Building A Mobile Text Recognizer via Truncated SVD-based Knowledge Distillation-Guided NAS

Weifeng Lin[1]
eelinweifeng@mail.scut.edu.cn
Canyu Xie[1]
canyuxie@qq.com
Dezhi Peng[1]
pengdzscut@foxmail.com
Jiapeng Wang[1]
scutjpwang@foxmail.com
Lianwen Jin[1]
eelwjin@scut.edu.cn
Wei Ding[2]
dingwei.dw@alibaba-inc.com
Cong Yao[2]
yaocong2010@gmail.com
Mengchao He[2]
mengchao.hmc@alibaba-inc.com

[1] South China University of Technology
Guangzhou, China
[2] Alibaba Group
Hangzhou, China

Abstract

Text recognition poses significant challenges in computer vision, with unresolved issues such as the trade-off between recognition accuracy, storage, and computation complexity in real-world applications. To tackle this challenge, we propose a mobile text recognizer that integrates Truncated Singular Value Decomposition (TSVD)-based Knowledge Distillation (KD) into the Neural Architecture Search (NAS) process. We also improved the search space of NAS by introducing a novel Mobile Char Block (MCB) and a channel-aware search. We conducted a series of experiments that demonstrated the efficacy of our search strategy in identifying a lightweight model that achieved comparable accuracy to existing methods but with significantly lower computation costs and smaller storage space. We evaluated our model on four benchmark datasets, including IAM, ICDAR2013, and SCUT-HCCDoc, for handwriting recognition, and on JS-Printed, a large-scale in-house bilingual dataset of printed documents. Our student model even outperformed the teacher model with 10.5× faster and 8.2× smaller on x86 and ARM devices on the widely used IAM dataset.

1 Introduction

Text plays a crucial role in the acquisition and preservation of information for humans. The widespread availability of text-based information has enabled various intelligent applications, highlighting the importance of digital text processing. Several popular mobile applications such as bill-card digitization, scanning translation, and street view positioning rely on text recognition technology, which requires deep learning-based systems to be deployed on terminal devices with limited computing resources. However, current approaches to improving text recognition accuracy have led to complex models with increased computation cost and number of parameters. This makes them unsuitable for deployment in settings with limited resources.

Previous research efforts have attempted to tackle this issue, such as those by [7, 21, 43, 50]. Although these methods have somewhat reduced the complexity and redundancy of text recognition, significant issues remain. For example, in [21], the Hamming classification mechanism determines that its computation is not less than that of the general fully connected (FC) layer. Meanwhile, in [7, 9, 43], the recognition models contain Long Short Term Memory (LSTM) [13] which reduces parallelism of the overall system. In [36], Shi et al. propose convolutional recurrent neural network (CRNN) by taking advantage of Long short term memory (LSTM) [13] and convolutional neural networks (CNN) for the image-based sequence recognition, which becomes one of the most popular training frameworks for text recognition task. In CRNN, given an image to be recognized, a CNN extracts the features of the image, LSTM performs the sequence modeling, and finally, decodes the sequence with Connectionist Temporal Classifier (CTC) [11]. AutoSTR [50] adopts NAS to address the limitations of manually designed networks, but its search space is limited, preventing it from searching for operations and downsample paths efficiently.

In this paper, we present a novel method for searching a mobile CTC-based text recognizer. Our approach involves utilizing the NAS method by redesigning the mobile search space, as well as incorporating the supervision of a powerful teacher in the NAS process. To achieve this, we introduce a TSVD-based knowledge distillation method, inspired by [22]. Notably, we use only Convolutional Neural Networks (CNN) and fully connected (FC) layers in our model, unlike previous approaches that include Long Short-Term Memory (LSTM), which is not parallelism-friendly.

The training process can be divided into two parts: the NAS part and the KD part. The objective of the NAS part is to search for an appropriate model for the text recognition task. Inspiring from ProxylessNAS [5], we make three improvements: (1) a new search space, designed carefully with a **M**obile **C**har convolutional **B**lock (MCB), (2) the inclusion of channel-aware dimension search, and (3) replacement of the Batch Normalization (BN) layer [17] with a Layer Normalization (LN) layer [2], considering the characteristics of the text recognition task. The objective of the KD part is to reduce the gap in accuracy between the teacher and student models. To achieve this, a regressor is required to match the shape of teacher and student feature maps for feature-based KD. Our proposed approach involves reusing the teacher's classifier weight via Truncated Singular Value Decomposition (TSVD) to obtain the regressor's weight. This method is more effective for dimensional reduction of the teacher's feature.

Experiments are conducted on four benchmarks, namely ICDAR2013 [46], IAM [28], SCUT-HCCDoc [49] and an in-house dataset JS-Printed. The results show that our search method can search for a better student for TSVD-based KD-Guided training, resulting in a significant performance improvement. On the widely used IAM dataset, the proposed model achieves comparable performance with other start-of-the-art methods with an accurate rate (AR) of 92.60%, while the inference time for one image is only 18ms, 24ms and 2.0ms on x86 devices, ARM devices and 1080ti GPUs respectively with 8.7MB storage.

The contributions of this paper are summarized as follows

1. We introduce three new techniques to the NAS process, including Mobile Char Block (MCB), channel-aware dimension search and layer normalization replacement to search for a better lightweight model.

2. We propose to reuse the weight of the teacher's fully connecting layers, and leverage TSVD to obtain the weight of the regressor for feature-based knowledge distillation, resulting in better feature matching between the teacher and student models

3. Extensive experiments on several mainstream text recognizers and lightweight models show the effectiveness of our method. The results demonstrated excellent performance on ICDAR2013, IAM, SCUT-HCCDoc and JS-Printed datasets.

2 Related Work

2.1 Lightweight Text Recognition

In the field of Optical Character Recognition (OCR), not only the recognition accuracy but also the inference efficiency and storage of the model require carefully considering in real scenarios. The end-to-end text recognition methods can be roughly divided into two main categories, CTC-based [36] and attention-based [37, 42]. Attention-based methods need to decode the recognition results serially, which is not friendly for fast inference [44]. CTC-based methods are often used in the task of lightweight text recognition as they can decode in parallel for fast decoding. In [7], the researchers adopt Tucker decomposition and knowledge distillation methods to design a lightweight model for the English recognition tasks. In [21], HammingOCR designs a lightweight hamming classifier to solve the problem of excessive classifier storage. In [43], Xie at el. first use tucker and SVD decomposition methods to accelerate a CNN-ResLSTM model, and then adopt unstructured network pruning and quantization to reduce the network's parameters. The aforementioned methods accelerate and compress the recognizer to a certain extent, but they still cannot be used in scenarios where resources are extremely scarce.

2.2 Neural Architecture Search

NAS (Neural Architecture Search) aims to design a specific model for a given task [33]. Traditional NAS algorithms treat architecture search tasks as a meta-learning process [51, 52]. A meta-controller is introduced to search for the optimal network architecture by training candidate networks in a loop, guiding the search process, and updating the controller during searching. However, such methods are time-consuming, particularly for large-scale tasks. Instead of training all candidate models for evaluation, one-shot NAS methods [3, 4, 5] build an over-parameterized network that includes all candidate paths and search for a sub-network from it. Since all candidate architectures share weights in one over-parameterized network, one-shot NAS methods require less time for model evaluation. For example, DARTS [25] constructs an over-parameterized network with both architecture and weight parameters, then trains the parameters in a loop and selects the sub-net according to the architecture parameters. In contrast, ProxylessNAS [5] trains the over-parameterized network by sampling one sub-net on each training iteration, reducing the memory cost in the search phase.

2.3 Knowledge Distillation

Knowledge Distillation (KD) is a process in which a compact model is taught using a more powerful teacher model and the teacher's knowledge can be divided into three categories: response-based knowledge, feature-based knowledge, and relation-based knowledge [10]. Response-based knowledge refers to the neural response of the last output layer, and the most popular type of response-based knowledge is soft target [1, 12], which introduces a temperature parameter in the softmax function. Feature-based knowledge [19, 30, 34, 47] uses the feature representation in the middle of the network as the teacher's knowledge. Relation-based knowledge [29, 40, 45] allows the student to explore the relationships between different layers or data samples instead of directly matching the output of models. Although knowledge distillation can be applied to any pair of teacher and student networks, the distillation performance may vary for different students [26].

3 Methodology

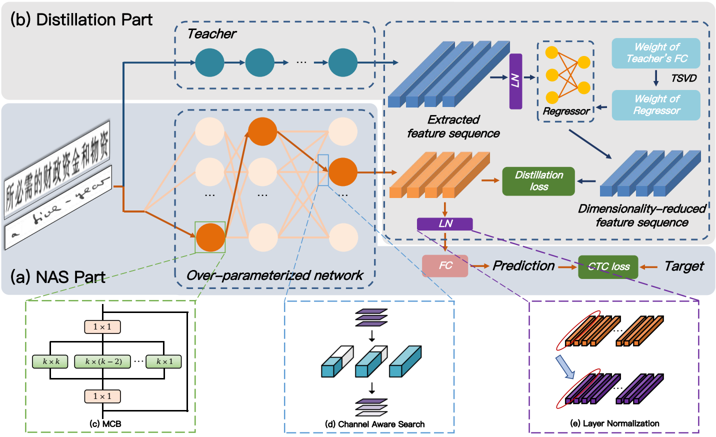

Figure 1: **Overall architecture of the proposed NAS method.** (a) An over-parameterized network with learnable blocks and channel configurations for neural architecture search. (b) The TSVD-based knowledge distillation is introduced to the NAS process to guide the student searching. (c) Mobile Char Block (MCB). (d) Channel-Aware Search. (e) Layer Normalization.

For the whole pipeline for text recognition, we firstly search for a lightweight student model from the over-parameterized network and subsequently train the search model with the proposed distillation method to get an enhanced performance.

3.1 Neural Architecture Search

One of the issues in NAS is the search efficiency [5, 32]. In this paper, we perform a network architecture search based on ProxylessNAS [5] owing to its lower resource cost. However, the performance of the search model is highly dependent on the search space. To address this issue, we first design a new block named Mobile Char Block (MCB) for the text line recognition tasks and then add channel dimension search to ProxylessNAS to expand the search space. Finally, we replace the BN layer with the LN layer in front of the FC classifier to get a better normalization for the extracted feature sequence.

3.1.1 ProxylessNAS

We briefly illustrate the principle of ProxylessNAS here. Given an over-parameterized network $\mathcal{N}$ with N layers, there are m_l candidate primitive operations $O_l = \{o_i^l\}$ at the l-th layer, and each candidate operation o_i^l is associated with a hyperparameter α_i^l, which relaxes the categorical choice of a particular operation to a softmax over all possible operations. The search problem can be formulated as:

$$\begin{aligned} &\min_{\alpha} L_{val}(\mathcal{N}(w^*, \alpha)) \\ \mathrm{s.t.}\, w^* = &\arg\min_{w} L_{train}(\mathcal{N}(w, \alpha)), \end{aligned} \tag{1}$$

where α is the set of α_i^l, and w is the weight of the over-parameterized network $\mathcal{N}$. L_{train} and L_{val} are the loss functions of training and validation datasets, respectively.

Eq.1 indicates that the weights and architecture parameters are trained with the training and validation datasets, respectively. When training the weight parameters, the architectural parameters are fixed, and vice versa. The training of the weights and architecture parameters is alternated.After completing the training of the over-parameterized network, the operations with the highest probability at each layer are selected to perform over the over-parameterized network, generating the best sub-architecture for the corresponding layer, which is finally formed and combined into a sub-network. ProxylessNAS adopts binary path learning and binarized parameter training to obtain the optimal w^* and α^* values, which can decrease the memory storage.

3.1.2 Mobile Char Block

In the CTC-based recognition model, we use a convolutional neural network (CNN) backbone to extract a sequence of features from an input image and predict the output sequence frame by frame. Although the receptive field of each frame could cover more than one character in the input image (as shown in Fig. 2) which can capture contextual information, the detailed features of individual characters are lost.

Inspired by this observation, we propose a Mobile Char Block (MCB, shown in Fig. 3) to enhance the network's ability to extract the feature of central characters. We start from the inverted residual block (Fig. 3(a)) and replace the single-branch depth-wise convolutional layer with multi-branch one. As shown in Fig. 3(b), for each branch, the width of the kernel is reduced by 2 compared to the previous branch until the width reaches 1, while the height remains the same. Keeping the receptive field of the original regular convolutional kernel constant, the receptive fields of the other convolution branches gradually decrease in width (see Fig. 2). Therefore, other branches are more inclined to perform feature extraction on a small range in width. When using multi-branch MCB, the ability to extract narrow-range character features can be enhanced. In the inference phase (Fig. 3(c)), we further use the structure re-parameterization technique [8] to merge the multi-branch into a single-branch owing to its linearity, which can reduce storage and computation while maintaining the effect of the multi-branch structure.

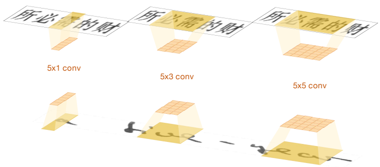

Figure 2: The receptive field of each branch in MCB (e.g. k=5).

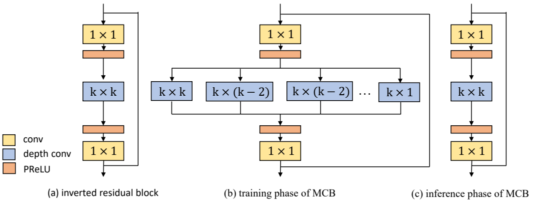

Figure 3: Structure of inverted residual block and MCB architecture in training and inference phase.

3.1.3 Channel-Aware Search

Although ProxylessNAS is powerful, its search space is limited only to the operations of the network, which may not be sufficient for some tasks. In addition to operations, the number of channels in the convolutional layer is also an essential configuration parameter that affects the network's performance. Therefore, we adopt a channel-masking mechanism proposed in [41] to enable the NAS algorithm to search with different channel configurations.

When we apply masks on the arbitrary channels of the output feature map, it is equivalent to prune the corresponding output filters in the convolutional layer. Therefore, instead of setting a great of search blocks in the search space, we only require setting the maximum channel of each layer and the masks of different ratios to achieve different output channels of the convolutional layer. Thus the channel-aware search process can be formulated as:

$$
\begin{aligned}
\mathcal{O}_{masked-i} &= \beta_1 M_1(\mathcal{O}_i) + \beta_2 M_2(\mathcal{O}_i) + \ldots + \beta_n M_n(\mathcal{O}_i) \\
&= \sum_{j}^{n} \beta_j M_j(\mathcal{O}_i),
\end{aligned}
\tag{2}
$$

where M_j is the mask function with a preset ratio r_j, β_j is a hyperparameter that represents the importance of the j-th mask, and $\sum_j^n \beta_j = 1$. $\mathcal{O}_i$ is the output of the i-th layer in the over-parameterized network $\mathcal{N}$. $\mathcal{O}_{masked-i}$ is the weighted sum of all masked feature maps. In the search phase, β_j and α are optimized together. When the search process is finished, the channel configuration can be obtained according to the mask with the maximum β.

3.1.4 Layer Normalization

More than half of the CTC-based models in the feature sequence belong to the blank category, which might influence the statistics of the mean and variance learned by BN[17].Therefore, we propose to use the LN layer[2] to normalize the feature sequence. The mean μ and variance σ of the LN are calculated as follows:

$$
\mu^j = \frac{1}{C}\sum_{i=1}^{C} a_i^j, \quad \sigma^j = \frac{1}{C}\sum_{i=1}^{C}(a_i^j - \mu^j)^2, \tag{3}
$$

where C is the channel number of each frame, a is the element in each frame, and j denotes the index of the frame in the feature sequence. As shown in Eq.3, the means and variances are calculated independently between different frames, thus avoiding the drawbacks of BN.

3.2 TSVD-based Knowledge Distillation

Knowledge distillation is a powerful technique that can narrow the performance gap between complex and lightweight models. In this paper, we employ feature-based distillation methods. The dimensions of the features extracted from the teacher and student models are different, which makes direct feature matching challenging. Inspired by FitNets [34], we utilize a regressor to align the features with different channels.

With the regressor, the feature-based knowledge distillation process can be formulated as:

$$L_{KD} = \frac{1}{2}||reg(\mathcal{F}^{T}_{CNN}, W_{reg}) - \mathcal{F}^{S}_{CNN}||^2, \tag{4}$$

where $\mathcal{F}^{T}_{CNN}$ and $\mathcal{F}^{S}_{CNN}$ are the feature extracted by the teacher and student's CNN backbone respectively. *reg* is the regressor function. W_{reg} is the parameter of the regressor.

However, the parameters of regressor in FitNets are randomly given, they may not be good at dimension matching. Instead, we find that the parameter of the regressor can be provided from the teacher's classification layer with singular value decomposition [38] technology. We propose to apply Truncated Singular Value Decomposition (TSVD) to the teacher's classification layers as follows:

$$W^{T}_{cls} = U_{m\times m}\Sigma_{m\times c^T}V^{*}_{c^T\times c^T} \approx U_{m\times c^S}\Sigma_{c^S\times c^S}V^{*}_{c^S\times c^T}, \tag{5}$$

where W^{T}_{cls} is the parameters of the teacher's classification layer. c^T and c^S is the channel dimension of teacher and student's feature respectively. $(\cdot)^*$ denotes the transpose operation. With the truncated value c^S, we can obtain the matrix $V^{*}_{c^S\times c^T}$ and treat it as the weight of the regressor. It is notable that the number of classes m must not be lower than the number of channels in the student network to enable the TSVD. However, this limitation is not a significant concern in our case of lightweight text recognition, as textual characters typically involves a sufficient number of classes and student networks typically have a smaller number of feature channels.

After using the regressor for the dimensionality reduction, the dimensionality-reduced features can be multiplied by $U_{m\times c^S}\Sigma_{c^S\times c^S}$ to obtain the approximate classification result of the teacher model, which indicates that the dimensionality-reduced features still contain valid information in the teacher's knowledge.Therefore, we propose the TSVD-based knowledge distillation loss such that:

$$L_{TSVD-KD} = \frac{1}{2}||\mathcal{F}^{T}_{CNN}V^{*}_{c^S\times c^T} - \mathcal{F}^{S}_{CNN}||^2, \tag{6}$$

Finally, the loss functions for both network searching and training are given by:

$$L = \alpha L_{CTC} + \beta L_{TSVD-KD} \tag{7}$$

where L_{CTC} and $L_{TSVD-KD}$ denote the CTC loss and the TSVD-based knowledge-distillation loss respectively. α and β are the weight factors of L_{CTC} and L_{TSVD} respectively. In our implementation, we empirically set $\alpha = 1.0$ and $\beta = 1.0$.

4 Experiments

4.1 Datasets

IAM The IAM handwriting database is based on handwritten English text copied from the LOB corpus. It contains 747 documents (6,482 lines) in the training set, 116 documents (976 lines) in the validation set and 336 documents (2,915 lines) in the test set.

CASIA-HWDB CASIA-HWDB dataset [24] is a large-vocabulary Chinese handwriting dataset. It contains 6 subsets, in which CASIA-HWDB1.0-1.2 contain 3,118,477 isolated characters and CASIA-HWDB2.0-2.2 contain 41,781 unconstrained handwritten texts. CASIA-HWDB dataset contains 7,356 character classes.

ICDAR-2013 ICDAR-2013 dataset is from ICDAR-2013 Chinese handwriting recognition competition [46] task 4. It contains 3,432 text lines. It is used as the test dataset on the experiments of CASIA-HWDB and ICDAR-2013.

SCUT-HCCDoc SCUT-HCCDoc [49] is a dataset of handwritten Chinese text in unconstrained camera-captured documents, which contains 93,411 text images for training and 23,218 images for testing. It contains 6,109 categories of characters.

JS-Printed JS-Printed is an in-house dataset of scanned bilingual printed text in English and Chinese, including real and synthetic data. The real data contains 590,000 training text images and 10,000 testing images, and the synthetic data is synthesized using TTF font files, which contains 1,000,000 images. It contains 27,767 categories of characters in total.

4.2 Implementation Details

The proposed method can be divided into search, training, and inference phases. In the search and training phases, we build a CTC recognizer with ResNet24 backbone and replace the normalization layer of feature sequence with LN layer. We treat it as the teacher model named ResNet24LN. The training process consists of two stages. In the first stage, a pre-trained teacher network assists the NAS method in searching for a student network. In the second stage, the searched student network is trained from scratch using label supervision and distillation methods. Our search space includes a 26-layer hyperparameter network. The initial layer process the image input through a 3×3 convolutional layer. The remaining 25 layers are divided into five stages. The first convolution in each stage downsamples the feature maps. Each layer offers seven distinct operations, including an identity mapping and six variations of the MCB module. We also incorporate multiple channel ratio options for each stage within the search space. All the experiments are conducted with PyTorch. In the inference phase, the model is deployed using MNN [18] for inference time measurement. We adopt ADADELTA [48] with a learning rate of 1.0 to optimize the objective function. The image size is set to 64×1024 for IAM, ICDAR2013 and SCUT-HCCDoc benchmarks and 32×1024 for JS-Printed benchmark. In the search phase, we first warm up the over-parameterized network by selecting and searching the sub-network uniformly for one epoch. Then we train the architecture and weight parameters for 60 epochs jointly. In the training phase, the learning rate is set to 1.0 and reduced to 0.1 in the 20-th epoch. The total training epoch is set to 30 for all benchmarks. All models are searched and trained on 2 NVIDIA 1080TI GPUs, and deployed on Snap-dragon 888 CPU (ARM device) and Intel i7-4790 CPU (x86 device).

4.3 Evaluation metrics

Following the previous study on text recognition, AR is adopted to evaluate the performance of the recognition system. AR is defined as:

$$AR = 1 - (D_e + S_e + I_e)/N_t, \tag{8}$$

where D_e, S_e, and I_e denote the total number of deletion, selection, and insertion errors. N_t denotes the number of label sequence length.

4.4 Comparison with existing Text-line recognition methods

As shown in Table 1, our search model achieves higher accuracy than some existing methods and 46× smaller than [20] on the IAM datasets. The search student model even outperformed the teacher with 30× less computational complexity. Although the AR of our search model is not the highest, we achieve the best performance in storage and inference speed. On the ICDAR2013 dataset, our model is 10.5× faster than the compact model in [43] on x86 devices, while achieving 1.36% better performance. Notably, the compact model [43] utilizes a series of compression methods such as low rank decomposition, unstructured network pruning, and quantization methods to compress the full model. These compression techniques can also be applied to our model to achieve further compression. Compared to [23] and [31], although there is still a performance gap, our method requires much less storage space than both of them (by 25× and 15×, respectively). For the SCUT-HCCDoc dataset, the search model's speed is only 11ms, 15ms, and 1.9ms on x86, mobile devices, and GPU, respectively, and the storage is only 5.74MB. Compared with [31], with limited accuracy gap (2.61%), our model requires 20× less storage space and achieves 15× faster on x86 devices. In general, the inference speed and storage of our search model can already achieve the purpose of application in not only real mobile scenarios but also GPUs, while keeping comparable recognition accuracy on all benchmarks.

Dataset	Method	AR	Storage(MB)	FLOPs(G)	Speed per line(ms)
IAM	Ingle et al. [16]	85.90%	42.4	-	-
	Chaudhary and Bali [6]	90.20%	112	-	-
	Kang et al. [20]	92.38%	400	-	-
	DAN [42]	**93.60%**	-	-	-
	ResNet24LN (Teacher)	92.00%	71.3	62.62	189(x86) / 246(ARM) / 12.8(GPU)
	Ours (Search)	92.60%	**8.7**	**1.92**	**18(x86) / 24(ARM) / 2.0(GPU)**
ICDAR2013	Xie et al. [43](Full model)	91.55%	61	16.57	318 (x86)
	Xie et al. [43](Compact model)	90.50%	**2.8**	4.46	146 (x86)
	Huang et al. [15]	91.82%	45.6	-	-
	Liu et al. [23]	93.62%	203	-	-
	Peng et al. [31]	**94.50%**	119	-	164 (x86)
	ResNet24LN (Teacher)	92.91%	80.10	63.42	141(x86) / 272(ARM) / 12.5(GPU)
	Ours (Search)	91.86%	8.34	**1.26**	**14(x86) / 18(ARM) / 1.9(GPU)**
SCUT-HCCDoc	Zhang et al. [49]	87.46%	59	16.40	312 (x86)
	Peng et al. [31]	**90.71%**	116.5	-	152 (x86)
	Liu et al. [23]	89.06%	200.5	-	-
	ResNet24LN (Teacher)	89.09%	77.80	63.26	142(x86) / 273(ARM) / 11.2(GPU)
	Ours (Search)	88.10%	**5.74**	**0.95**	**11(x86) / 15(ARM) / 1.9(GPU)**
JS-Printed	ResNet24LN (Teacher)	**99.24%**	120.00	33.96	98(x86) / 186(ARM) / 9.4(GPU)
	Ours (Search)	98.25%	**12.3**	**1.03**	**15(x86) / 18(ARM) / 1.9(GPU)**

Table 1: Comparison with existing Text-line recognition methods

4.5 Comparison with existing lightweight models

We conducted a comparison between our proposed method and existing lightweight models, including both manually designed and NAS models. To ensure fairness in the comparison, we multiplied the number of channels in all models by an appropriate width factor to achieve similar FLOPs. The results, presented in Table 2, clearly indicate that our model outperforms the existing methods by a significant margin, particularly in terms of the AR metric. Notably, our model even outperforms AutoSTR [50], which was specifically designed for scene text recognition tasks.

Model	ICDAR2013			IAM			SCUT-HCCDoc			JS-Printed		
	AR	Storage (MB)	FLOPs (G)	AR	Storage (MB)	FLOPs (G)	AR	Storage (MB)	FLOPs (G)	AR	Storage (MB)	FLOPs (G)
ShuffleNetV2 [27]	84.92%	10.5	0.98	89.52%	10.52	1.85	81.09%	9.72	0.96	89.61%	15.3	1.03
MobileNetV2 [35]	85.35%	8.23	**0.97**	89.22%	9.32	**1.14**	81.14%	8.1	0.96	93.71%	14.2	0.99
MobileNetV3 [17]	86.45%	10.7	1.00	90.00%	11.08	1.17	81.96%	14.7	1.34	93.60%	16.1	1.00
DARTS [25]	84.21%	11.0	0.99	89.37%	11.72	1.35	80.85%	10.9	1.04	86.61%	18.2	1.21
FBNetV2-F1 [41]	85.83%	12.1	0.99	89.77%	13.84	1.70	81.36%	11.6	0.97	92.95%	15.7	1.00
EfficientNet-B0 [39]	85.52%	12.5	1.00	89.52%	12.84	1.25	73.21%	19.2	1.02	88.36%	18.5	1.20
AutoSTR [50]	88.33%	**6.8**	0.99	90.80%	11.51	2.01	84.82%	6.33	0.97	95.37%	12.4	**0.97**
Ours	**91.86%**	8.34	1.26	**92.60%**	**8.7**	1.92	**88.10%**	**5.74**	**0.95**	**98.25%**	**12.3**	1.03

Table 2: Comparison with existing lightweight models

4.6 Ablation study

4.6.1 The effectiveness of the refined NAS search space

First, we study the effectiveness of the refined NAS search space. Specifically, for each layer of the over-parameterized network, we set 7 different operations including 6 different MCB and 1 skip connection. The maximum MCB kernel sizes in search space are set to $\{5, 7\}$, and expanded ratios are set to $\{2, 4, 6\}$. The candidate channels are set to $\{32, 36, 40, 44, 48\}$ in the first stage, and the number of channels in each subsequent layer is $1.5\times$ the number of channels in the previous layer. As shown in Table 3, the proposed MCB, channel-aware search and LN replacement can bring significant improvements step by step. With these components, the search model outperforms the ProxylessNAS method by 2.16%, 1.39%, 2.53% and 1.56% on ICDAR2013, IAM, SCUT-HCCDoc and JS-Printed respectively, which demonstrates the effectiveness of our proposed refinements to NAS.

Method	ICDAR2013	IAM	SCUT-HCCDoc	JS-Printed
	AR	AR	AR	AR
ProxylessNAS	88.38%	90.36%	84.08%	96.11%
+ MCB	89.43%(↑1.05%)	91.30%(↑0.94%)	84.99%(↑0.91%)	97.06%(↑0.95%)
+ MCB + CAS	90.11%(↑1.73%)	91.58%(↑1.22%)	85.92%(↑1.57%)	97.48%(↑1.37%)
MCB + CAS + LN	**90.54%(↑2.16%)**	**91.75%(↑1.39%)**	**86.88%(↑2.53%)**	**97.67%(↑1.56%)**

Table 3: Effectiveness of the refined NAS search space. 'MCB', 'CAS' and 'LN' denote Mobile Char Block, Channel-Aware Search and LayerNorm respectively.

4.6.2 Effectiveness of the proposed search with distillation mechanism

As shown in Table 4, the performance of the lightweight models searched with and without distillation are comparable under previous regular training. However, the result using our search model with KD achieves a significant improvement. This indicates that knowledge distillation can guide the NAS algorithm in finding a suitable architecture for distillation training and achieve a better accuracy-speed trade-off. We further compare the proposed TSVD-based distillation method with other KD methods. Comparison methods include relation-based [29, 40] and feature-based [34, 47] methods. The results from Tab. 4 show that our method obtains the best performance. In [34], FitNets uses a random weight regressor to do the feature matching. Although the performance of the student model is improved in distillation learning, its improvement is not as good as our method. This suggests that the TSVD of the classifier could provide more reasonable regressor parameters.

Model	Search With KD	Method	Types	ICDAR2013 AR	IAM AR	SCUT-HCCDoc AR	JS-Printed AR
ResNet24LN (Teacher)	-	-	-	92.91%	92.00%	89.09%	99.24%
NAS (Student)	✗	No Search	-	90.54%	91.75%	86.88%	97.67%
	✓	SP [40]	Relation-based	91.02%	92.12%	86.80%	98.07%
	✓	RKD [29]	Relation-based	90.97%	91.93%	86.98%	98.15%
	✓	FitNets [34]	Feature-based	90.58%	92.02%	85.01%	98.05%
	✓	AT [47]	Feature-based	91.07%	92.16%	87.08%	98.16%
	✓	**Ours**	Feature-based	**91.86%**	**92.60%**	**88.10%**	**98.25%**

Table 4: Effectiveness of the proposed search with existing knowledge distillation

5 Conclusions

In this paper, we introduce a TSVD-based Distillation-Guided NAS approach to search for a fast and compact text recognizer suitable for mobile application scenarios. We meticulously designed the mobile search space of NAS and the regressor of knowledge distillation for the text recognition task. Specifically, we proposed three methods: the Mobile Char Block (MCB), channel-aware search, and LN replacement. Additionally, we introduced Truncated Singular Value Decomposition (TSVD) into NAS for improved teacher-student knowledge distillation learning, resulting in a very fast and compact student model. Experiments demonstrate that the proposed search strategy can bring a better accuracy-speed trade-off for the lightweight model. On the IAM, ICDAR2013, SCUT-HCCDoc and JS-Printed benchmarks, our model achieved $12\times - 28.9\times$ faster, $6.7\times - 46\times$ smaller than previous text recognition models, while maintaining comparable recognition performance.

References

[1] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? *Advances in neural information processing systems*, 27, 2014.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[3] Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and simplifying one-shot architecture search. In *International conference on machine learning*, pages 550–559. PMLR, 2018.

[4] Andrew Brock, Theo Lim, JM Ritchie, and Nick Weston. Smash: One-shot model architecture search through hypernetworks. In *International Conference on Learning Representations (ICLR)*, 2018.

[5] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. In *International Conference on Learning Representations (ICLR)*, 2019.

[6] Kartik Chaudhary and Raghav Bali. Easter: Simplifying text recognition using only 1d convolutions. In *Canadian Conference on AI*, 2021.

[7] Haisong Ding, Kai Chen, and Qiang Huo. Compressing cnn-dblstm models for ocr with teacher-student learning and tucker decomposition. *Pattern Recognition*, 96:106957, 2019.

[8] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13733–13742, 2021.

[9] Yuning Du, Chenxia Li, Ruoyu Guo, Cheng Cui, Weiwei Liu, Jun Zhou, Bin Lu, Yehua Yang, Qiwen Liu, Xiaoguang Hu, et al. Pp-ocrv2: Bag of tricks for ultra lightweight ocr system. *arXiv preprint arXiv:2109.03144*, 2021.

[10] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

[11] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 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.

[12] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[13] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[14] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019.

[15] Yuhao Huang, Lianwen Jin, and Dezhi Peng. Zero-shot chinese text recognition via matching class embedding. In *International Conference on Document Analysis and Recognition*, pages 127–141. Springer, 2021.

[16] R Reeve Ingle, Yasuhisa Fujii, Thomas Deselaers, Jonathan Baccash, and Ashok C Popat. A scalable handwritten text recognition system. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 17–24. IEEE, 2019.

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[18] Xiaotang Jiang, Huan Wang, Yiliu Chen, Ziqi Wu, Lichuan Wang, Bin Zou, Yafeng Yang, Zongyang Cui, Yu Cai, Tianhang Yu, et al. Mnn: A universal and efficient inference engine. *Proceedings of Machine Learning and Systems*, 2:1–13, 2020.

[19] Xiao Jin, Baoyun Peng, Yichao Wu, Yu Liu, Jiaheng Liu, Ding Liang, Junjie Yan, and Xiaolin Hu. Knowledge distillation via route constrained optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1345–1354, 2019.

[20] Lei Kang, Pau Riba, Marçal Rusiñol, Alicia Fornés, and Mauricio Villegas. Pay attention to what you read: non-recurrent handwritten text-line recognition. *Pattern Recognition*, 129:108766, 2022.

[21] Bingcong Li, Xin Tang, Xianbiao Qi, Yihao Chen, and Rong Xiao. Hammingocr: A locality sensitive hashing neural network for scene text recognition. *arXiv preprint arXiv:2009.10874*, 2020.

[22] Changlin Li, Jiefeng Peng, Liuchun Yuan, Guangrun Wang, Xiaodan Liang, Liang Lin, and Xiaojun Chang. Block-wisely supervised neural architecture search with knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2020.

[23] Brian Liu, Weicong Sun, Wenjing Kang, and Xianchao Xu. Searching from the prediction of visual and language model for handwritten chinese text recognition. In *International Conference on Document Analysis and Recognition*, pages 274–288. Springer, 2021.

[24] Cheng-Lin Liu, Fei Yin, Da-Han Wang, and Qiu-Feng Wang. Casia online and offline chinese handwriting databases. In *2011 International Conference on Document Analysis and Recognition*, pages 37–41. IEEE, 2011.

[25] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *International Conference on Learning Representations (ICLR)*, 2019.

[26] Yu Liu, Xuhui Jia, Mingxing Tan, Raviteja Vemulapalli, Yukun Zhu, Bradley Green, and Xiaogang Wang. Search to distill: Pearls are everywhere but not the eyes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7539–7548, 2020.

[27] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pages 116–131, 2018.

[28] U-V Marti and Horst Bunke. The iam-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5(1):39–46, 2002.

[29] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3967–3976, 2019.

[30] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 268–284, 2018.

[31] Dezhi Peng, Lianwen Jin, Weihong Ma, Canyu Xie, Hesuo Zhang, Shenggao Zhu, and Jing Li. Recognition of handwritten chinese text by segmentation: A segment-annotation-free approach. *IEEE Transactions on Multimedia*, 2022.

[32] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *International conference on machine learning*, pages 4095–4104. PMLR, 2018.

[33] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4):1–34, 2021.

[34] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *International Conference on Learning Representations (ICLR)*, 2015.

[35] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[36] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2016.

[37] Baoguang Shi, Mingkun Yang, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Aster: An attentional scene text recognizer with flexible rectification. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2035–2048, 2018.

[38] Gilbert Strang, Gilbert Strang, Gilbert Strang, and Gilbert Strang. *Introduction to linear algebra*, volume 3. Wellesley-Cambridge Press Wellesley, MA, 1993.

[39] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[40] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1365–1374, 2019.

[41] Alvin Wan, Xiaoliang Dai, Peizhao Zhang, Zijian He, Yuandong Tian, Saining Xie, Bichen Wu, Matthew Yu, Tao Xu, Kan Chen, et al. Fbnetv2: Differentiable neural architecture search for spatial and channel dimensions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12965–12974, 2020.

[42] Tianwei Wang, Yuanzhi Zhu, Lianwen Jin, Canjie Luo, Xiaoxue Chen, Yaqiang Wu, Qianying Wang, and Mingxiang Cai. Decoupled attention network for text recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 12216–12224, 2020.

[43] Canyu Xie, Songxuan Lai, Qianying Liao, and Lianwen Jin. High performance offline handwritten chinese text recognition with a new data preprocessing and augmentation pipeline. In *International Workshop on Document Analysis Systems*, pages 45–59. Springer, 2020.

[44] Zecheng Xie, Yaoxiong Huang, Yuanzhi Zhu, Lianwen Jin, Yuliang Liu, and Lele Xie. Aggregation cross-entropy for sequence recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6538–6547, 2019.

[45] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4133–4141, 2017.

[46] Fei Yin, Qiu-Feng Wang, Xu-Yao Zhang, and Cheng-Lin Liu. Icdar 2013 chinese handwriting recognition competition. In *2013 12th international conference on document analysis and recognition*, pages 1464–1470. IEEE, 2013.

[47] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *International Conference on Learning Representations (ICLR)*, 2017.

[48] Matthew D Zeiler. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[49] Hesuo Zhang, Lingyu Liang, and Lianwen Jin. Scut-hccdoc: A new benchmark dataset of handwritten chinese text in unconstrained camera-captured documents. *Pattern Recognition*, 108:107559, 2020.

[50] Hui Zhang, Quanming Yao, Mingkun Yang, Yongchao Xu, and Xiang Bai. Autostr: efficient backbone search for scene text recognition. In *European Conference on Computer Vision*, pages 751–767. Springer, 2020.

[51] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2017.

[52] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.