智能系统学报
CAAI TRANSACTIONS ON INTELLIGENT SYSTEMS

改进单点定位模型的轻量级端到端文本识别方法

曹锦纲, 张泽恩, 张铭泉

引用本文:
曹锦纲, 张泽恩, 张铭泉. 改进单点定位模型的轻量级端到端文本识别方法[J]. 智能系统学报, 2024, 19(6): 1503-1517.
CAO Jingang, ZHANG Zeen, ZHANG Mingquan. A lightweight end-to-end text recognition method based on SPTS[J]. *CAAI Transactions on Intelligent Systems*, 2024, 19(6): 1503-1517.

在线阅读 View online: https://dx.doi.org/10.11992/tis.202307012

DOI: 10.11992/tis.202307012
网络出版地址: https://link.cnki.net/urlid/23.1538.tp.20240910.1140.006

改进单点定位模型的轻量级端到端文本识别方法

曹锦纲[1,2], 张泽恩[1,2], 张铭泉[1,2]

（1. 华北电力大学 控制与计算机工程学院, 河北 保定 071003; 2. 华北电力大学 复杂能源系统智能计算教育部工程研究中心, 河北 保定 071003）

摘　要: 针对现有文本识别方法推理速度慢、模型参数量大的问题, 提出一种改进单点定位模型（single-point scene text spotting, SPTS）的轻量级端到端文本识别方法。首先, 引入 PP-LCNet 作为骨干网络进行特征提取; 接着, 在解码器之前设计三局部通道注意力模块, 通过 3 种不同尺度的一维卷积增强通道间的信息交互; 然后, 提出用局部增强注意力模块替换原解码器中的前馈网络部分, 通过深度可分离卷积增强文本特征空间关联性; 再后, 在各层解码器之后设计标记选择模块, 通过显著性标记突出文本特征, 减少无关像素的累积; 最后, 通过自回归解码方式预测出相应识别结果。将所提方法在 Total-Text、CTW1500 和 ICDAR2015 数据集上进行实验, 并与 6 种先进方法（ABCNet、MANGO、ABCNet v2、SPTS、SwinTextSpotter 和 TESTR）对比。相比于 SPTS 方法, 所提方法的推理速度分别提高了 19.6、35.7、21.1 f/s, 参数量减少了 70.7%, 证明了所提方法的有效性。

关键词: 注意力模块; 自回归解码; 轻量级网络; 单点定位; 文本识别; 端到端; 编码器; 解码器
中图分类号: TP391　　**文献标志码**: A　　**文章编号**: 1673−4785(2024)06−1503−15

中文引用格式: 曹锦纲, 张泽恩, 张铭泉. 改进单点定位模型的轻量级端到端文本识别方法 [J]. 智能系统学报, 2024, 19(6): 1503–1517.
英文引用格式: CAO Jingang, ZHANG Zeen, ZHANG Mingquan. A lightweight end-to-end text recognition method based on SPTS[J]. CAAI transactions on intelligent systems, 2024, 19(6): 1503–1517.

A lightweight end-to-end text recognition method based on SPTS

CAO Jingang[1,2], ZHANG Zeen[1,2], ZHANG Mingquan[1,2]

(1. School of Control and Computer Engineering, North China Electric Power University, Baoding 071003, China; 2. Engineering Research Center of intelligent Computing for Complex Energy Systems Ministry of Education, Baoding 071003, China)

Abstract: Addressing the problems of slow reasoning speed and the large number of model parameters in existing text spotting methods, this paper presents a lightweight end-to-end text spotting method based on single-point scene text spotting. First, PP-LCNet was introduced as the backbone network for feature extraction. Then, a three-local channel attention module was designed before the decoder, utilizing three different scales of one-dimensional convolution to enhance information interaction between channels. Next, a locally enhanced attention module was proposed to replace the feedforward network component in the original decoder, thereby improving the spatial correlation of text features using depthwise separable convolution. Subsequently, a token selector module was added after each decoder layer to highlight text features with saliency markers and reduce the accumulation of irrelevant pixels. Finally, recognition results were predicted using an autoregressive decoding method. The proposed method was tested on three datasets, namely, Total-Text, CTW1500, and ICDAR2015, and then compared with six advanced methods (ABCNet, MANGO, ABCNet v2, SPTS, SwinTextSpotter, and TESTR). Compared to the SPTS method, the proposed method achieved increments in inference speed of 19.6, 35.7, and 21.1 frames/s, respectively, and reduced the number of parameters by 70.7%, demonstrating its effectiveness.

Keywords: attention module; autoregressive decoder; lightweight network; single point position; text spotting; end to end; encoder; decoder

自然场景中的文本识别是计算机视觉领域的一项基本任务, 在自动驾驶、智能导航以及图像检索等领域取得了广泛应用[1]。传统方法通常将

收稿日期: 2023−07−13.　**网络出版日期**: 2024−09−10.
基金项目: 中央高校基本科研业务费专项项目（2021MS092）.
通信作者: 张铭泉. E-mail: mqzhang@ncepu.edu.cn.

文本检测与字符识别看成 2 个分离的子任务，文本检测的目的在于通过边界框定位文本实例，字符识别则是将检测到的文本区域转换为字符标签序列进行识别，该方法存在 2 点局限性：1) 文本检测的结果直接影响到文本识别的性能，单独优化检测与识别任务可能无法最大化提高整体模型的性能；2) 单独维护 2 个子任务会引入计算开销。

端到端的文本识别方法将文本检测任务和文本识别任务结合在统一的网络模型中，共享骨干网络的文本特征并根据文本识别的损失反向指导文本检测。现有的文本定位方法可以分为两大类，即两阶段的端到端文本识别方法和单阶段的端到端文本识别方法。两阶段方法[2-6]遵循先检测后识别的原则，首先通过检测器检测图像中的文本实例，然后利用基于感兴趣区域 (region of interest, RoI) 的连接器提取检测区域内的特征，接着将提取到的特征送入识别器得到结果。尽管这些方法已经取得了很大的进步，但仍存在以下 2 点局限：1) 检测器与识别器中间需要基于 RoI 的连接器 (如 RoI-Align[2]、RoI-Slide[3] 和 Bezier-Align[4-5] 等)，识别结果的精度依赖于连接器获取的文本边界精度；2) 面对任意形状的文本区域时，使用矩形的 RoI 可能会引入额外的背景，影响文字的识别，而复杂的多边形注释会使标注成本大大增加。单阶段的方法[7-10]为了减少两阶段过程中的误差累积，采用基于分割的方法共享检测和识别的共同主干，并在具有共享主干的并行多任务框架中完成训练，但需要启发式分组后处理来收集非结构化组件。

最近，SwinTextSpotter[6]、SRSTS[7]、SPTS[8]、TESTR[9] 和 TTS[10] 等方法将 Transformer[11] 引入到文本识别模型，效果显著。SwinTextSpotter 使用 Swin Transformer[12] 编码器作为检测器，通过识别转换机制将文本检测与识别 2 个阶段统一起来，通过识别损失反向指导文本定位。TESTR 采用基于 Transformer 的单编码器和双解码器联合文本定位与字符识别，同时设计了一个边界框到多边形框的引导方法，能够有效处理弯曲文本。SRSTS 通过共享特征图上的正锚点来连接并行的检测和识别 2 个分支，使识别不再着眼于精确的文本边界，进而减少两者的相互依赖，有效地降低了检测网络所需的注释成本。SPTS 基于 DETR[13]，将场景文本标记作为序列预测任务处理，表明使用低成本的单点甚至仅脚本注释是可行的。此外，TTS 中的朴素查询公式遵循通用的目标检测方法，未能考虑场景文本的独特特征，如位置和形状。SRSTS、SPTS、TTS 方法可以省去连接器和启发式的后处理，但存在以下问题：1) 推理速度慢，SPTS 使用 RTX 3090 GPU 在 Total-Text 数据集上的推理速度仅为 0.6 f/s，SwinTextSpotter 的推理速度仅为 2.9 f/s，即使目前最快的实时文本识别方法 ABCNet v2 也只有 9.8 f/s，推理速度方面仍有较大提升空间；2) 计算复杂度高，由于 Transformer 的时间复杂度是 $O(n^2)$，时间复杂度用于描述算法在输入规模 n 增加时所需的计算时间如何变化。在处理高分辨率图像对基于 Transformer 的文本识别方法上会使模型的内存和计算成本大大增加，影响整体识别效率。

考虑到 SPTS 成功使用低成本的单点注释完成端到端的文本识别以及 PP-LCNet[14] 的实时高效，本研究提出一种基于 SPTS 的轻量级的网络 LWSPTS(light-weight single-point scene text spotting)。首先，选择 PP-LCNet 作为骨干网络，使用逐通道卷积和逐点卷积代替标准卷积减少了参数量；然后，为了加强所得特征图通道间的信息交互，提出一种三局部通道注意力 (three localized channel attention, TLA) 模块将空间上的二维特征图平均池化为一维特征后采用 3 种并行的一维卷积进行信息融合；接着，在原有编码器的基础上提出用局部增强注意力 (local enhanced attention, LEA) 模块替换原有的前馈网络 (feed-forward networks, FFN) 以增强局部信息交互能力，突出小尺度文本特征；再后，在各层解码器之后设计标记选择模块，通过给不同信息赋予相应的权重以减少无关像素的累积；最后，解码器结合图片中编码后的信息采用自回归的方式预测出文本识别结果。

1　相关工作

在两阶段的识别方法中[15-22]，检测和识别模块依次执行，识别过程通常依赖于检测的输出，检测结果的微小偏差就可能导致识别的错误。单阶段的端到端文本识别方法则去掉了基于 RoI 的连接器，采用平行分支分割字符和文本实例。作为单阶段的端到端文本识别方法，CharNet[23] 在检测文本行的同时进行字符的分割，检测和识别直接在共享的特征图上完成，简化了端到端识别的流程，但与 Mask TextSpotter[17,20] 类似，训练时需要字符级的标注。为了减少训练成本，PGNet[24] 也采用单字符分割的方法，将一种字符点聚合的思想引入连接时序分类器（connectionist temporal classification, CTC），并提出图修正模块 (graph re-

finement module, GRM) 增加了识别精度, 极大地提高了识别的速度。MANGO[25] 通过位置感知掩码注意力 (position-aware mask attention, PMA) 模块将不同文本实例分配到不同特征映射通道上, 最后使用轻量级解码器识别出字符序列。以上基于分割的方法隔离了检测与识别 2 个子任务, 但是需要启发式分组处理识别结果。为了同时摆脱 RoI 和复杂后处理方法的限制, TESTR[9]、DeepSolo[26] 和 SPTS[8] 等提出利用 Transformer 的注意力机制进行文字特征的聚合后解码出对应的识别结果。TESTR 应用了 DETR 目标检测框架, 使用单编码器和双解码器联合文本定位与字符识别, 同时设计了一个边界框到多边形框的引导方法, 能够有效处理弯曲文本。针对 TESTR 无法有效地编码文本特征 (位置、形状、语义) 的问题, DeepSolo 设计了包含单解码器和多预测头的文本识别模型, 对于每个文本实例, 首先将字符序列表示为有序的点, 其中每个点都有明确的位置, 与文本顶部和底部边界的偏移量和类别属性, 并用可学习的显式点查询对其进行建模。TTS[10] 在 Deformable-DETR[27] 之后增加了循环神经网络 (recurrent neural network, RNN) 识别器, 在弱标注下显著提升了文本识别精度。为了进一步降低标注的成本, SPTS 遵循 Pix2seq[28] 的范式, 将场景文本标记作为序列预测任务处理, 在 DETR 的基础上采用自回归的方式预测文本序列, 表明使用低成本的单点注释是可行的, 但是该方法需要大量的计算资源, 推理速度较慢而且面对高分辨率图片不能有效提取小尺度文本的特征。

2　本研究模型

针对 SPTS 模型推理速度慢、小尺度文本检测精度低的情况, 本研究提出一种轻量级网络 LWSPTS, 如图 1 所示。首先, 模型的输入为原始文本图像, 经过 PP-LCNet 提取文本特征; 接着, 采用 TLA 模块对 $\boldsymbol{S}_5$ 特征图处理, 通过对每个特征通道赋予不同的权重, 以增强不同文本信息之间的关联性; 随后, 通过 Transformer 编码器生成文本序列信息; 之后, 通过标记选择器进行显著性标记, 以抑制无关像素累积的干扰; 最后, 通过 Transformer 解码器预测出文本识别结果。在训练过程中, 解码器采用并行解码的方式, 而在推理过程中, 采用自回归的方式进行解码, 图 1 中的实线表示训练过程, 虚线表示推理过程。

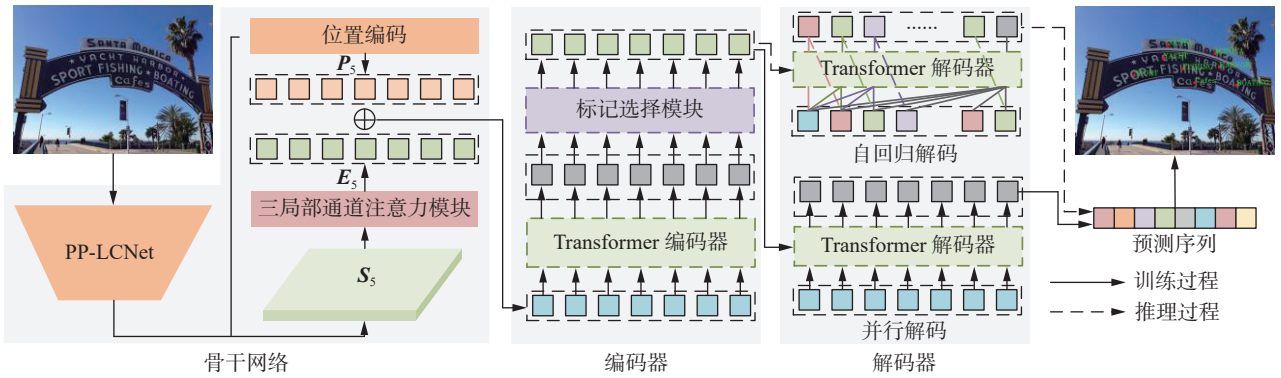


图 1　LWSPTS 模型结构

Fig. 1　Structure of LWSPTS model

2.1　轻量级骨干网络

PP-LCNet 使用深度可分离卷积 (depthwise separable convolution, DSC) 作为基本模块, 避免了元素相加与拼接, 保证了模型的推理速度, 结构如图 2 所示。首先经过一个 3×3 的卷积处理扩充维度; 接着经过 6 个 3×3 的 DSC 模块处理, 其中, 每 2 个模块为 1 层, 共有 3 层, 每经过 1 层特征图的通道数会加倍, 长宽会减半; 之后使用了 7 个 5×5 的 DSC 模块进行处理, 继续进行下采样操作, 最终输出的特征图 $\boldsymbol{S}_5$ 大小为原图的 1/16。

DSC 模块结构如图 2 右侧所示, 包含逐通道卷积 (depthwise convolution, DW) 和逐点卷积 (pointwise convolution, PW) 2 种。逐通道卷积的每个通道只与一个卷积核运算, 所以输出特征图的通道数与输入的通道数完全相同, 而逐点卷积使用 1×1 的卷积, 将上一步产生的特征图按通道进行加权组合。另外, 使用 H-Swish 激活函数替换 ReLU, 可以在不增加推理时间的情况下提高文本识别性能。除此之外, 本研究在最后 2 个 DSC 模块中加入压缩−激励网络 (squeeze-and excitation network, SE) 模块, 以加强通道间信息交互并保留更多文本细节信息。在 SE 模块中, 为了避免大量的指数运算, 使用 H-Sigmoid 代替 Sigmoid 激活函数, H-Sigmoid 使用线性函数简化了运算和求导过程。

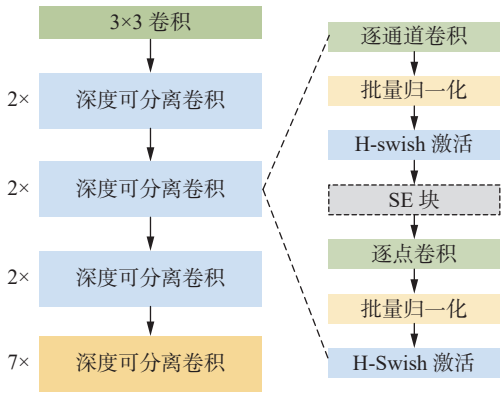


图 2　PP-LCNet 结构

Fig. 2　Structure of PP-LCNet

2.2　TLA 模块

各个通道的特征图都对应相应的文本特征，不同文本特征之间相互关联。为了充分学习特征通道之间的依赖关系，本研究提出 TLA 模块，结构如图 3 所示，首先将空间上的二维特征图平均池化并压缩为一维特征；接着采用 3 种并行的一维卷积进行特征提取，卷积核大小分别设置为 3、5、7；最后使用全连接层融合并经过 H-Sigmoid 激活得到融合权重，与原始输入特征相乘后即为输出。计算过程为

$$\boldsymbol{E}_5=\sigma(\mathrm{Linear}(f_{\{W_3,W_5,W_7\}}(\boldsymbol{S}_5)))\otimes\boldsymbol{S}_5 \tag{1}$$

式中：$\boldsymbol{S}_5$ 表示 TLA 的输入；$\boldsymbol{E}_5$ 表示 TLA 的输出；$\sigma(\cdot)$ 表示 H-Sigmoid 激活函数；Linear(·) 表示将三维映射到一维的全连接层；$f_{\{W_3,W_5,W_7\}}(\cdot)$ 表示经过 3 种卷积处理后的特征；$\otimes$ 表示逐像素相乘。

由式 (1) 可以看出 TLA 的整体流程，经过并行卷积处理后，通过全连接降维并激活得到融合权重，再与原特征相乘：

$$f_{\{W_i\}}(\boldsymbol{S}_5)=\boldsymbol{W}_i\times g(\boldsymbol{S}_5),i=3,5,7 \tag{2}$$

式中：$\boldsymbol{W}_i$ 表示输入通道和输出通道均为 1、卷积核大小为 i 的一维卷积；$g(\cdot)$ 表示全局平均池化操作。式 (2) 给出了 3 种一维卷积的处理过程。

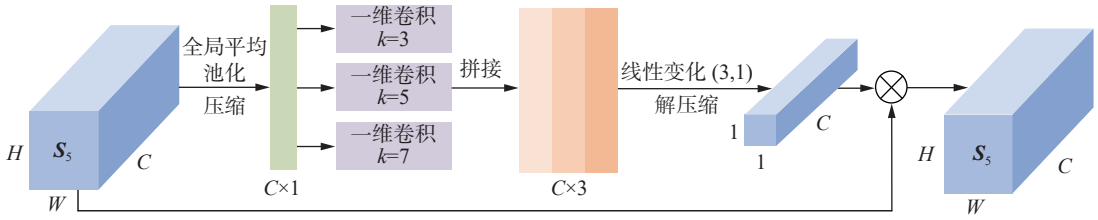


图 3　TLA 模块结构

Fig. 3　Structure of TLA module

2.3　Transformer 编码器

LWSPTS 基于 DETR 设计了一种轻量级编码器−解码器框架，在编码器部分做出以下改进：第一，使用 LEA 替换原解码器中的前馈神经网络 (feed forward neural network, FFN)，可以充分提取文本的局部信息，增强相邻文本之间的空间关联性；第二，为了加快模型的推理速度，减少模型的参数量，本研究将编码器的层数由 6 减小为 3。整体编码过程如下。

编码器由多头自注意力 (multi-head self-attention)、归一化层 (LayerNorm) 和 LEA 模块 3 部分组成，编码器的 2 个输入分别是特征图 $\boldsymbol{E}_5\in\mathbf{R}^{H\times W\times C}$（$H$ 和 W 分别为原始图像的高和宽，C 表示输入的维度）和对应的位置编码 (positional encoding)$\boldsymbol{P}_5\in\mathbf{R}^{H\times W\times C}$，多头自注意力的键 (key, $\boldsymbol{K}$)$\in\mathbf{R}^{N\times C}$、$\boldsymbol{Q}$(query)$\in\mathbf{R}^{N\times C}$ 是 $\boldsymbol{E}_5$ 与 $\boldsymbol{P}_5$ 相加的结果，其中，N=H×W，是对特征图维度进行压缩后的结果，$\boldsymbol{E}_5$ 为三局步通道注意力模块的输出，$\boldsymbol{V}$(value)$\in\mathbf{R}^{N\times C}$ 是特征图 $\boldsymbol{E}_5$ 的直接输入；接着，多头自注意力的输出与原特征图相加后经过层归一化处理输入到 LEA；最后，LEA 的输出与 LEA 的输入经过残差连接后再进行层归一化处理，输出结果与特征图尺寸相同。LWSPTS 的编码部分包含 3 个相同的编码器层，每个编码器层的输出经过标记选择模块处理后作为下一个编码器层的输入，第 3 个编码器层的最终输出被送入解码器部分。

LEA 模块　在编码器中，为了增强相邻序列的相关性建立长期依赖关系，本研究提出 LEA 模块，结构如图 4 所示。LEA 处理过程如下。

1) 对输入序列 $\boldsymbol{X}_{\mathrm{h}}\in\mathbf{R}^{N\times C}$ 进行维度扩张得到 $N\times1\times C$，接着还原成与输入图像尺度相同的特征 $\boldsymbol{X}_{\mathrm{s}}\in\mathbf{R}^{H\times W\times C}$，具体过程为

$$\boldsymbol{X}_{\mathrm{s}}=\mathrm{Reshape}(\mathrm{Unsqueeze}(\boldsymbol{X}_{\mathrm{h}})) \tag{3}$$

式中：$\boldsymbol{X}_{\mathrm{h}}$ 表示 LEA 模块的输入；$\boldsymbol{X}_{\mathrm{s}}$ 表示新形成的特征图；Unsqueeze(·) 表示维度扩张操作；Reshape(·) 表示重塑操作。把 $N\times1\times C$ 重塑为 $H\times W\times C$，可以在不改变向量数目的情况下改变向量的形状。

2) 采用逐点卷积进行通道间信息交互的同时增加通道维度，得到 $\boldsymbol{X}_{\mathrm{p}}^{\mathrm{h}}\in\mathbf{R}^{H\times W\times C_{\mathrm{m}}}$，$C_{\mathrm{m}}$ 为表示中间维度的超参数，初始设置为 1 024，调参部分后面介绍，具体过程为

$$\boldsymbol{X}_{\mathrm{p}}^{\mathrm{h}}=\tau(\mathrm{BN}(\mathrm{PWConv}(\boldsymbol{X}_{\mathrm{s}}))) \tag{4}$$

式中: $\boldsymbol{X}_{\mathrm{p}}^{\mathrm{h}}$表示经过逐点卷积处理后的结果; $\tau(\cdot)$ 表示 H-Swish 激活函数; BN(·) 表示批量归一化 (BatchNorm) 操作; PWConv(·) 表示卷积核大小为 1×1、步幅为 1、填充为 0 的卷积操作。

3) 对重新得到的特征图进行卷积核大小为 k 的深度卷积运算, 以加强与周围 k^2-1 个位置信息的相关性, 得到 $\boldsymbol{X}_{\mathrm{d}}\in\mathbf{R}^{H\times W\times C_{\mathrm{m}}}$, 公式表示为

$$\boldsymbol{X}_{\mathrm{d}}=\tau(\mathrm{BN}(\mathrm{DWConv}(\boldsymbol{X}_{\mathrm{p}}^{\mathrm{h}}))) \tag{5}$$

4) 再次利用逐点卷积降维后得到 $\boldsymbol{X}_{\mathrm{p}}^{\mathrm{l}}\in\mathbf{R}^{H\times W\times C}$, 接着压缩为原始维度的序列 $\boldsymbol{X}_{\mathrm{h}}^{\mathrm{l}}\in\mathbf{R}^{N\times C}$。在每次卷积操作之后, 都会增加批量归一化和 H-Swish 激活, 具体流程为

$$\boldsymbol{X}_{\mathrm{p}}^{\mathrm{l}}=\tau(\mathrm{BN}(\mathrm{PWConv}(\boldsymbol{X}_{\mathrm{d}}))) \tag{6}$$

$$\boldsymbol{X}_{\mathrm{h}}^{\prime}=\mathrm{Flatten}(\boldsymbol{X}_{\mathrm{p}}^{\mathrm{l}}) \tag{7}$$

式中: $\boldsymbol{X}_{\mathrm{p}}^{\mathrm{l}}$表示经过逐点卷积压缩通道后的结果; $\boldsymbol{X}_{\mathrm{h}}^{\prime}$表示经过降维处理后的结果, 也作为 LEA 模块的输出; Flatten(·) 表示维度压缩处理。

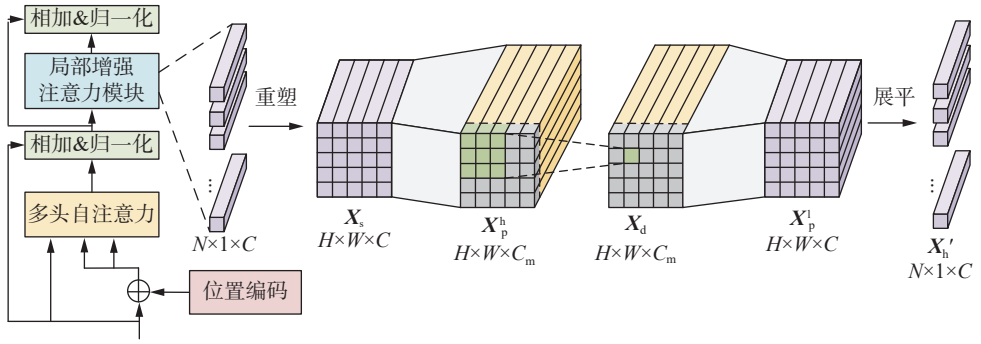


图 4　LEA 模块结构

Fig. 4　Structure of LEA module

LWSPTS 通过 LEA 模块结合了卷积神经网络强化局部信息与 Transformer 提取全局信息的优势, 增强了文本表征能力。

多头自注意力　多头自注意力的 3 个输入分别为 $\boldsymbol{Q}$、$\boldsymbol{K}$、$\boldsymbol{V}$, $\boldsymbol{Q}$、$\boldsymbol{K}$ 是特征图与位置编码相加的结果, $\boldsymbol{V}$ 是特征图直接输入, 使用缩放点积注意力 (Scaled Dot-product Attention) 处理后的输出与输入序列尺度相同, 均为 $\mathbf{R}^{N\times C}$。具体过程为

$$\mathrm{MHA}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V})=[\mathrm{head}_1,\mathrm{head}_2,\cdots,\mathrm{head}_h]\boldsymbol{W}^{\mathrm{O}} \tag{8}$$

$$\mathrm{head}_i=\mathrm{Attention}(\boldsymbol{Q}\boldsymbol{W}_i^q,\boldsymbol{K}\boldsymbol{W}_i^k,\boldsymbol{V}\boldsymbol{W}_i^v) \tag{9}$$

式中: MHA(·) 表示多头注意力操作; $\mathrm{head}_i(\cdot)$ 表示第 i 个注意力头的操作, 共分为 h 个头分别计算, 每个头的维度为 $\mathbf{R}^{N\times C/h}$; N 表示 $\boldsymbol{Q}$ 向量的维度; C 表示输入的维度; Attention(·) 表示缩放点积注意力操作; $\boldsymbol{W}_i^q$、$\boldsymbol{W}_i^k$、$\boldsymbol{W}_i^v$表示 $C\times C/h$ 维度的权重矩阵, $\boldsymbol{W}^{\mathrm{O}}$ 表示 $C\times C$ 维度的权重矩阵。

2.4　标记选择模块

为了对编码后的信息进行筛选, 弥补由于编码器层数减少导致文本信息编码不充分的情况, 本研究提出一种标记选择器, 根据包含的文本信息量对标记进行重要性评价, 标记选择模块如图 5 所示。

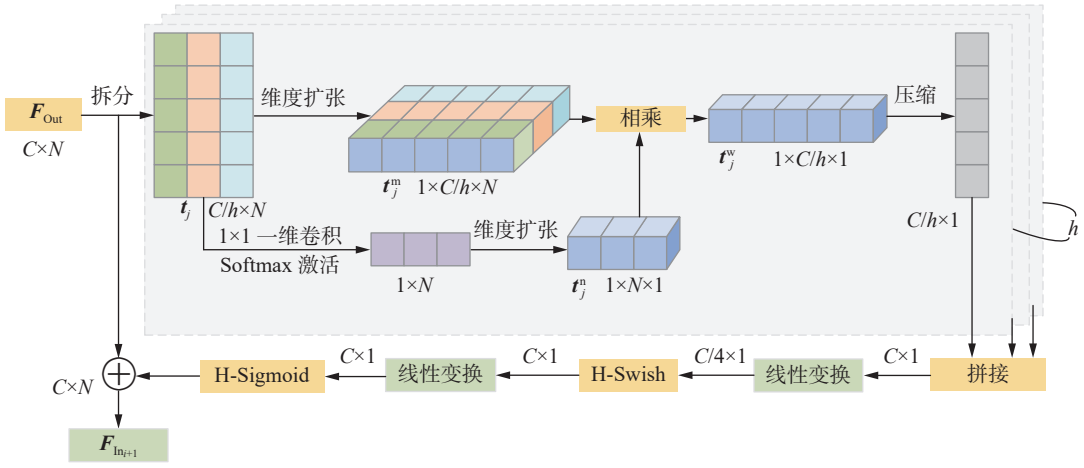


图 5　标记选择模块结构

Fig. 5　Structure of token selection module

1) 标记选择器的输入为上一层编码器的结果，对结果按通道拆分，对拆分结果分别处理，公式表示为

$$\boldsymbol{F}_{\mathrm{Out}_i}=[\boldsymbol{t}_1\ \ \boldsymbol{t}_2\ \ \cdots\ \ \boldsymbol{t}_h],\boldsymbol{F}_{\mathrm{Out}_i}\in\mathbf{R}^{N\times C}\tag{10}$$

式中：$\boldsymbol{t}_j\in\mathbf{R}^{N\times C/h},j=1,2,\cdots,h$，为拆分后的结果，$h$ 为多头自注意力的头数。

2) 对拆分后的结果 $\boldsymbol{t}_j$ 进行 2 种处理，第一是进行维度扩张得到特征 $\boldsymbol{t}_j^{\mathrm{m}}\in\mathbf{R}^{1\times C/h\times N}$，第二是通过 1×1 的卷积处理融合成单通道特征后进行维度扩张得到 $\boldsymbol{t}_j^{\mathrm{n}}\in\mathbf{R}^{1\times N\times 1}$，公式表示分别为

$$\boldsymbol{t}_j^{\mathrm{m}}=\mathrm{Unsqueeze}(\boldsymbol{t}_j)\tag{11}$$

$$\boldsymbol{t}_j^{\mathrm{n}}=\mathrm{Unsqueeze}(\varepsilon(\mathrm{Conv}_1(\boldsymbol{t}_j)))\tag{12}$$

式中：$\varepsilon(\cdot)$ 表示 Softmax 激活函数；$\mathrm{Conv}_1(\cdot)$ 表示输入通道为 C/h、输出通道为 1、卷积核大小为 1 的一维卷积操作。

3) 对 $\boldsymbol{t}_j^{\mathrm{m}}$ 和 $\boldsymbol{t}_j^{\mathrm{n}}$2 个特征相乘，所得结果压缩后得到对应位置的权重 $\boldsymbol{t}_j^{\mathrm{w}}\in\mathbf{R}^{C/h\times 1}$。

$$\boldsymbol{t}_j^{w}=\mathrm{Squeeze}(\boldsymbol{t}_j^{\mathrm{m}}\times\boldsymbol{t}_j^{\mathrm{n}})\tag{13}$$

4) 对每个头的权重进行拼接，得到总体权重 $\boldsymbol{w}_i=\left[\boldsymbol{t}_1^{*}\ \ \boldsymbol{t}_2^{W}\ \ \cdots\ \ \boldsymbol{t}_j^{*}\right],j=1,2,\cdots,h$，将权重进行通道变化后与原输入 $\boldsymbol{F}_{\mathrm{Out}_i}$ 相加后得到输出。

$$\boldsymbol{F}_{\mathrm{In}_{i+1}}=\sigma(\mathrm{Linear}_2(\tau(\mathrm{Linear}_1(\boldsymbol{w}_i))))+\boldsymbol{F}_{\mathrm{Out}_i}\tag{14}$$

2.5　Transformer 解码器

解码器包含 N=3 个解码器层，每个解码器层包括掩码多头自注意力、多头注意力和前馈网络(FFN)3 个主要模块，如图 6 所示。

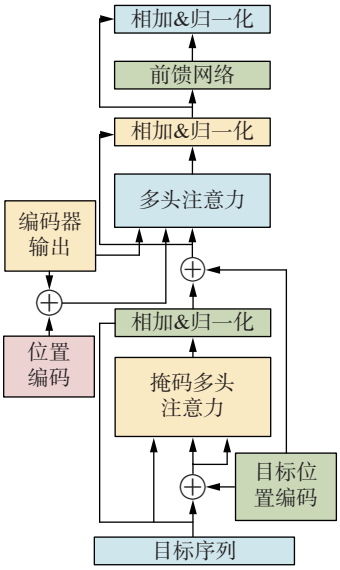


图 6　解码器结构

Fig. 6　Structure of decoder

掩码多头自注意力　多头自注意力的输入 $\boldsymbol{Q}\in\mathbf{R}^{L\times C}$ 为目标序列 (target)，L 为目标序列的长度，$\boldsymbol{K}\in\mathbf{R}^{L\times C}$ 和 $\boldsymbol{V}\in\mathbf{R}^{L\times C}$ 均为目标位置编码 (query position) 与目标序列之和，经过缩放点积注意力处理的输出与输入序列尺度相同。此外，在解码器中引入下三角掩码矩阵，对权重矩阵进行处理，把未来的位置设置为负无穷大，使解码器在 t 时刻只能获取到 t 时之前的输出信息。通过引入掩码矩阵，解码器可以同时输出所有时间的预测结果，避免顺序输出导致训练缓慢，实现了训练过程的高度并行。

多头注意力　该模块的输入相对复杂，$\boldsymbol{Q}\in\mathbf{R}^{L\times C}$ 为掩码多头自注意力的输出与目标位置编码之和，$\boldsymbol{K}\in\mathbf{R}^{N\times C}$ 为编码器的输出 (memory) 与对应位置编码 (positional encoding) 之和，$\boldsymbol{V}\in\mathbf{R}^{N\times C}$ 为编码器的输出，经过多头注意力后输出序列与向量 $\boldsymbol{Q}$ 尺度相同。

前馈网络　前馈网络由 2 个全连接层组成，在 2 层之间引入了 H-Swish 激活函数，公式表示为

$$\mathrm{FFN}(x)=\tau(\boldsymbol{x}\boldsymbol{W}_1+\boldsymbol{b}_1)\boldsymbol{W}_2+\boldsymbol{b}_2\tag{15}$$

式中：$\boldsymbol{x}$ 表示 FFN(·) 的输入；$\boldsymbol{W}_1\in\mathbf{R}^{C\times C_{\mathrm{m}}}$、$\boldsymbol{W}_2\in\mathbf{R}^{C_{\mathrm{m}}\times C}$ 分别为 2 个权重矩阵；$\boldsymbol{b}_1\in\mathbf{R}^{C_{\mathrm{m}}}$、$\boldsymbol{b}_2\in\mathbf{R}^{C}$ 为对应的偏置；C 表示嵌入输出的维度；C_{m} 表示 2 个线性变化中间的维度；FFN(·) 可以对多头注意力得到的特征进行非线性处理，通过挖掘各维度间的非线性关系，增强了特征的表示能力。

训练阶段的并行解码器和推理阶段的自回归解码器在网络结构上是相同的，二者的差别在于输入的目标序列。在训练阶段，输入的目标序列由开始符<SOS>和图像中的所有文本实例组成，同时输入到解码器，通过下三角掩码矩阵，可以预测出所有时刻的结果，输出的结果为所有文本实例以及结束符<EOS>。在推理阶段，解码器只能按步骤依次预测出单个字符，对于某次解码过程，输入的目标序列为前面步骤的输出结果，本次解码器的输出又成为后面步骤的输入，直到预测出结束标识符<EOS>或者解码器序列长度达到最大长度。随后，预测出来的序列又被划分为多个固定长度的文本实例。最后，将标记翻译成原图对应的坐标和文本，输出识别结果。

2.6　目标序列构建

目标序列对应图 6 中解码器的输入，由开始标识符、随机排序的文本实例和结束标识符 3 部分组成，1 个文本实例主要包括文本中心点的坐标和转录文本 2 部分。首先将输入图像压缩到特定长度 n_{bins}，然后将文本中心点坐标 (x,y) 统一离散化为 $[1,n_{\mathrm{bins}}]$ 之间的整数，具体公式为

$$\begin{cases} \mathrm{Trans}(x_i)=\lfloor x_i/w\times n_{\mathrm{bins}}\rfloor \\ \mathrm{Trans}(y_i)=\lfloor y_i/h\times n_{\mathrm{bins}}\rfloor \end{cases} \tag{16}$$

式中：Trans(·) 表示坐标离散化；w 表示输入图像的宽度；h 表示输入图像的高度；n_{bins} 表示压缩后指定长度；x_i、y_i 分别表示横纵坐标；⌊⌋表示向下取整操作。

由于转录文本本身是离散的，每个字符本身对应一个类别，对于不同长度的文本实例的转录结果长度也是不同的，为了避免因文本长度不同导致的错位问题，使用<PAD>标记将文本序列到填充到固定长度 l_{tr}。由于文本区域的中心点坐标长度为 2，所以 1 个文本实例长度固定为 $2+l_{\mathrm{tr}}$。接着随机插入不同的文本实例，形成长度为 $(2+l_{\mathrm{tr}})\times n_{\mathrm{ti}}$ 的序列，n_{ti} 表示图像中所包含文本实例的个数。最后将<SOS>和<EOS>标记插入到序列的头部和尾部，分别表示序列的开始和结束。

设使用的字符标记有 n_{cls} 类，则用于构建目标序列的词典共包含（$n_{\mathrm{bins}}+n_{\mathrm{cls}}+3$）个类别，$n_{\mathrm{cls}}$ 设置为 96，表示包含 96 个英文文本和字符，n_{bins} 设置为 1 000，l_{tr} 和 n_{ti} 分别设置为 25和 60，表示图片的中心点坐标会映射到 0~1 000 的整数，单个文本实例最多包含为 25 个字符，单张图片最多包含 60 个文本实例，具体对应情况见表 1。

表 1　n_{bins}=1 000 的目标序列词典

Table 1　Target sequence dictionary for n_{bins}=1 000

序号	文本字符
0~999	(*x*,*y*)
1 000~1 019	<space>!"#$%&\'()*+,-/0123
1 020~1 039	456 789:;<=>?@ABCDEFG
1 040~1 059	HIJKLMNOPQRSTUVWXYZ[
1 060~1 079	\\]^_`abcdefghijklmno
1 080~1 094	pqrstuvwxyz{|}~
1 095	NaN
1 096	<PAD>
1 097	<SOS>
1 098	<EOS>

表 1 为 n_{bins}=1 000 时目标序列词典对应情况，左侧为序号，右侧为对应的字符标记，其中，“0~999”表示离散化的坐标，“1 000~1 094”对应不同的字符，“1 095”表示未识别结果，“1 096”表示填充，“1 097”表示开始符号，“1 098”表示结束符号。以图 7 为例，宽度为 240 像素，高度为 320 像素，最下方“*CHICAGO*”实例的原始标注为“118.32” “238.40” “*C*” “*H*” “*I*” “*C*” “*A*” “*G*” “*O*”，根据其转换流程，得到的目标序列为“493” “745” “1 035” “1 040” “1 041” “1 035” “1 033” “1 039” “1 047” “1 096” “1 096”⋯ “1 096”。其中，“493”和“745”为原始中心点根据式 (16) 转换后新的坐标，“1 035” —“1 047”为文本字符根据表 1 转录的结果，“1 096”为填充标记，通过填充使每个文本实例的长度固定为 27。

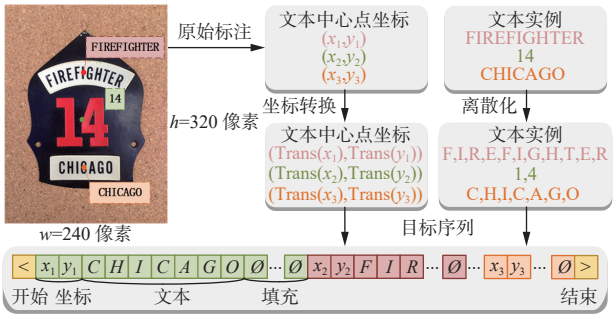


图 7　目标序列结构

Fig. 7　Diagram of target sequence construction

3　实验结果与分析

3.1　数据集

SynthText150K 数据集[29]　包含 94 723 张多方向文本的图像和 54 327 张弯曲文本的图像。每张图片都有单字级和字符级的边界框标注以及文本序列的标注。

ICDAR2015 数据集[30]　由 1 000 张训练图像和 500 张测试图像组成，包含了复杂背景、多尺度、多方向的文本实例。

CTW1500 数据集[31]　由 1 000 张训练图像和 500 张测试图像组成的多语言场景文本数据集。这些图像中包含了许多弯曲文本实例，每个文本实例是由 14 个点组成的多边形标记。适用于检测模型在弯曲文本条件/场景下的性能。

Total-Text 数据集[32]　共包含 1 555 张图像，分为 1 255 张训练图像和 300 张测试图像，包含多方向文本和弯曲文本，用多边形框代替传统的矩形框进行标注。

ICDAR2017-MLT 数据集[33]　包含 7 200 张训练图像、1 800 张验证图像和 9 000 张测试图像，共包含中、日、韩、英、法等 9 种语言，其图像采集自各种各样的场景，其中的文本的长度、字体、尺寸、颜色千变万化，同时还包含了许多真实场景的噪声，包括光照、遮挡、倾斜和文字堆叠等。

3.2　实验设置

本研究训练和测试过程是在 Ubuntu18.04 系统下进行的，使用 2 块 32 GB 显存的 NVIDIA V100 GPU 进行训练，1 块 24 GB 显存的 RTX 3090 GPU 进行测试，所用 PyTorch 的版本为 1.10。

数据增强　1) 随机缩放，在训练过程中，将输入图像的短边缩短到 480 像素至 896 像素范围内，间隔为 32，同时长边保持在 1 600 像素以内；在推理阶段，将短边尺寸调整为 1 000 像素，同时保持长边短于 1 824 像素。2) 在 [−15°,15°] 进行随机旋转。3) 随机裁剪。除此之外，还在训练过程中随机改变图像的亮度、对比度和饱和度。

预训练　本研究模型首先在 SynthText150K、ICDAR2017-MLT、Total-Text 和 ICDAR2015 4 个数据集上进行 100 次迭代，采用 AdamW 优化器进行优化，并将初始学习率设置为 5×10^{-4}，100 次迭代后线性衰减为 1×10^{-5}，批量大小设置为 4。

微调　接着在 CTW1500、Total-Text 和 ICDAR-2015 这 3 个目标数据集上分别进行 300 次迭代，对预训练得到的模型进行微调，此过程中学习率固定为 1×10^{-5}，批量大小设置为 2。

3.3　对比实验

为了验证本研究所提方法的有效性，在 Total-Text、CTW1500 和 ICDAR2015 这 3 个数据集上将 LWSPTS 方法与 ABCNet[4]、MANGO[25]、ABCNet v2[5]、SPTS[8]、SwinTextSpotter[6] 和 TESTR[9]6 种方法进行比较，结果见表 2，表 2 中：None 表示不使用词汇表下的识别准确率；Full 表示使用测试集中出现的所有单词组成的词汇表下的识别准确率；模型每秒能够处理的帧数，反映模型的处理速度；A_S(strong)、A_W(weakly)、A_G(generic)表示使用 3 类不同词汇表下识别的准确率，其中，A_S 表示从数据集选取的 100 个单词组成的强词汇表，A_W 表示训练集和测试集中所有单词组成的词汇表，A_G 表示包含 9 万个单词的通用词汇表。ABCNet 首次使用参数化的贝塞尔曲线来适应任意形状的文本并设计了 Bezier-Align 层，以精确提取任意形状的文本特征。MANGO 设计了掩码关注模块，以生成对每个文本实例及其字符的关注权重，而且只用粗略的位置信息和文本注释便可以进行端到端的训练。ABCNet v2 在 ABCNet 的基础上简化了后处理过程并采用简单而有效的坐标卷积编码卷积滤波器的位置。SPTS 将场景文本识别作为序列预测任务来处理，将所需的检测和识别结果表示为一系列离散标记，并使用解码器预测该序列。SwinTextSpotter 使用带有动态头的 Transformer 编码器作为检测器，设计了一种识别转换机制统一检测和识别 2 个任务，并通过识别损失指导文本定位。TESTR 使用一种单编码器和双解码器结构，用于联合文本框控制点回归和字符识别。

表 2　7 种模型在 Total-Text、CTW1500 和 ICDAR2015 数据集上的定量比较

Table 2　Quantitative comparison of the seven models on the Total-Text, CTW1500 and ICDAR2015 datasets

模型	Total-Text			CTW1500			ICDAR2015			
	None/%	Full/%	处理速度/(f/s)	None/%	Full/%	处理速度/(f/s)	A_S/%	A_W/%	A_G/%	处理速度/(f/s)
ABCNet	64.2	75.7	8.9	45.2	74.1	11.2	79.2	74.1	66.8	7.4
MANGO	72.9	83.6	4.1	58.9	78.7	8.4	81.8	78.9	67.3	3.9
ABCNet v2	70.4	78.1	9.8	57.5	77.2	12.7	82.7	78.5	73.0	8.6
SPTS	74.2	82.4	0.6	**63.6**	**83.8**	0.8	77.5	70.2	65.8	1.2
SwinTextSpotter	**74.3**	**84.1**	2.9	51.8	77.0	5.3	83.9	77.3	70.5	4.5
TESTR	73.3	83.9	7.8	56.0	81.5	15.9	**85.2**	**79.4**	**73.6**	9.8
LWSPTS	69.5	79.1	**20.2**	57.5	78.1	**36.5**	79.2	73.6	66.4	**22.3**

注：黑体表示每列最优值。

3.3.1　各数据集定量分析

在 Total-Text 数据集上，LWSPTS 方法的 None 和 Full 分别为 69.5% 和 79.1%，与 ABCNet 方法相比超出 5.3 百分点和 3.4 百分点，与 ABCNet v2 相比 None 指标低 0.9 百分点，Full 指标高 1.0 百分点。SPTS、SwinTextSpotter 和 TESTR 3 种方法的识别精度类似，None 为 74.2%、74.3% 和 73.3%，分别超出 LWSPTS 方法 4.7 百分点、4.8 百分点和 3.8 百分点。在速度方面，LWSPTS 方法达到了 20.2 f/s，与目前速度较快的 ABCNet v2 相比提高了 10.4 f/s，与 SPTS、SwinTextSpotter、TESTR 方法相比，分别提高了 19.6、17.3、12.4 f/s，主要原因在于 LWSPTS 使用轻量级骨干网络 PP-LCNet 进行特征提取，在减少解码器和编码器层数的同时增加了标记选择模块，提升速度的同时保证了识别精度。LWSPTS在识别精度方面与

ABCNet v2 方法类似, 但是在速度方面比现有最快的 ABCNet v2 提升一倍左右。

在 CTW1500 数据集上, LWSPTS 方法的 None 和 Full 分别为 57.5% 和 78.1%, 分别超出 ABCNet 和 SwinTextSpotter 方法 12.3 百分点、4.0 百分点和 5.7 百分点、1.1 百分点, 与 MANGO 方法相比, None 和 Full 分别下降了 1.4 百分点和 0.6 百分点, 与 ABCNet v2 方法相比, None 指标相同为 57.5%, Full 指标提升了 0.9 百分点, 但是 LWSPTS 与 SPTS 仍有一定差距, None 相差 6.1 百分点, Full 相差 5.7 百分点。在速度方面, 本研究方法达到了最快的 36.5 f/s, 超出位于第 2 的 TESTR 方法 19.7 f/s, 超出位于第 3 的 ABCNet v2 方法 23.8 f/s。

在 ICDAR2015 数据集上, LWSPTS 方法的 A_S、A_W 和 A_G 分别为 79.2%、73.6% 和 66.4%, 与 SPTS 相比分别提升了 1.7 百分点、3.4 百分点和 0.6 百分点, 主要是由于 SPTS 方法是直接从整个图像特征中解码, 没有专门的 RoI 操作, 导致对小尺度文本识别情况较差, 而本研究方法在骨干网络之后设计了 TLA 模块, 对每个特征通道赋予不同的权重, 增强了不同文本信息之间的关联性, 而且在解码器中使用 LEA模块替换 FFN, 突出了局部信息, 增强了小尺度文本的表征能力。在速度方面, 本研究达到了 22.3 f/s, 相比于 ABCNet v2 和 TESTR 分别提升了 13.7 f/s 和 12.5 f/s。

3.3.2　Total-Text 数据集定性分析

图 8 分别为原图、ABCNet、MANGO、ABCNet v2、SPTS、SwinTextSpotter、TESTR 和 LWSPTS 方法的识别结果。除图 8(e)、(h) 使用单个点描述文本区域中心外, 其他方法均使用任意多点描述文本区域边界, 图 8(b)、(d)、(g) 使用蓝色表示预测的文本区域, 图 8(c)、(f) 则使用多种颜色表示。

由图 8(e) 可以看出, SPTS 方法的结果有 2 处文本未被检测, 由图 8(c) 可以看出, MANGO 方法出现 1 处漏检, 其他方法均能正确预测文本区域。从识别的准确度来看, 由图 8(b) 可知, ABCNet 方法对左上角“POKKA”单词产生了 3 种识别结果, 分别为“PO” “KKA”和“DOKKA”,说明 ABCNet 方法虽然能检测到文本实例, 但部分字母之间缺乏空间关联性, 导致识别的准确性有待提升; 对于下方“HOME” “ROASTED”和“COFFEE”3 处文本实例, 挑战在于字符“O”为艺术字形式, ABCNet 方法只能正确识别出“COFFEE”, 将“HOME”和“ROASTED”分别识别为“HODME”和“ROAATED”, 错误原因在于网络特征提取能力不足, 未能处理好不规则文本字符造成的影响。由图 8(d)、(g) 可知, ABCNet v2 方法和 TESTR 方法均出现了 1 处错误识别的情况, 分别为“DASTED”和“TOKKA”。由图 8(c)、(f) 可知, MANGO 和 SwinTextSpotter 方法的结果均能正确识别出检测到的文本实例, 但是检测精度有待提高。由图 8(e)、(h) 可知, 只有 SPTS 和 LWSPTS2 种方法能正确识别全部的文本实例, 说明 LWSPTS 在弯曲文本的定位和识别方面与 SPTS 方法的效果相似, 而且对于不规则字符的识别有良好的鲁棒性。

(a) 原图　(b) ABCNet

(c) MANGO　(d) ABCNet v2

(e) SPTS　(f) SwinTextSpotter

(g) TESTR　(h) LWSPTS

图 8　Total-Text 数据集识别结果

Fig. 8　Recognition results on Total-Text dataset

3.3.3　CTW1500 数据集定性分析

图 9 为 7 种方法在 CTW1500 数据集上的检测结果, 与图 8 中场景文本图片只包含弯曲文本实例不同, 图 9(a) 中同时包含倾斜、弯曲和水平的文本 (分别位于图像的左侧、中间上部以及中间的中下部), 可以衡量不同模型对任意方向文本的识别能力。

(a) 原图　(b) ABCNet
(c) MANGO　(d) ABCNet v2
(e) SPTS　(f) SwinTextSpotter
(g) TESTR　(h) LWSPTS

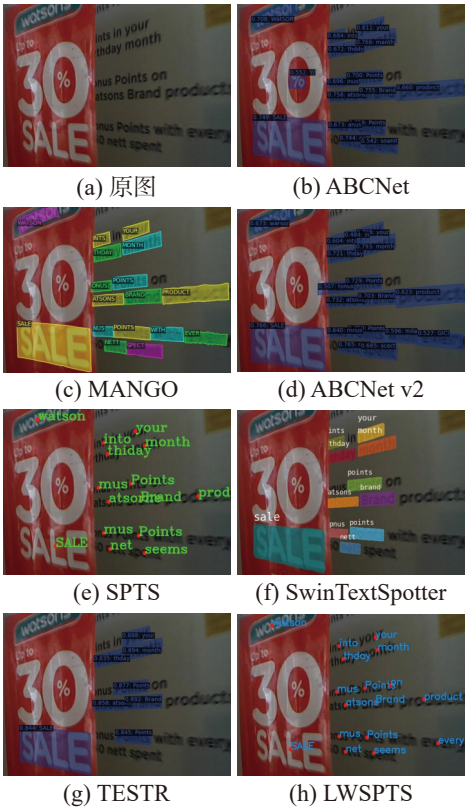
图 9　CTW1500 数据集识别结果

Fig. 9　Recognition results on CTW1500 dataset

首先对于倾斜文本实例(“ALASKAN” “FUDGE”和“COMPANY”), 由图 9(f)、(g) 左侧可知, SwinTextSpotter 和 TESTR 方法的结果均出现了 1 处漏检, 为最下方的“COMPANY”, 其他方法均能正确检测。由图 9(b) 可知, ABCNet 方法将“FUDGE”识别为“PUDCE”, 有 2 个字符识别错误, 其他方法对倾斜文本均能正确识别。接着, 对于弯曲文本实例“EVERYTHING”, 7 种方法能准确识别。最后, 对中间部分的水平方向文本(“SWEET” “AND”和“DELICIOUS”), 由图 9(b)、(g) 可知, ABCNet 和 TESTR 2 种方法的结果均出现了 1 处漏检的现象, 检测到的文本均能正确识别, 而图 9(c) 的 MANGO 方法出现了大规模漏检。由图 9(d)、(e)、(f) 可知, 虽然水平文本能识别, 但 ABCNet v2、SPTS 和 SwinTextSpotter 3 种方法对文本尺度变化不敏感, 将“SWEET”和“AND”2 个单词连成一体进行识别。只有 LWSPTS 方法能同时识别倾斜、弯曲和水平方向的文本, 并能根据实际场景精确划分不同文本实例, 说明 LWSPTS 方法对任意方向的文本识别具有一定优势。

3.3.4　ICDAR2015 数据集定性分析

图 10 为 7 种方法在 ICDAR2015 数据集上的识别结果, 该数据集中图像的特点为文本分布密集, 各实例之间间隔小, 存在大面积遮挡现象, 而且图像模糊也给识别任务带来了挑战。从整体来看, 由图 10(f) 和图 10 (g) 可知, SwinTextSpotter 和 TESTR 方法的漏检程度明显高于其他方法, 由图 10(b)、(c)、(d)、(e)、(h) 可知, ABCNet、MANGO、ABCNet v2、SPTS 和 LWSPTS 方法分别检测出 16、17、18、15、17 处文本, 其中, ABCNet 和 ABCNet v2 方法对遮挡文本识别的准确率偏低, 比如 ABCNet 方法中的“thddy” “manth” “soant”和 ABCNet v2 方法中的“mila” “GICI” “sceet”, 均存在部分字符识别错误, SPTS 方法出现了“thiday”和“seems”2 处明显错误, LWSPTS 方法识别准确率优于 SPTS 方法, 但略低于 MANGO 方法。从识别文本实例数量和精度 2 方面可以看出, LWSPTS 方法处理小尺度文本和密集文本的性能优于大部分主流方法。



(a) 原图　(b) ABCNet
(c) MANGO　(d) ABCNet v2
(e) SPTS　(f) SwinTextSpotter
(g) TESTR　(h) LWSPTS

图 10　ICDAR2015 数据集识别结果

Fig. 10　Recognition results on ICDAR2015 dataset

3.3.5　模型参数量比较

表 3 为 ABCNet、MANGO、ABCNet v2、SPTS、SwinTextSpotter、TESTR 和 LWSPTS 7 种方法在参数量和权重文件大小 2 方面的比较结果, LWSPTS 的参数量为 10.59×10^6, 权重文件大小为 65.2 MB, 相比 ABCNet、ABCNet v2、SPTS、SwinTextSpotter 和 TESTR 参数量分别降低了 26.28×10^6、37.38×10^6、25.56×10^6、51.93×10^6、38.91×10^6, 主要是由于 ABCNet 和 ABCNet v2需要复杂后处理以确定文本区域边界, SwinTextSpotter 的骨干网络 Swin-Transformer 的参数量较高, SPTS、TESTR 依赖于 DETR 框架, 需要进行多重编码和解码过

程。LWSPTS 模型的参数量和权重文件大小分别为 SPTS 的 29.3% 和 47.1%, 说明了本研究模型在参数量方面具有一定优势。

表 3　模型参数量与权重文件大小比较

Table 3　Comparison of model parametric quantities and weighting files

模型	参数量/10^6	权重文件大小/MB
ABCNet	36.87	281.1
MANGO	—	159.5
ABCNet v2	47.97	365.9
SPTS	36.15	138.5
SwinTextSpotter	62.52	550.4
TESTR	49.50	556.8
LWSPTS	**10.59**	**65.2**

注: 黑体表示每列最优值。

3.4　消融实验

3.4.1　采用不同模块的结果对比

为了验证本文所提 TLA 模块、LEA 模块和标记选择模块的有效性, 在 Total-Text 数据集上进行消融实验。表 4 为采用不同模块在 Total-Text 数据集上的结果比较, 第 1 行是不加入 TLA 模块, 直接使用 PP-LCNet 提取的特征进行编码, 而且在编码器中不使用 LEA 模块, 不加入标记选择模块的结果; 第 2 行是只加入 TLA 模块, 不加入 LEA 模块和标记选择模块的结果; 第 3 行是在引入 TLA 模块的基础上使用 LEA 模块替换编码器中 FFN 的结果; 第 4 行是在引入 TLA 模块的基础上加入标记选择模块的结果; 第 5 行为同时引入 TLA 模块、LEA 模块和标记选择模块的结果。

表 4　在 Total-Text 数据集采用不同模块的结果比较

Table 4　Comparison of results using different modules in the Total-Text dataset　%

TLA模块	LEA模块	标记选择模块	None	Full
—	—	—	60.7	72.8
√	—	—	61.9	73.2
√	√	—	66.2	75.6
√	—	√	65.8	74.3
√	√	√	**69.5**	**79.1**

注: 黑体表示每列最优值。

TLA 模块的作用　通过表 4 第 1、2 行可以看出, 在引入 TLA 模块后, None 和 Full 指标分别提高了 1.2 百分点和 0.4 百分点, 主要是由于 TLA 可以通过 3 种并行的一维卷积充分学习特征通道之间的依赖关系, 使得文本识别精度有一定提升。

TLA 和 LEA 模块的作用　通过表 4 第 1、3 行可以看出, 同时引入 TLA 和 LEA 模块时, None 和 Full 分别提高了 5.5 百分点和 2.8 百分点, 而通过第 2、3 行可以看出, 引入 LEA 模块后, None 和 Full 分别提高了 4.3 百分点和 2.4 百分点, 因为 LEA 模块可以通过一组深度可分离卷积增强空间信息交互能力, 在一定程度上减少了小尺度文本漏检的情况。

TLA 和标记选择模块的作用　通过表 4 第 1、4 行可以看出, 同时引入 TLA 和标记选择模块时, None 提高了 5.1 百分点, Full 提高了 1.5 百分点, 通过第 2、4 行可以看出, 引入标记选择模块后, None 和 Full 分别提高了 3.9 百分点和 1.1 百分点, 说明标记选择模块可以根据序列中的重要性程度突出文本信息, 减少迭代过程中无关信息的累积, 缓解了文本信息编码不充分对识别精度造成的影响。

TLA、LEA 和标记选择模块的作用　通过表 4 第 1、5 行可以看出, 同时引入 TLA、LEA 和标记选择模块时, None 和 Full 提高了 8.8 百分点和 6.3 百分点, 证明了本研究所提各模块的有效性。

3.4.2　采用不同骨干网络对实验结果的影响

为了验证本研究采用以 PP-LCNet[14] 作为骨干网络方法的有效性, 将 PP-LCNet 与 ShuffleNetV2[34]、MobileNetV3[35]、GhostNet[36]、MixNet[37] 和 FBNetV2[38] 骨干网络对比, 表 5 为采用不同骨干网络在 Total-Text 数据集的结果。由表 5 可以看出, 以 PP-LCNet 为骨干网络的方法参数量为 10.59×10^6, None 和 Full 分别为 69.5% 和 79.1%。以 ShuffleNetV2 为骨干网络的方法参数量为 8.31×10^6, 相比于 PP-LCNet 降低了 2.28×10^6, 但是 None 和 Full 却降低了 3.0 百分点和 3.8 百分点。MobileNetV3、GhostNet 和 FBNetV2 这 3 种方法的参数量在 13×10^6 左右, 与 PP-LCNet 相差不大, 但是在识别精度方面与本研究方法有一定差距。在速度方面, 本研究方法的 20.2 f/s 位于第二, 低于 ShuffleNetV2 方法 2.1 f/s, 超出 FBNetV2 方法 0.6 f/s。

表 5　在 Total-Text 数据集采用不同骨干网络结果比较

Table 5　Comparison of results using different backbone networks in the Total-Text dataset

骨干网络	参数量/10^6	FPS/(f/s)	Non/%	Full/%
ShuffleNetV2	**8.31**	**22.3**	66.5	75.3
MobileNetV3	12.77	18.8	67.6	75.8
GhostNet	12.81	17.5	66.7	74.6
MixNet	19.49	16.8	68.3	77.2
FBNetV2	13.19	19.6	67.5	76.1
PP-LCNet	10.59	20.2	**69.5**	**79.1**

注: 黑体表示每列最优值。

3.4.3　采用并行解码方式的结果对比

为了验证本研究所使用的自回归解码方式的有效性，将自回归解码和并行解码 2 种方式对比，结果见表 6，其中在并行解码方式中，采用随机生成的序列作为目标序列。由表 6 可知，并行解码方式的 None 和 Full 分别为 44.1% 和 54.7%，与自回归解码方式相差 25.4 百分点和 24.4 百分点，主要是并行解码方式中随机生成目标序列的不确定性对最终的结果产生了较大影响。

表 6　采用不同解码方式在 Total-Text 数据集结果比较

Table 6　Comparison of results in the Total-Text dataset using different decoding methods　%

解码方式	None	Full
并行解码	44.1	54.7
自回归解码	69.5	79.1

2 种解码方式的可视化结果如图 11 所示，图 11(a) 为使用自回归解码方式的结果，其中各处文本实例定位清晰，识别准确。而图 11(b) 的结果中文本实例坐标混乱，比如“On” “the” “Beach”分布在文本实例的上下两侧，而且识别结果中存在重复，比如出现了 2 次“STOPPY”。说明了本研究采用自回归解码方式的有效性。

(a) 自回归解码方式的结果　(b) 并行解码方式的结果

图 11　自回归解码和并行解码方式对比

Fig. 11　Comparison of autoregressive and parallel decoding methods

3.4.4　设置不同超参数对结果的影响

本研究方法包含的超参数为编码器和解码器的层数，LEA 模块中隐藏层的维度 C_{m} 以及目标序列构建中的图片离散化后的尺度 n_{bins}。为了验证超参数对识别精度的影响，分别对以上 3 个超参数进行验证。

编码器和解码器的层数　本研究方法将解码器层数和编码器层数均设置为 3，为了验证不同编码器和解码器层数对结果的影响，进行如表 7 所示的实验，第 1 组为编码器和解码器层数相同时的结果，可以看到随着编码器和解码器的层数增加，模型的参数量也在增加，文本识别精度也有了一定提高。从第 1 组实验可以看出，当编码器和解码器层数为 3 与层数为 5 时，模型的参数量增加了 45.0 百分点，但是 None 和 Full 只增加了 1.3 百分点和 1.2 百分点。将编码器与解码器层数设置为 4 时，参数量增加了 22.6 百分点，None 和 Full 增加了 0.8 百分点和 0.4 百分点，识别精度提升与模型参数增加程度相差较大。而当编码器和解码器层数设置为 2 时，参数量虽然降低了 22.4 百分点，但是 None 和 Full分别降低了 3.3 百分点和 2.6 百分点，使得文本识别精度出现了较大的下降，也不适用于文本方法。因此通过第 1 组实验，将编码器和解码器层数设定为 3。第 2 组实验是将编码器层数设置为 3 的情况下，验证解码器层数在 2、3、4、5 情况下的文本识别精度，当编码器层数为 2 时，参数量为 9.54×10^6，None 和 Full 分别为 67.9% 和 77.0%，与编码器为 3 时相差 1.6 百分点和 2.1 百分点，而当编码器层数为 4 时，只有在 None 指标上提高了 0.2 百分点。经过比较，本研究将解码器的层数设置为 3。由此证明本研究所提方法的有效性。

表 7　采用不同层数编码器和解码器的结果比较

Table 7　Comparison of results using different layers of encoders and decoders

组数	编码器	解码器	参数量/10^6	None/%	Full/%
1	2	2	8.21	66.2	76.5
	3	3	10.59	69.5	79.1
	4	4	12.98	70.3	79.5
	5	5	15.36	70.8	80.3
2	3	2	9.54	67.9	77.0
	3	3	10.59	69.5	79.1
	3	4	11.65	69.7	79.1
	3	5	12.70	69.8	79.6

LEA 模块中隐藏层的维度 C_{m}　为了验证不同隐藏层维度对结果的影响，本研究分别将隐藏层维度设置为 512、1 024 和 2 048，在 Total-Text 数据集的结果见表 8。当隐藏层维数为 512 时，None 和 Full 分别为 67.7% 和 77.8%。将隐藏层维度调整为 1 024 时，模型参数量增加了 2.39×10^6，None 和 Full 分别提高了 1.8 百分点和 1.3 百分点，原因在于增加隐藏层维度后，模型的非线性表征能力

增强, 使得识别精度有了一定程度提高。但是当将隐藏层维度设置为 2 048 后, 相比 1 024, 参数量增加了 4.77×10^6, Full 增加了 0.5 百分点, None 下降了 0.2 百分点, 主要原因在于隐藏层维度过高导致文本之间的关联性变弱, 识别精度提升有限。由此证明本研究将 LEA 模块中的隐藏层维度设置为 1 024 是有效的。

表 8　对于 LEA 模块设置不同隐藏层维度的结果比较
Table 8　Comparison of results for LEA modules with different hidden layer dimensions

隐藏层维度	参数量/10^6	None/%	Full/%
512	8.20	67.7	77.8
1 024	10.59	69.5	79.1
2 048	15.36	69.3	79.6

目标序列中的 n_{bins}　为了验证图像训练过程中的离散化程度对识别结果的影响, 本研究将 n_{bins} 分别设置为 600、800 和 1 000, 并在 Total-Text 数据集上进行实验, 结果见表 9。当 n_{bins}=1 000 时, 模型参数量为 10.59×10^6, None 和 Full 分别为 69.5% 和 79.1%。相比 n_{bins}=800 时, 参数量降低了 0.1×10^6, None 和 Full 分别提升了 0.6 百分点和 1.3 百分点。相比 n_{bins}=600 时, None 和 Full 分别提升了 0.6 百分点和 1.3 百分点。主要原因在于随着 n_{bins} 的增大, 转换后的图像分辨率增加, 有利于细化原始文本图像中的小尺度文本信息, 突出局部文本特征。通过比较, 本研究将目标序列中图像的离散化程度 n_{bins} 设置为 1 000。

表 9　训练过程中设置不同离散程度的结果比较
Table 9　Comparison of the results of setting different levels of dispersion during training

n_{bins}	参数量/10^6	None/%	Full/%
600	10.38	67.6	76.9
800	10.49	68.9	77.8
1 000	10.59	69.5	79.1

4　结束语

针对 SPTS 模型推理速度慢、小尺度文本检测精度低的情况, 本研究提出一种轻量级网络 LWSPTS。首先, 设计了 TLA 模块对骨干网络的特征图处理, 通过 3 种并行卷积加强了通道间信息交互; 随后, 在原有编码器的基础上提出用 LEA 模块替换原有的前馈网络, 减少了小尺度文本的漏检情况; 然后, 设计了标记选择模块抑制了迭代过程中的无关因素累积, 增加了字符识别的准确性, 缓解了编码器层数减少导致的编码不充分问题; 最后, 通过自回归 Transformer 解码器预测出文本识别结果。

在未来的工作中, 将尝试在目标序列中添加随机噪声, 以提高模型的鲁棒性, 由于目标序列的各文本实例之间是随机组合、互不影响的, 也将会对解码结构进行改进, 实现图像中的不同文本实例的同时预测, 进一步提高模型推理速度。

参考文献:

[1] 刘崇宇, 陈晓雪, 罗灿杰, 等. 自然场景文本检测与识别的深度学习方法[J]. 2021, 26(6): 1330−1367.
LIU Congyu, CHEN Xiaoxue, LUO Canjie, et al. Deep learning methods for scene text detection and recognition [J]. Journal of image and graphics, 2021, 26(6): 1330−1367.

[2] FENG Wei, HE Wenhao, YIN Fei, et al. TextDragon: an end-to-end framework for arbitrary shaped text spotting [C]//2019 IEEE/CVF International Conference on Computer Vision. Seoul: IEEE, 2019: 9075−9084.

[3] QIAO Liang, TANG Sanli, CHENG Zhanzhan, et al. Text perceptron: towards end-to-end arbitrary-shaped text spotting[C]//Proceedings of the AAAI conference on artificial intelligence. New York: AAAI, 2020: 11899−11907.

[4] LIU Yuliang, CHEN Hao, SHEN Chunhua, et al. ABCNet: real-time scene text spotting with adaptive bezier-curve network[C]//2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle: IEEE, 2020: 9806−9815.

[5] LIU Yuliang, SHEN Chunhua, JIN Lianwen, et al. ABCNet v2: adaptive bezier-curve network for real-time end-to-end text spotting[J]. IEEE transactions on pattern analysis and machine intelligence, 2022, 44(11): 8048−8064.

[6] HUANG Mingxin, LIU Yuliang, PENG Zhenghao, et al. SwinTextSpotter: scene text spotting via better synergy between text detection and text recognition[C]//2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition. New Orleans: IEEE, 2022: 4583−4593.

[7] WU Jingjing, LYU Pengyuan, LU Guangming, et al. Decoupling recognition from detection: single shot self-reliant scene text spotter[C]//Proceedings of the 30th ACM International Conference on Multimedia. Lisboa : ACM, 2022: 1319−1328.

[8] PENG Dezhi, WANG Xinyu, LIU Yuliang, et al. SPTS: single-point text spotting[C]//Proceedings of the 30th ACM International Conference on Multimedia. Lisboa:

ACM, 2022: 4272−4281.

[9] ZHANG Xiang, SU Yongwen, TRIPATHI S, et al. Text spotting transformers[C]//2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition. New Orleans: IEEE, 2022: 9509−9518.

[10] KITTENPLON Y, LAVI I, FOGEL S, et al. Towards weakly-supervised text spotting using a multi-task transformer[C]//2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition. New Orleans: IEEE, 2022: 4594−4603.

[11] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[EB/OL]. (2017−06−12)[2021−01−01]. http://arxiv.org/abs/1706.03762.

[12] LIU Ze, LIN Yutong, CAO Yue, et al. Swin Transformer: hierarchical Vision Transformer using Shifted Windows [C]//2021 IEEE/CVF International Conference on Computer Vision. Montreal: IEEE, 2021: 9992−10002.

[13] CARION N, MASSA F, SYNNAEVE G, et al. End-to-end object detection with transformers[M]//Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020: 213−229.

[14] CUI Cheng, GAO Tingquan, WEI Shengyu, et al. PP-LCNet: a lightweight CPU convolutional neural network [EB/OL]. (2021−09−17)[2021−01−01]. http://arxiv.org/abs/2109.15099.

[15] LIU Xuebo, LIANG Ding, YAN Shi, et al. FOTS: fast oriented text spotting with a unified network[C]//2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City: IEEE, 2018: 5676−5685.

[16] ZU Xinyan, YU Haiyang, LI Bin, et al. Towards accurate video text spotting with text-wise semantic reasoning [C]//Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence. Macau: International Joint Conferences on Artificial Intelligence Organization, 2023: 1858−1866.

[17] LYU Pengyuan, LIAO Minghui, YAO Cong, et al. Mask TextSpotter: an end-to-end trainable neural network for spotting text with arbitrary shapes[C]//European Conference on Computer Vision. Cham: Springer, 2018: 71−88.

[18] HE Kaiming, GKIOXARI G, DOLLÁR P, et al. Mask R-CNN[C]//2017 IEEE International Conference on Computer Vision. Venice: IEEE, 2017: 2980−2988.

[19] GARCIA-BORDILS S, KARATZAS D, RUSIÑOL M. STEP - towards structured scene-text spotting[C]//2024 IEEE/CVF Winter Conference on Applications of Computer Vision. Waikoloa: IEEE, 2024: 872−881.

[20] LIAO Minghui, PANG Guan, HUANG Jing, et al. Mask TextSpotter v3: segmentation proposal network for robust scene text spotting[C]//VEDALDI A, BISCHOF H, BROX T, et al. European Conference on Computer Vision. Cham: Springer, 2020: 706−722.

[21] WANG Wenhai, XIE Enze, LI Xiang, et al. PAN++: towards efficient and accurate end-to-end spotting of arbitrarily-shaped text[J]. IEEE transactions on pattern analysis and machine intelligence, 2022, 44(9): 5349−5367.

[22] RONEN R, TSIPER S, ANSCHEL O, et al. GLASS: global to local attention for scene-text spotting[C]//AVIDAN S, BROSTOW G, CISSÉ M, et al. European Conference on Computer Vision. Cham: Springer, 2022: 249−266.

[23] LIU Wei, CHEN Chaofeng, WONG K Y. Char-net: a character-aware neural network for distorted scene text recognition[C]//Proceedings of the AAAI conference on artificial intelligence New Orleans: AAAI, 2018.

[24] WANG Pengfei, ZHANG Chengquan, QI Fei, et al. PGNet: real-time arbitrarily-shaped text spotting with point gathering network[J]. Proceedings of the AAAI conference on artificial intelligence, 2021, 35(4): 2782−2790.

[25] QIAO Liang, CHEN Ying, CHENG Zhanzhan, et al. MANGO: a mask attention guided one-stage scene text spotter[J]. Proceedings of the AAAI conference on artificial intelligence, 2021, 35(3): 2467−2476.

[26] YE Maoyuan, ZHANG Jing, ZHAO Shanshan, et al. DeepSolo: let transformer decoder with explicit points solo for text spotting[EB/OL]. (2022−11−19)[2022−12−01]. http://arxiv.org/abs/2211.10772.

[27] ZHU Xizhou, SU Weijie, LU Lewei, et al. Deformable DETR: deformable transformers for end-to-end object detection[EB/OL]. (2010−10−18)[2021−01−01]. http://arxiv.org/abs/2010.04159.

[28] CHEN Ting, SAXENA S, LI Lala, et al. Pix2seq: a language modeling framework for object detection[EB/OL]. (2021−09−22)[2021−12−01]. http://arxiv.org/abs/2109.10852.

[29] GUPTA A, VEDALDI A, ZISSERMAN A. Synthetic data for text localisation in natural images[C]//2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas: IEEE, 2016: 2315−2324.

[30] KARATZAS D, GOMEZ-BIGORDA L, NICOLAOU A, et al. ICDAR 2015 competition on Robust Reading[C]//2015 13th International Conference on Document Analys-

is and Recognition. Tunis: IEEE, 2015: 1156−1160.

[31] LIU Yuliang, JIN Lianwen, ZHANG Shuaitao, et al. Detecting curve text in the wild: new dataset and new solution[EB/OL]. (2017−12−06)[2021−01−01]. http://arxiv.org/abs/1712.02170.

[32] CHNG C K, CHAN C S. Total-text: a comprehensive dataset for scene text detection and recognition[C]//2017 14th IAPR International Conference on Document Analysis and Recognition. Kyoto: IEEE, 2017: 935−942.

[33] NAYEF N, YIN Fei, BIZID I, et al. ICDAR2017 robust reading challenge on multi-lingual scene text detection and script identification - RRC-MLT[C]//2017 14th IAPR International Conference on Document Analysis and Recognition. Kyoto: IEEE, 2017: 1454−1459.

[34] MA Ningning, ZHANG Xiangyu, ZHENG Haitao, et al. ShuffleNet V2: practical guidelines for efficient CNN architecture design[M]//Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018: 122−138.

[35] HOWARD A, SANDLER M, CHEN Bo, et al. Searching for MobileNetV3[C]//2019 IEEE/CVF International Conference on Computer Vision. Seoul: IEEE, 2019: 1314−1324.

[36] HAN Kai, WANG Yunhe, TIAN Qi, et al. GhostNet: more features from cheap operations[C]//2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle: IEEE, 2020: 1577−1586.

[37] TAN Mingxing, LE Q V. MixConv: mixed depthwise convolutional kernels[EB/OL]. (2019−07−22)[2021−01−01]. http://arxiv.org/abs/1907.09595.

[38] WAN A, DAI Xiaoliang, ZHANG Peizhao, et al. FBNetV2: differentiable neural architecture search for spatial and channel dimensions[C]//2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle: IEEE, 2020: 12962−12971.

作者简介：

曹锦纲，讲师，博士，主要研究方向为图像处理和模式识别，发表学术论文 10 余篇。E-mail：caojg168@126.com。

张泽恩，硕士研究生，主要研究方向为深度学习和文本检测。E-mail：zze15832206526@126.com。

张铭泉，副教授，博士，主要研究方向为计算机组成、机器学习、模式识别，发表学术论文 20 余篇。E-mail：mqzhang@ncepu.edu.cn。