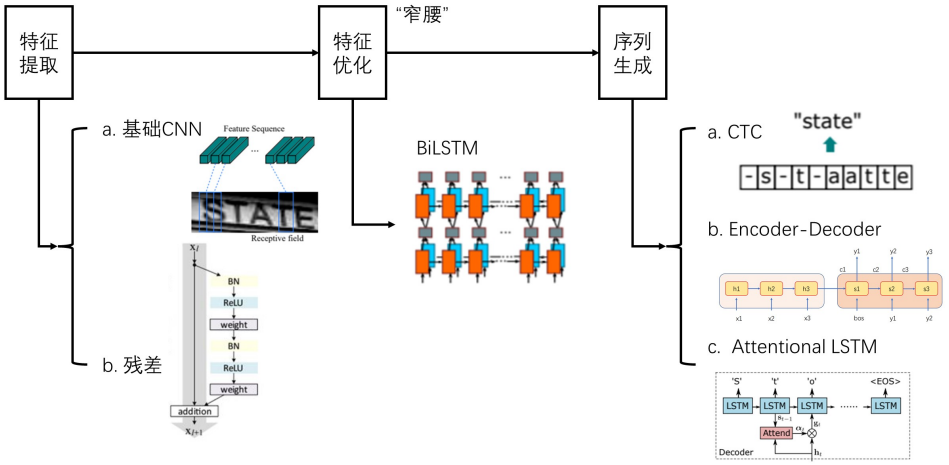

Figure 2: “特征提取-特征优化-序列生成”范式

文本失真的机制)。值得注意的是，这些特征向量应该具有受限的局部视野，即每个向量的数据仅来自于原图中的一部分区域，而非整个图像，我们发现这对于模型的性能有显著的影响。

特征优化部分的任务是结合上下文信息对特征向量进行优化。特征提取部分得到的特征向量具有局部视野，而特征优化部分可以在特征向量之间进行信息沟通，从而避免局部视野的局限性。这部分建议使用输入和输出等长的BiLSTM实现。

序列生成部分需要将特征向量转换为序列输出，而特征向量本身就可以视为一个序列，因此这是一个一般非等长的seq2seq问题。原论文使用简单的全连接网络对特征向量进行处理，然后使用CTC规则得到文本输出。实际上也可以采用更常规的seq2seq结构，例如Encoder-Decoder结构。

上述三部分中，特征提取和序列生成均有较为丰富的实现方式，而特征优化部分的实现方式较为单一，也就成为模型的“窄腰”。这约束了前后两部分的特征向量格式和语义，使得这两个部分的实现可以独立地进行替换。

我们实现了一些替换模块进行测试，括号中为下文将会采用的简称。

特征提取部分：(a)基础CNN(CNN)；(b)残差网络(Res)。

序列生成部分：(a)CTC(CTC)；(b)基础的Encoder-Decoder结构(ED)，训练时采用teacher forcing，验证和测试时采用free run；(c)带有注意力机制的LSTM结构(Attention)。

进行测试时，由于计算资源的限制，我们采用如下的规则进行比较：使用MJSynth数据集，采用第3-5个epoch内最好的验证集准确率及对应的损失函数值作为一个模型的结果，而非完全收敛后的准确率。测试结果如下。

需要注意，两种RNN结构的收敛速度均较慢，且Attention收敛更慢，因此在上述规则下两者表现较差，根据更多的训练数据，Attention实际上优于ED。序列生成部分采用CTC的模型和采用RNN的模型分别使用两种损失函数，因此两者的损失函数值之间不具有可比性。

我们根据上述范式进行模块替换后得到的多种模型的性能虽然有差异，但都处于较好的范围内。我们认为，这可以说明“特征提取-特征优化-序列生成”模型范式的合理性。

分析特征提取部分的不同实现，可以发现残差结构有效提升了模型的收敛速度。不过我们发现在更长期的训练中，采用残差结构的模型会陷入瓶颈，难以将准确率提高到98%以上，这可能是因为残差结构显著地增加了参数的规模，因此提高了训练难度。

组合方式	CNN+CTC	CNN+ED	CNN+Attention	Res(15层)+CTC	Res(23层)+CTC
准确率	95.52	94.77	93.67	97.40	97.60
损失值	0.008634	1.447	1.148	0.006217	0.006236

Table 2: 替换模块实验结果

分析序列生成部分的不同实现，我们发现RNN结构的训练难度明显高于简单的CTC结构，且在更长期的训练后RNN结构的准确率并未与CTC结构产生明显差异。在两种RNN结构之间比较，我们发现Attention在准确率方面有微小的优势，而损失函数值显著地低于ED，这说明带有注意力机制的结构可以对给出的结果有更高的信心。我们认为这表明注意力机制可以在一定程度上提升模型的性能。

3.3.2 乐谱识别

我们将CRNN结构迁移到另一种典型的图像序列识别任务——乐谱识别任务，以进一步探索该结构在序列识别上的优势。该任务要求根据乐谱图片生成乐谱符号序列，其中需包含调性、音高、音长等信息。数据集使用PrIMuS数据集，即未经扰动的乐谱图片，以及经过部分扭曲得到的Camera-PrIMuS数据集[4]。3、4分别为两种数据集的图片示例。

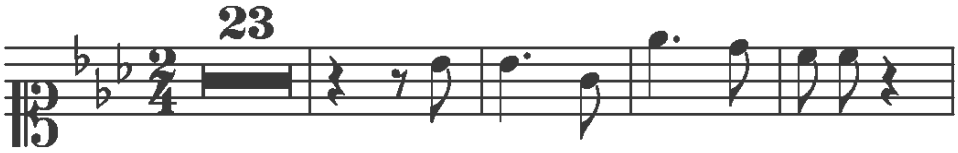

Figure 3: 无扰动乐谱示例

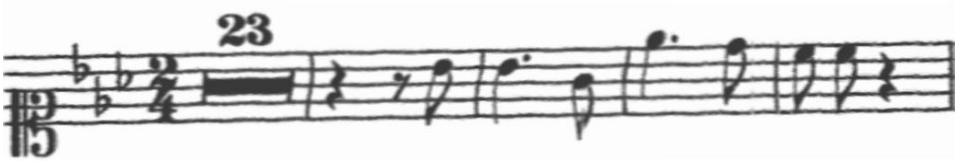

Figure 4: 带扰动乐谱示例

数据集中为乐谱提供两种编码方式，即语义编码与不可知编码。前者包含了符号的音乐意义；而后者仅包含了符号的形状与位置，如会将D大调编码为两个“#”号。前者涉及对符号间关系的分析，一个编码可能是多个子符号的组合，要求结合上下文推断；同时其符号表大小为1781，显著超过不可知编码的758个符号，辨别难度较大，因此我们选择实现语义编码的生成。[5]

模型参数较单词识别有较大改动。由于数据集仅包含87678个样本，卷积层数量有所减少；由于符号表大小远大于单词识别，输入图片大小与RNN输入维数都得以增加以识别与表示更复杂的符号形状；RNN输入序列长度有所增加以匹配乐谱序列长度。模型参数见表3。

采用3种实验方式，分别在无扰动数据集上训练后用无扰动数据测试、在带扰动数据集上训练后用带扰动数据测试、在带扰动数据集上训练后用无扰动数据测试，用序列准确率度量效果。实验结果见表4。

模型在两种数据集上的表现均达到较高水平，表明CRNN结构能应对符号表较大的情形，善于辨别复杂符号。

[4]https://grfia.dlsi.ua.es/primus/

[5]示例乐谱的编码为“clef-C1 keySignature-EbM timeSignature-2/4 multirest-23 barline rest-quarter rest-eighth note-Bb4_eighth barline note-Bb4_quarter. note-G4_eighth barline note-Eb5_quarter. note-D5_eighth barline note-C5_eighth note-C5_eighth rest-quarter barline”。

模块	参数
输入	128×800
卷积	32, k5×5, p2×2, s1×1
最大池化	k4×2, s4×2
卷积	64, k3×3, p1×1, s1×1
最大池化	k3×2, s3×2
卷积	128, k3×3, p1×1, s1×1
最大池化	k2×2, s2×2
卷积	256, k3×3, p1×1, s1×1
最大池化	k2×1, s2×1
序列	len100×dim512
Bi-LSTM	hidden size512
Bi-LSTM	hidden size512
转录	-

Table 3: 模型参数，每层卷积层后均有BatchNorm与ReLU。

组合方式	无扰动+无扰动	带扰动+带扰动	带扰动+无扰动
实验准确率	**93.14**	80.77	81.56

Table 4: 乐谱识别实验结果

3.3.3 性能测试

我们比较了CRNN在Jittor与PyTorch框架下的推断速度。从IIIT5K，SVT，IC03，IC13分别取128项得到大小为128的batch，在Jittor与PyTorch版本模型中重复推断100次，得到推断一张图片的频率(FPS)。

两种框架在四个数据集的推断频率见表5。

数据集	IIIT5K	SVT	IC03	IC13
Jittor	10454.2 ± 402.8	10267.5 ± 486.3	10363.2 ± 201.1	10345.0 ± 180.1
PyTorch	9164.6 ± 203.6	9099.1 ± 174.5	9136.4 ± 27.9	9136.4 ± 28.5

Table 5: 单图片推断频率

可见在各数据集中Jittor框架的推断速度均显著优于PyTorch框架。

4 总结

我们归纳提出针对环境文本识别任务的“特征提取-特征优化-序列生成”模型范式，这有助于让模型的各部分解耦，从而让模型结构研究更加灵活和规范化。我们基于此范式，对少量结构进行了对比测试，可以为后续的相关研究提供参考。

在乐谱识别任务上的成功表明，CRNN模型对于序列识别任务具有通用性，可以应对大符号表、复杂的记号等各种情况。模型在Jittor框架下的推断速度相比PyTorch框架有显著提升。

参考文献

References

[1] Shi, B., X. Bai, C. Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(11):2298–2304, 2017.

[2] Graves, A., S. Fernández, F. Gomez, et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. 2006.

[3] He, K., X. Zhang, S. Ren, et al. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778. 2016.

[4] Shi, B., M. Yang, X. Wang, et al. Aster: An attentional scene text recognizer with flexible rectification. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2035–2048, 2018.

[5] Jaderberg, M., K. Simonyan, A. Vedaldi, et al. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*, 2014.

[6] Lucas, S. M., A. Panaretos, L. Sosa, et al. Icdar 2003 robust reading competitions: entries, results, and future directions. *International Journal of Document Analysis and Recognition (IJDAR)*, 7(2-3):105–122, 2005.

[7] Karatzas, D., F. Shafait, S. Uchida, et al. Icdar 2013 robust reading competition. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1484–1493. IEEE, 2013.

[8] Mishra, A., K. Alahari, C. Jawahar. Scene text recognition using higher order language priors. In *BMVC-British Machine Vision Conference*. BMVA, 2012.

[9] Wang, K., B. Babenko, S. Belongie. End-to-end scene text recognition. In *2011 International Conference on Computer Vision*, pages 1457–1464. IEEE, 2011.