Multi-modal In-Context Learning Makes an Ego-evolving Scene Text Recognizer

Zhen Zhao[1,2,*], Jingqun Tang[2,†], Chunhui Lin[2], Binghong Wu[2], Can Huang[2],
Hao Liu[2], Xin Tan[1], Zhizhong Zhang[1], Yuan Xie[1,†]
[1] East China Normal University [2]ByteDance
{51255901056}@stu.ecnu.edu.cn, {zzzhang,xtan,yxie}@cs.ecnu.edu.cn
{tangjingqun,linchunhui.26,wubinghong,haoliu.0128,can.huang}@bytedance.com

Abstract

Scene text recognition (STR) in the wild frequently encounters challenges when coping with domain variations, font diversity, shape deformations, etc. A straightforward solution is performing model fine-tuning tailored to a specific scenario, but it is computationally intensive and requires multiple model copies for various scenarios. Recent studies indicate that large language models (LLMs) can learn from a few demonstration examples in a training-free manner, termed "In-Context Learning" (ICL). Nevertheless, applying LLMs as a text recognizer is unacceptably resource-consuming. Moreover, our pilot experiments on LLMs show that ICL fails in STR, mainly attributed to the insufficient incorporation of contextual information from diverse samples in the training stage. To this end, we introduce E²STR, a STR model trained with context-rich scene text sequences, where the sequences are generated via our proposed in-context training strategy. E²STR demonstrates that a regular-sized model is sufficient to achieve effective ICL capabilities in STR. Extensive experiments show that E²STR exhibits remarkable training-free adaptation in various scenarios and outperforms even the fine-tuned state-of-the-art approaches on public benchmarks. The code is released at https://github.com/bytedance/E2STR.

1. Introduction

Scene Text Recognition (STR) is a fundamental task in computer vision, with extensive applications in several domains such as autonomous driving [45], augmented reality [30, 33], industrial print recognition [28] and visual understanding [26].

Current progress in STR [3, 17, 20, 35] has demonstrated remarkable performance in numerous scenarios.

However, as shown in Figure 1 (a), STR models are supposed to perform robustly over diversified scenarios in the real world, where the scene text is hard to recognize because of domain variation, font diversity, shape deformation, etc. As shown in Figure 1 (b), a straightforward solution involves collecting the corresponding data and then fine-tuning the model for the specific scenario [3, 17, 20]. This process is computationally intensive and requires multiple model copies for diverse scenarios.

The development of a comprehensive and reliable STR

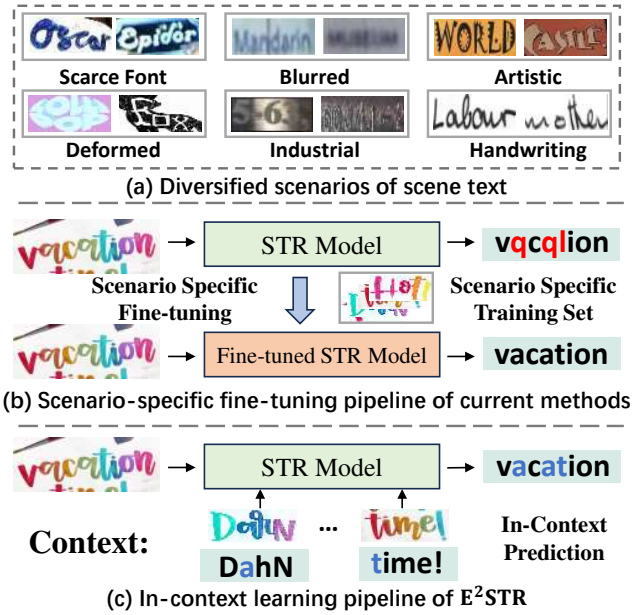

Figure 1. Demonstration of real-world scene text scenarios and the adaptation pipeline. (a) Diversified scenarios of scene text in the real world. (b) The adaptation pipeline of current methods. They typically have to fine-tune upon a trained STR model with the training set, under a specific scenario. (c) The adaptation pipeline of our proposed E²STR. Our method automatically selects in-context prompts and performs training-free adaptation when faced with novel scenarios. Blue characters denote ambiguous scene text that is easily misrecognized.

† Corresponding authors.
*This work is done when Zhen Zhao is an intern at ByteDance.

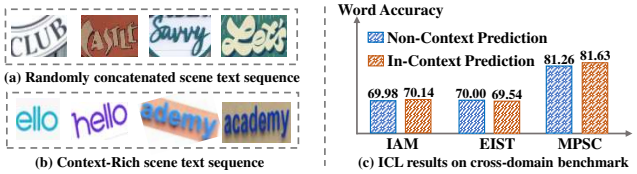

Figure 2. Our pilot experiments. (a) The randomly concatenated scene text sequence. (b) Our proposed context-rich scene text sequence. (c) By training an STR model based on the randomly concatenated scene text sequence, we evaluate the model on three cross-domain datasets.

model that can effectively handle many real-world scenarios remains a significant challenge.

Fortunately, plenty of studies [1, 6, 21, 38] have shown that Large Language Models (LLMs) can easily adapt without additional training. This adaptation is achieved by leveraging only a handful of input-label pairs as context (prompting information), a phenomenon known as "In-Context Learning" (ICL). The advantages of ICL inspire our interest in implementing it in STR, such that by fetching a few in-context prompts, a single model can be rapidly adapted to various scenarios without fine-tuning.

However, the equipment of ICL in STR still poses challenges under the existing circumstances. Firstly, it is deemed excessively costly to apply Multi-Modal Large Language Models (M-LLMs) with billions of parameters as a scene text recognizer. And the ICL capabilities in regular-sized models have been barely explored currently.

Secondly, it is hard to acquire ICL capabilities for a STR model with current training strategies. Previous studies have observed that sending image-text sequences for training would naturally endow ICL for M-LLMs [1, 21, 38], while such a phenomenon is hard to achieve in STR. As shown in Figure 2 (a), we generate sequential training data by randomly concatenating scene text samples. This practice fails as the trained model does not exhibit any performance improvement even when provided with in-domain prompts (Figure 2 (c)). The major cause of this failure is the lack of *context* in the generated scene text sequences during the training phase. The arbitrary concatenation of scene text does not provide any contextual information (*i.e.*, sample connections) between different samples (Figure 2 (a)). Consequently, the model lacks the ability to effectively use information derived from in-context prompts(Figure 2 (c)), which implies that *in-context training* is essentially important for the effective implementation of ICL in STR.

Based on the above analysis, we propose E^{2}STR (Ego-Evolving STR), a paradigm that facilitates adaptation across diverse scenarios in a training-free manner. Specifically, we propose an in-context training strategy, which enables the model to exploit contextual information from the generated context-rich scene text sequences (Figure 2 (b)). The context-rich scene text sequences are formed using our ST-strategy, which involves random **S**plitting and **T**ransformation of scene text, hence generating a set of "sub-samples". The sub-samples are inner-connected in terms of both visual and linguistic aspects. In the inference stage, E^{2}STR fetches in-context prompts based on visual similarities, and utilizes the prompts to assist the recognition, shown in Figure 1 (c). In practice, it is found that with proper training and inference strategies, ICL capabilities can also be observed in regular-sized STR models (hundreds of millions of parameters).

Finally, the proposed E^{2}STR effectively captures contextual information from the in-context prompts and performs rapid adaptation in various novel scenarios in a training-free manner (Please refer to Section 4.2). On common benchmarks, E^{2}STR achieves SOTA results, with an average improvement of 0.8% over previous methods and 1.1% over itself without ICL. Most importantly, when evaluated on unseen domains, E^{2}STR achieves impressive performance with only a few prompts, even outperforming the fine-tuning results of SOTA methods by 1.2%. Our contributions are summarized below:

(1) We propose E^{2}STR, a robust STR paradigm that can perform rapid adaptation over diverse scenarios in a training-free manner.

(2) We provide an in-context training strategy for equipping STR models with ICL capabilities, as well as an in-context inference strategy for STR models to leverage contextual information from in-context prompts.

(3) We demonstrate that ICL capabilities can be effectively incorporated into regular-sized STR models via appropriate training and inference strategies.

(4) Extensive experiments show that E^{2}STR exceeds state-of-the-art performance across diverse benchmarks, even surpassing the fine-tuned approaches in unseen domains.

2. Related Work

2.1. Scene Text Recognition

Recent years have witnessed extensive studies in STR, which can be generally divided into Language-free methods and Language-aware methods.

Language-free STR. Language-free models directly utilize visual features for prediction, without considering the relationship between the characters. In this branch CTC-based [11] methods [5, 24] play the most prominent part. They typically consist of a CNN for feature extraction and an RNN for sequential feature processing, which are trained end-to-end with the CTC loss [11]. Other methods like [23, 40] focus on treating STR as a character-level segmentation task. The lack of linguistic information limits the application of language-free methods in scenarios with

occluded or incomplete characters.

Language-aware STR. Language-aware models leverage linguistic information to assist the recognition, typically utilizing an external language model (LM) [10, 44] or training internal LMs [2, 7, 36]. SRN [44] and ABINet [10] feed visual predictions to an external LM for linguistic refinement. The direct application of an external LM without considering visual features leads to possible erroneous correction. On the other hand, methods like PARSeq [3] and MAERec [17] implicitly train an internal LM in an auto-regressive manner, which have achieved decent performance. In this paper we base our model on the language-aware design, training a transformer-based language decoder inner-connected with the vision encoder.

2.2. Multi-Modal In-Context Learning

Recent large language models (LLMs) [6, 46] have demonstrated their excellent few-shot adaptation capabilities. By concatenating a few examples with the input as the prompt at reference time, LLMs quickly adapt to novel tasks without parameter updating. This phenomenon introduces a novel learning paradigm termed "In-Context Learning". Meanwhile, unlike LLMs, vision-language models (VLMs) struggle to understand complex multi-modal prompts [47]. A large set of approaches [13–15, 34] have been proposed to empower VLMs with multi-modal in-context learning (M-ICL) capabilities, but they typically utilize vision models (like image caption models) to translate images to text [15, 34, 43], or view the LLM as a scheduler learning to call vision experts based on a few examples [13]. These approaches do not truly establish a VLM with M-ICL capabilities. Recently, several work [1, 21, 38] proposes to train VLMs with sequential multi-modal data, and have achieved great success in prompting VLMs with multi-modal examples. In this paper, we aim to train a scene text recognizer equipped with M-ICL capabilities based on this sequential training paradigm. We demonstrate that the arbitrary concatenation of scene text fails as stated above, which motivates us to generate context-rich scene text sequences.

3. Methodology

3.1. Preliminary of Multi-Modal In-Context Learning

Multi-modal in-context Learning enables M-LLMs to perform quick adaptation in downstream tasks in a training-free manner, hence eliminating the redundant computation and time expenses of fine-tuning. In this subsection, we introduce how to formulate multi-modal in-context learning for addressing the STR task.

For a scene text tuple $(\mathbf{x}, \mathbf{y})$ where $\boldsymbol{x}$ is the scene image and $\boldsymbol{y}$ is the ground-truth text, the STR task involves generating the label $\boldsymbol{y}$ by maximizing the conditional probability under the classic auto-regressive paradigm as follows: $p(\boldsymbol{y}|\boldsymbol{x}) = \prod_{l=1}^{L} p(\boldsymbol{y}_l|\boldsymbol{x}, \boldsymbol{y}_{<l})$, where $\boldsymbol{y}_l$ is the l-th character in $\boldsymbol{y}$, $\boldsymbol{y}_{<l}$ is the set of preceding characters, and L is the number of characters in $\boldsymbol{y}$.

While previous state-of-the-art studies typically need to fine-tune pre-trained models when confronted with novel scenarios [3, 17, 20], we propose in this study to leverage multi-modal in-context learning to enable STR models to be rapidly adapted across diverse scenarios without fine-tuning. Specifically, we define the probability of generating the target label $\boldsymbol{y}$ for a given image $\boldsymbol{x}$ and the multi-modal context C as follows:

$$p(\boldsymbol{y}|\boldsymbol{x}, C) = \prod_{l=1}^{L} p(\boldsymbol{y}_l|\{\underbrace{\boldsymbol{x}_1^c, \cdots, \boldsymbol{x}_n^c}_{\text{vision context}}; \boldsymbol{x}\}, \{\underbrace{\boldsymbol{y}_1^c, \cdots, \boldsymbol{y}_n^c}_{\text{language context}}; \boldsymbol{y}_{<l}\}), \tag{1}$$

where the context $C = \{(\boldsymbol{x}_1^c, \boldsymbol{y}_1^c), \cdots, (\boldsymbol{x}_n^c, \boldsymbol{y}_n^c)\}$ is the set of the in-context prompts, $(\boldsymbol{x}_i^c, \boldsymbol{y}_i^c)$ are the scene image and the ground-truth text of the context prompts, and n is the number of context prompts.

3.2. Framework Overview and Model Architecture

Our proposed E^{2}STR consists of three stages. Firstly, E^{2}STR is trained in the standard auto-regressive framework to learn the fundamental STR ability.

Secondly, as shown in the top of Figure 3, E^{2}STR is further trained based on our proposed In-Context Training paradigm. In this stage E^{2}STR learns to understand the connection between different samples, allowing it to profit from in-context prompts. Finally, as shown in the bottom of Figure 3, E^{2}STR fetches in-context prompts based on visual similarity during inference, allowing the test sample to absorb context information.

As shown in the top of Figure 3, the model architecture of E^{2}STR consists of a vision encoder and a language decoder. The vision encoder receives image inputs and the language decoder processes text inputs in an auto-regressive manner. Following [1], a set of cross attention layers are utilized to bridge the output tokens of the vision encoder and the language decoder. Under the ICL framework, the vision encoder receives numerous images as input. To control the length of the vision token sequence, a fixed number of query tokens are learned by performing cross attention against the output tokens of the vision encoder.

3.3. Training Strategy

Our training process is split into two stages: vanilla STR training and in-context STR training.

3.3.1 Vanilla Scene Text Recognition Training

The first training phase seeks to provide E^{2}STR with the fundamental skills in STR. For a scene text tuple $(\boldsymbol{x}, \boldsymbol{y})$ the

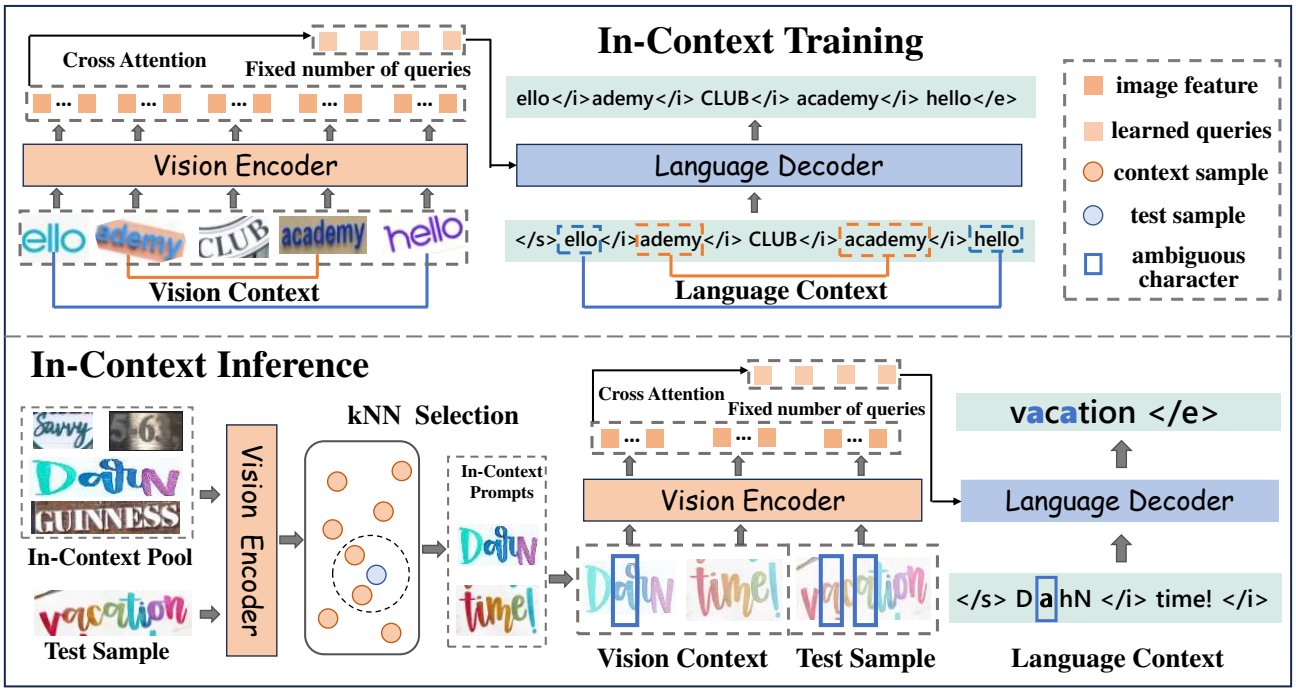

Figure 3. Pipeline of our E^{2}STR. Top: E^{2}STR is trained with our in-context training strategy to obtain the ICL capability. Down: During inference, E^{2}STR selects in-context prompts based on a kNN strategy, then the test sample grasps context information from the prompts to assist the recognition. Specifically, the ambiguous character "a" in the test sample is easily misrecognized as "q". With the vision-language context produced by the in-context prompts (*i.e.*, "a" in the first in-context prompt), E^{2}STR rectifies the result. Note that in practice the in-context pool maintains image tokens and thus does not need to go through the vision encoder.

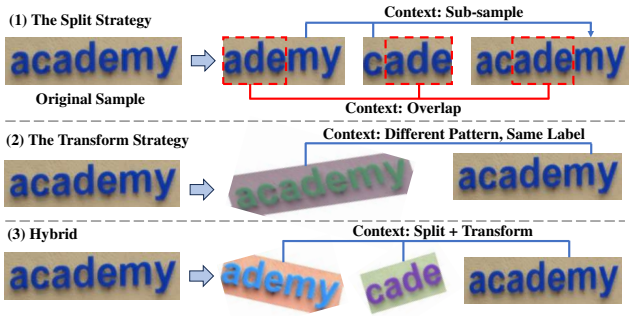

Figure 4. Illustration of the split strategy, the transform strategy, and how we hybrid them in practice.

input to the vision encoder is $\boldsymbol{x}$ and the initial input to the language decoder is a start token </s>. The training in this phase makes use of the next-token prediction loss:

$$\mathcal{L} = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\left[-\sum_{l=1}^{L}\log p(\boldsymbol{y}_l|\boldsymbol{y}_{<l},\boldsymbol{x})\right], \qquad (2)$$

where $\mathcal{D}$ is the training set.

3.3.2 In-Context Training

The objective of the in-context training phase is to equip E^{2}STR with the capability of In-Context Learning. As depicted in the top of Figure 3, the model is trained with context-rich scene text sequences as stated before. In these sequences, we interleave a placeholder </i> in the text for each image. This serves to make the language decoder distinguish between different samples following [1]. In this stage, we propose two strategies to generate context-rich scene text sequences: the Split Strategy and the Transform Strategy (the ST strategy).

The Split Strategy. As shown in Figure 4 (a), when presented with a training tuple $(\boldsymbol{x}, \boldsymbol{y})$, we split the sample and hence generating a set of "sub-samples". It is evident that the sub-samples exhibit a strong connection to the original training sample. Furthermore, the sub-samples themselves demonstrate interconnectivity as they overlap with one another. Next, we proceed to concatenate the sub-samples with $(\boldsymbol{x}, \boldsymbol{y})$ and additional randomly selected samples to form a context-rich sample sequence.

We randomly shuffle the whole sequence before generating the actual input text (*i.e.*, interleaving the </i> token to the text sequence).

In practice, to accurately split the training samples, we

synthesize 600k scene text images based on [4] and record the accurate bounding boxes of every single character. Our subsequent experiments show that the synthesized data does not change E^{2}STR's non-context text recognition ability, but the Split Strategy based on them equips E^{2}STR with a strong capability of in-context learning.

The Transform Strategy. As shown in Figure 4 (b), given a training tuple $(\boldsymbol{x}, \boldsymbol{y})$ (whether with character-wise bounding boxes or not), we perform data augmentation (a set of image transformations, *e.g.,* color/direction transformations) on $\boldsymbol{x}$. In this way, we also generate a set of sub-samples with the same label but different image patterns from the original sample.

In practice, as depicted in Figure 4 (c), we hybrid the above strategies. The training set is formed by concatenating the synthesized data and the original training data used in the first training phase. For the synthesized data with character-wise bounding boxes, both the Split Strategy and the Transform Strategy are utilized. For the original training data, only the Transform Strategy is implemented.

Finally, after generating the sample sequence $(\boldsymbol{X}, \boldsymbol{Y})$, where $\boldsymbol{X}$ is the image sequence and $\boldsymbol{Y}$ is the text sequence, $\boldsymbol{X}$ is fed into the vision encoder, while $\boldsymbol{Y}$ is processed by the language decoder under the auto-regressive framework. The loss function is formulated as:

$$\mathcal{L}_{(\boldsymbol{X},\boldsymbol{Y})} = -\sum_{l=1}^{L} \log p(\boldsymbol{Y}_l | \boldsymbol{Y}_{<l}, \boldsymbol{X}_{\leq l}), \tag{3}$$

where $\boldsymbol{X}_{\leq l}$ is the set of image tokens preceding token $\boldsymbol{Y}_l$ in the input sequence.

3.4. In-Context Inference

The In-Context Learning ability is acquired by our E^{2}STR model through the above two-stage training approach. AS shown in the bottom of Figure 3, when presented with a test image $\boldsymbol{x}$, the framework selects N samples $\{(\boldsymbol{x}_i^c, \boldsymbol{y}_i^c)\}_{i=1}^N$ from a in-context pool $\mathcal{D}^c$. The selected samples have the highest visual similarities to $\boldsymbol{x}$ in the latent space. Specifically, we calculate the image embedding $\boldsymbol{I}$ of $\boldsymbol{x}$ by averaging the visual token sequence $Encoder(\boldsymbol{x})$. The in-context prompts are then formed by choosing N samples from $\mathcal{D}^c$, where the image embeddings of these samples have the top-N highest cosine similarity with $\boldsymbol{I}$, *i.e.,*

$$\mathcal{I} = \underset{i \in 1,2,\cdots,|\mathcal{D}^c|}{\text{argTopN}} \frac{\boldsymbol{I}^{\mathrm{T}} \boldsymbol{I}_i^c}{\|\boldsymbol{I}\|_2 \|\boldsymbol{I}_i^c\|_2}, \tag{4}$$

where $\mathcal{I}$ is the index set of the top-N similar samples in $\mathcal{D}^c$, and $\boldsymbol{I}_i^c$ is the image embedding of the i-th sample in $\mathcal{D}^c$. The in-context prompts are then defined as:

$$\boldsymbol{E} = \{(\boldsymbol{x}_i^c, \boldsymbol{y}_i^c) | i \in \mathcal{I}\}. \tag{5}$$

As shown in the bottom of Figure 3, $\boldsymbol{E}$ is concatenated with the test sample $\boldsymbol{x}$ and our in-context prediction is formulated as $p(\boldsymbol{y}|\boldsymbol{E}, \boldsymbol{x})$. In practice, the in-context pool $\mathcal{D}^c$ retains solely the output tokens generated by the vision encoder, resulting in a highly efficient selection process. Furthermore, because the in-context pool is tiny and we do straight inference without training, the extra consumption is kept to a minimum (Please refer to Section 4.3).

4. Experiment

4.1. Experimental Setup

Implementation Details. Following MAERec [17], we choose Vision Transformer [9] pre-trained under the MAE [16] framework as the vision encoder. The default language decoder is set as OPT-125M [46]. We use the cosine learning rate scheduler without warm-up and the AdamW optimizer with a weight decay of 0.01. We train our model for 10 epochs with an initial learning rate of 1e-4 during the first training stage, and 5 epochs with an initial learning rate of 5e-6 during the second in-context training stage. The training batch size is 64 for the first stage and 8 for the second stage. During inference for E^{2}STR-ICL, we select two in-context prompts based on the kNN selection strategy.

Datasets and Metrics. We use the real-world training dataset Union14M-L[17] for the two-stage training. The same training dataset (including the synthesized data) is adopted for all compared methods. E^{2}STR is evaluated under various publicly available benchmarks, including Regular Benchmarks IIIT5k [29], SVT [37], IC13 [18], Irregular Benchmarks IC15 [19], SVTP [31], CUTE80 (CT80) [32], COCO Text (COCO) [39], CTW [25], Total Text (TT) [8], Occluded Benchmarks OST (HOST and WOST) [41] and artistic benchmark WordArt [42]. In cross domain scenarios the evaluated datasets including the metal-surface benchmark MPSC [12] and the handwriting benchmark IAM [27], as well as a more difficult real-world industrial text recognition dataset EIST (Enhanced Industrial Scene Text) collected by us. EIST is collected from the real-world industrial scenario, which contains 200 training samples and 8000 test samples. We use Word Accuracy[17] as the evaluation metric for all compared methods.

4.2. Main Results

4.2.1 Results on Common Benchmarks

Table 1 presents the performance of E^{2}STR on common benchmarks. We evaluate E^{2}STR on 12 commonly used STR benchmarks and compare with SOTA methods. E^{2}STR-base refers to non-context prediction without prompts. For E^{2}STR-ICL, a tiny in-context pool is maintained by randomly sampling 1000 images from the training data (less than 0.025% of the number of training samples). As we can see, E^{2}STR-base achieves 90.25% average word

Method	Venue	Regular			Irregular						Occluded		Others	AVG
		IIIT	SVT	IC13	IC15	SVTP	CT80	COCO	CTW	TT	HOST	WOST	WordArt	
		3000	647	1015	2077	645	288	9896	1572	2201	2416	2416	1511	
ASTER [36]	PAMI'18	95.03	89.49	93.79	85.48	82.02	90.28	62.25	76.53	78.69	43.34	64.65	65.59	77.26
NRTR [35]	ICDAR'19	97.43	93.82	96.06	85.15	84.03	91.32	65.94	81.74	81.83	50.83	71.52	64.06	80.31
SAR [22]	AAAI'19	97.70	94.13	96.35	87.47	87.60	93.06	67.41	83.91	86.23	46.36	70.32	72.40	81.91
SATRN [20]	AAAI'20	97.83	95.83	97.44	89.46	90.85	96.18	73.06	84.61	87.91	56.71	75.62	75.71	85.10
ABINet [10]	CVPR'21	97.90	95.98	96.16	91.66	90.23	93.75	71.46	83.72	86.01	56.54	75.75	75.25	84.53
PARSeq* [3]	ECCV'22	99.10	97.84	98.13	89.22	96.90	98.61	-	-	-	-	-	-	-
MAERec [17]	ICCV'23	98.93	97.99	98.62	93.04	94.57	98.96	78.84	88.87	93.91	73.97	85.72	82.59	90.50
E^{2}STR-base		99.10	98.15	98.03	92.99	96.43	98.96	77.29	88.36	93.46	73.30	85.51	81.47	90.25
E^{2}STR-ICL		99.23	98.61	98.72	93.82	96.74	99.31	78.38	88.99	94.68	74.75	86.59	86.17	91.33

Table 1. Results on common benchmarks. All methods are trained on the same dataset except for PARSeq. *: PARSeq is trained on its self-collected real-world dataset and we directly quote the results from its original paper. Red and blue values denote the best and the secondary performance. E^{2}STR-base refers to non-context inference.

Method	Industrial		Handwriting	AVG
	MPSC	EIST	IAM	
	2941	8000	3000	
ASTER [36]	63.48	48.76	52.50	54.91
NRTR [35]	73.24	61.77	59.53	64.85
SAR [22]	73.85	58.26	56.63	62.91
ABINet [10]	75.35	62.85	61.57	66.59
SATRN [20]	76.10	65.42	59.47	67.00
MAERec [17]	81.81	70.33	70.27	74.14
E^{2}STR-base	81.26	69.66	69.51	73.48
E^{2}STR-ICL	83.64	76.77	74.10	78.17

Table 2. Results on cross domain scenarios. Three datasets under two unseen domains are evaluated. All approaches are evaluated in a training-free manner.

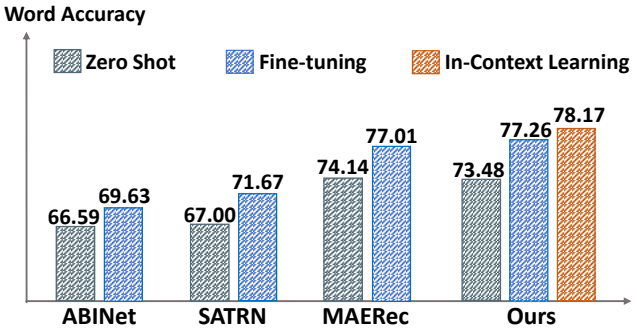

Figure 5. Comparison with the fine-tuned models. We report the average performance on three cross-domain datasets. Please note that ABINet [10], SATRN [20] and MAERec [17] are fine-tuned with the in-domain data, while our E^{2}STR-ICL is training-free.

accuracy over 12 datasets, 0.25% lower than MAERec [17]. However, by fetching in-context prompts and exploiting in-context information, E^{2}STR-ICL achieves an average word accuracy of 91.33%, which is 1.08% higher than E^{2}STR-base and 0.83% higher than MAERec. Please note that this improvement is automatic and training-free.

Specifically, on the six traditional STR benchmarks (*i.e.*, IIIT, SVT, IC13, IC15, SVTP, and CT80) which have nearly reached saturation in recent years[17], E^{2}STR still push the performance limit from 97.02% to 97.74%, leading to a 24% error rate decrease. On the 6 larger and harder STR benchmarks (*i.e.*, COCO Text, CTW, TT, HOST, and WOST), E^{2}STR-ICL outperforms MAERec by 0.94%.

4.2.2 Results on Cross Domain Scenarios

We compare with SOTA methods on cross domain benchmarks. Two novel scenarios are introduced: the industrial scenario (MPSC and EIST) and the handwriting scenario (IAM). In each dataset, only 100 training samples are provided. For E^{2}STR-ICL we simply use the training samples as the in-context pool. We compare the training-free results in Table 2 and the fine-tuning results in Figure 5.

As we can see, on both industrial and handwriting scenarios our E^{2}STR-ICL reaches SOTA performance. As shown in Table 2, under the training-free constraint E^{2}STR-ICL reaches an average performance of 78.17%, which is 4.69% higher than E^{2}STR-base and 4.03% higher than the SOTA method MAERec. Specifically, on EIST and IAM the application of ICL brings a huge improvement of 7.11% and 4.59%, which demonstrates the extraordinary adaptation ability of E^{2}STR-ICL.

We further compare the fine-tuned methods and our E^{2}STR-ICL. We fine-tune ABINet [10], SATRN [20] and MAERec [17] with the same data preserved in the in-context pool. As shown in Figure 5, E^{2}STR-ICL outperforms MAERec by 1.16% even if the latter is fine-tuned with in-domain data, which is an exciting result given that E^{2}STR-ICL requires no parameter updating. In a word, our E^{2}STR can be rapidly implemented in a training-free manner in various novel scenarios and even achieves better per-

	COCO		HOST		WordArt	
annotation rate	10%	20%	10%	20%	10%	20%
MAERec [17]	0	0	0	0	0	0
w/ fine-tuning	0.82	1.67	1.03	1.72	1.34	2.23
E^{2}STR-base	0	0	0	0	0	0
E^{2}STR-ICL	10.12	12.92	12. 43	13.76	26.22	32.02

Table 3. Results on hard case rectification. "Hard Cases" are test samples misrecognized by both MAERec [17] and our E^{2}STR-base. By providing annotation of a small part of the hard cases, we compare the performance increase in the rest test samples between the fine-tuned MAERec and our E^{2}STR-ICL.

Training Task			Word Accuracy	
VT	TS	SS	Non-Context	In-Context
✓			69.69	26.82
✓	✓		69.80	75.60
✓		✓	69.66	73.09
✓	✓	✓	69.66	76.77

Table 4. Ablation of our proposed training strategies, where VT, TS, and SS refer to vanilla STR training, the Transform Strategy, and the Split Strategy. The experiment is performed on EIST.

formance than the fine-tuned SOTA methods.

4.2.3 Results on Hard Case Rectification

We demonstrate the rectification ability of E^{2}STR, which can handle hard cases in STR conveniently and effectively, in a training-free manner. Specifically, we define "hard cases" as the scene text samples that are wrongly recognized by both E^{2}STR-base and the SOTA method MAERec. A small number of hard cases are then annotated, and we study how the model can benefit from the annotated hard cases and decrease the error rate of the rest hard cases. Shown in Table 3, we perform experiments on COCO Text, HOST, and WordArt. As we can see, by annotating 10% to 20% of the hard cases, E^{2}STR-ICL decreases the error rate of the rest hard cases by up to 32%. This improvement is achieved by simply putting the annotated hard cases into the in-context pool, without any hassle of re-training the model. By comparison, by fine-tuning on the annotated hard cases, MAERec only decreases the error rate by up to 2.23%, completely incomparable to our E^{2}STR-ICL. As a result, E^{2}STR-ICL can rapidly learn from hard cases and improve the performance in a training-free manner, while SOTA methods like MAERec can hardly benefit from hard samples even with fine-tuning.

4.3. Ablation Studies

Impact of Split-and-Transform Training Strategies. We perform an experiment to show the effectiveness of our proposed Split Strategy and Transform Strategy. Shown in

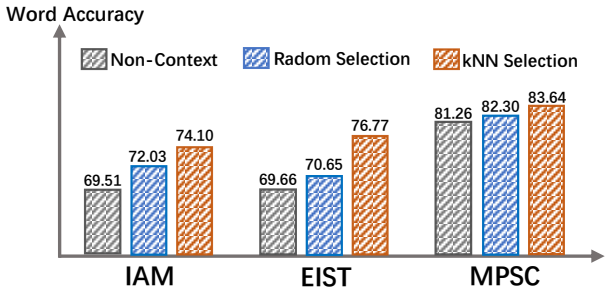

Figure 6. Comparison between different in-context prompt selection strategies. "Random Selection" refers to randomly selecting two samples as in-context prompts from the in-context pool. X-axis is the evaluated benchmarks.

Table 4, the vanilla STR training brings a word accuracy of 69.69%, but the model cannot understand context information, and the performance even severely decreases to 26.82% when provided with in-context prompts. The application of the Transform Strategy and the Split Strategy in the second training stage does not increase the non-context performance (concerning that the synthesized data is typically weaker than the real-world data used in the vanilla training stage), but the model learns to profit from context, and the performance is improved to 75.60% and 73.09% respectively when provided with in-context prompts. Finally, the hybrid of the above two strategies further enhances the ICL ability, and the performance reaches 76.77%.

Impact of Nearest Neighbor In-Context Prompt Selection. In Section 3.4 we propose to select samples most similar to the test image in the latent space based on the kNN strategy. Here we demonstrate the effectiveness of this strategy by comparing the performance to Random Selection, *i.e.*, randomly selecting in-context prompts from the in-context pool. Shown in Figure 6, on all three evaluated datasets, random selection can improve the performance of non-context prediction by a small margin, but is far from comparing with kNN selection. Specifically, on EIST random selection improves the performance of non-context from 69.66% to 70.65%, while kNN selection reaches 76.77% word accuracy under the same condition.

Impact of In-Context Pool Size. We next study the impact of varying the size of the in-context pool. Shown in Figure 7, we perform experiments on IAM, EIST, and MPSC, by varying the number of samples maintained in the in-context pool. As we can see, in general, the larger in-context pool brings about better performance, and this improvement effect weakens as the pool continually expands. To be specific, on IAM the word accuracy is increased from 69.51% to 74.10% (4.59% improvement) when the pool size is 100, while it only increases the performance from 74.10% to 75.50% (1.40% improvement) when the pool is expanded with another 100 samples. The above fact implies that a

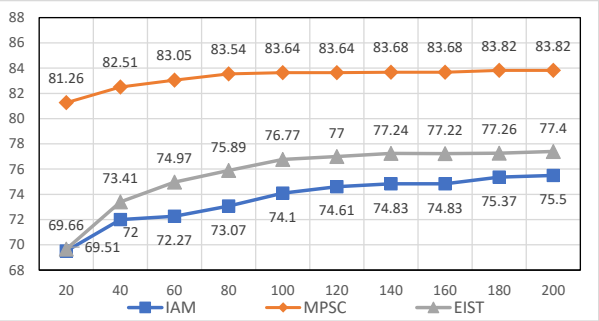

Figure 7. The performance change brought by different sizes of the in-context pool. The X-axis is the size of the in-context pool and the Y-axis is the word accuracy results.

ICL prompts	0	1	2	4	8	16
HOST	73.30	74.34	74.75	74.83	74.83	74.92
TT	93.46	94.68	94.68	94.78	94.88	94.91
WordArt	81.47	86.04	86.17	86.17	86.27	86.35

Table 5. The performance change brought by the different number of in-context prompts.

	MAERec [17]	E^{2}STR-base	E^{2}STR-ICL
Inference Time (s)	0.092	0.071	0.094

Table 6. Comparison of the mean inference time of each test sample. All results are reported under the same hardware environment.

small number of samples is adequate to bring about huge performance improvement when deploying E^{2}STR-ICL.

Impact of the Number of In-Context prompts. We analyze the influence of the number of in-context prompts. Shown in Table 5, the experiment is performed on HOST, ToTal Text, and WordArt. Similar to the in-context pool size, the increase in the number of in-context prompts also generally boosts the performance of E^{2}STR-ICL. However, as we can see, one to two in-context prompts are adequate for improving the performance by a large margin, and the further increase of in-context prompts brings about a limited improvement. This phenomenon is possibly caused by the fact that usually only a few characters are wrongly recognized for a bad case, which can be rectified by the context information from one or two in-context prompts.

Computational Complexity. We experimentally compare the inference speed of E^{2}STR and MAERec [17]. Shown in Table 6, the inference speed of E^{2}STR-ICL is on par with MAERec. Compared to E^{2}STR-base, the in-context prompts of E^{2}STR-ICL bring extra consumption, but this leads to a limited inference time increase (*i.e.*, from 0.071 to 0.094). It makes sense since we only maintain the visual tokens in the in-context pool and directly feed the visual tokens of the selected prompts to the language model.

Visualization and Further Analysis. We further study how the test sample learns from context. Shown in Figure 8, we select one context prompt for the test sample, and study the model pays attention to which region of the context image. This is achieved by collecting the attention maps between the language tokens and the image features. As we can see, when the language tokens pay close attention to the misrecognized image region, they also focus on the context image region which has similar patterns. For example, on the last row of Figure 8, E^{2}STR misrecognized the test image as "simplest" without context. By providing a context prompt "Display", one language token focuses on the "la" region of both images, which have similar image patterns. Finally, E^{2}STR rectified the misrecognized "e" to "a" with the help of context ground-truth "la" of the focused region.

Figure 8. Cross attention visualization between the language tokens and the vision tokens. Left: Non-context prediction of E^{2}STR. Error characters are marked in red. Right: In-context prediction of E^{2}STR-ICL, where only one in-context prompt is selected. We visualize how the language tokens attend to the prompt image and the test image.

5. Limitations

There are two limitations in our study. Firstly, there is a very slim chance that E^{2}STR-ICL erroneously rectifies predictions due to misleading prompts (please refer to supplementary materials). Additionally, our model still could not recognize characters that are not included in the lexicon.

6. Conclusion

In this paper, we propose E^{2}STR, an ego-evolving scene text recognizer equipped with in-context learning capabilities. Through our proposed in-context training strategy incorporating context-rich scene text sequences, E^{2}STR performs rapid adaptation across diverse scenarios without additional fine-tuning. Extensive experiments demonstrate that E^{2}STR not only achieves SOTA performance on com-

mon STR benchmarks but also outperforms even the approaches that have been fine-tuned specifically for cross-domain scenarios. The model's ability to easily and effectively handle difficult text cases further underscores its potential as a unified text recognizer. Overall, this research represents a significant step toward efficient and highly adaptive text recognition models well-suited for diverse real-world applications.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022. 2, 3, 4, 1

[2] Jeonghun Baek, Geewook Kim, Junyeop Lee, Sungrae Park, Dongyoon Han, Sangdoo Yun, Seong Joon Oh, and Hwalsuk Lee. What is wrong with scene text recognition model comparisons? dataset and model analysis. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4715–4723, 2019. 3

[3] Darwin Bautista and Rowel Atienza. Scene text recognition with permuted autoregressive sequence models. In *European Conference on Computer Vision*, pages 178–196. Springer, 2022. 1, 3, 6

[4] Belval. Generator. https://github.com/Belval/TextRecognitionDataGenerator. 5

[5] Fedor Borisyuk, Albert Gordo, and Viswanath Sivakumar. Rosetta: Large scale system for text detection and recognition in images. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 71–79, 2018. 2

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 2, 3

[7] Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, and Shuigeng Zhou. Focusing attention: Towards accurate text recognition in natural images. In *Proceedings of the IEEE international conference on computer vision*, pages 5076–5084, 2017. 3

[8] Chee Kheng Ch'ng and Chee Seng Chan. Total-text: A comprehensive dataset for scene text detection and recognition. In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, pages 935–942. IEEE, 2017. 5

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 5

[10] Shancheng Fang, Hongtao Xie, Yuxin Wang, Zhendong Mao, and Yongdong Zhang. Read like humans: Autonomous, bidirectional and iterative language modeling for scene text recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7098–7107, 2021. 3, 6

[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. 2

[12] Tongkun Guan, Chaochen Gu, Changsheng Lu, Jingzheng Tu, Qi Feng, Kaijie Wu, and Xinping Guan. Industrial scene text detection with refined feature-attentive network. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(9):6073–6085, 2022. 5

[13] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14953–14962, 2023. 3

[14] Jiabang He, Lei Wang, Yi Hu, Ning Liu, Hui Liu, Xing Xu, and Heng Tao Shen. Icl-d3ie: In-context learning with diverse demonstrations updating for document information extraction. *arXiv preprint arXiv:2303.05063*, 2023.

[15] Jiabang He, Lei Wang, Yi Hu, Ning Liu, Hui Liu, Xing Xu, and Heng Tao Shen. Icl-d3ie: In-context learning with diverse demonstrations updating for document information extraction. *arXiv preprint arXiv:2303.05063*, 2023. 3

[16] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 5

[17] Qing Jiang, Jiapeng Wang, Dezhi Peng, Chongyu Liu, and Lianwen Jin. Revisiting scene text recognition: A data perspective. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20543–20554, 2023. 1, 3, 5, 6, 7, 8

[18] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazan Almazan, and Lluis Pere De Las Heras. Icdar 2013 robust reading competition. In *2013 12th international conference on document analysis and recognition*, pages 1484–1493. IEEE, 2013. 5

[19] Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, et al. Icdar 2015 competition on robust reading. In *2015 13th international conference on document analysis and recognition (ICDAR)*, pages 1156–1160. IEEE, 2015. 5

[20] Junyeop Lee, Sungrae Park, Jeonghun Baek, Seong Joon Oh, Seonghyeon Kim, and Hwalsuk Lee. On recognizing texts of arbitrary shapes with 2d self-attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 546–547, 2020. 1, 3, 6

[21] Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Jingkang Yang, and Ziwei Liu. Otter: A multi-modal model with in-context instruction tuning. *arXiv preprint arXiv:2305.03726*, 2023. 2, 3

[22] Hui Li, Peng Wang, Chunhua Shen, and Guyu Zhang. Show, attend and read: A simple and strong baseline for irregular text recognition. In *Proceedings of the AAAI conference on artificial intelligence*, pages 8610–8617, 2019. 6

[23] Minghui Liao, Jian Zhang, Zhaoyi Wan, Fengming Xie, Jiajun Liang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Scene text recognition from two-dimensional perspective. In *Proceedings of the AAAI conference on artificial intelligence*, pages 8714–8721, 2019. 2

[24] Wei Liu, Chaofeng Chen, Kwan-Yee K Wong, Zhizhong Su, and Junyu Han. Star-net: a spatial attention residue network for scene text recognition. In *BMVC*, page 7, 2016. 2

[25] Yuliang Liu, Lianwen Jin, Shuaitao Zhang, Canjie Luo, and Sheng Zhang. Curved scene text detection via transverse and longitudinal sequence connection. *Pattern Recognition*, 90: 337–345, 2019. 5

[26] Mengkai Ma, Qiu-Feng Wang, Shan Huang, Shen Huang, Yannis Goulermas, and Kaizhu Huang. Residual attention-based multi-scale script identification in scene text images. *Neurocomputing*, 421:222–233, 2021. 1

[27] 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: 39–46, 2002. 5

[28] Qiang Mei, Qinyou Hu, Chun Yang, Hailin Zheng, and Zhisheng Hu. Port recommendation system for alternative container port destinations using a novel neural language-based algorithm. *IEEE Access*, 8:199970–199979, 2020. 1

[29] Anand Mishra, Karteek Alahari, and CV Jawahar. Top-down and bottom-up cues for scene text recognition. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2687–2694. IEEE, 2012. 5

[30] Imene OUALI, Mohamed BEN HALIMA, and WALI Ali. Augmented reality for scene text recognition, visualization and reading to assist visually impaired people. *Procedia Computer Science*, 207:158–167, 2022. 1

[31] Trung Quy Phan, Palaiahnakote Shivakumara, Shangxuan Tian, and Chew Lim Tan. Recognizing text with perspective distortion in natural scenes. In *Proceedings of the IEEE international conference on computer vision*, pages 569–576, 2013. 5

[32] Anhar Risnumawan, Palaiahankote Shivakumara, Chee Seng Chan, and Chew Lim Tan. A robust arbitrary text detection system for natural scene images. *Expert Systems with Applications*, 41(18):8027–8048, 2014. 5

[33] Abdul Khader Jilani Saudagar and HabeebVulla Mohammad. Augmented reality mobile application for arabic text extraction, recognition and translation. *Journal of Statistics and Management Systems*, 21(4):617–629, 2018. 1

[34] Zhenwei Shao, Zhou Yu, Meng Wang, and Jun Yu. Prompting large language models with answer heuristics for knowledge-based visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14974–14983, 2023. 3

[35] Fenfen Sheng, Zhineng Chen, and Bo Xu. Nrtr: A no-recurrence sequence-to-sequence model for scene text recognition. In *2019 International conference on document analysis and recognition (ICDAR)*, pages 781–786. IEEE, 2019. 1, 6

[36] 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. 3, 6

[37] Cunzhao Shi, Chunheng Wang, Baihua Xiao, Song Gao, and Jinlong Hu. End-to-end scene text recognition using tree-structured models. *Pattern Recognition*, 47(9):2853–2866, 2014. 5

[38] Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021. 2, 3

[39] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. *arXiv preprint arXiv:1601.07140*, 2016. 5

[40] Zhaoyi Wan, Minghang He, Haoran Chen, Xiang Bai, and Cong Yao. Textscanner: Reading characters in order for robust scene text recognition. In *Proceedings of the AAAI conference on artificial intelligence*, pages 12120–12127, 2020. 2

[41] Yuxin Wang, Hongtao Xie, Shancheng Fang, Jing Wang, Shenggao Zhu, and Yongdong Zhang. From two to one: A new scene text recognizer with visual language modeling network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14194–14203, 2021. 5

[42] Xudong Xie, Ling Fu, Zhifei Zhang, Zhaowen Wang, and Xiang Bai. Toward understanding wordart: Corner-guided transformer for scene text recognition. In *European Conference on Computer Vision*, pages 303–321. Springer, 2022. 5

[43] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. An empirical study of gpt-3 for few-shot knowledge-based vqa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3081–3089, 2022. 3

[44] Deli Yu, Xuan Li, Chengquan Zhang, Tao Liu, Junyu Han, Jingtuo Liu, and Errui Ding. Towards accurate scene text recognition with semantic reasoning networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12113–12122, 2020. 3

[45] Chongsheng Zhang, Yuefeng Tao, Kai Du, Weiping Ding, Bin Wang, Ji Liu, and Wei Wang. Character-level street view text spotting based on deep multisegmentation network for smarter autonomous driving. *IEEE Transactions on Artificial Intelligence*, 3(2):297–308, 2021. 1

[46] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022. 3, 5

[47] Haozhe Zhao, Zefan Cai, Shuzheng Si, Xiaojian Ma, Kaikai An, Liang Chen, Zixuan Liu, Sheng Wang, Wenjuan Han,

and Baobao Chang. Mmicl: Empowering vision-language model with multi-modal in-context learning. *arXiv preprint arXiv:2309.07915*, 2023. 3

Multi-modal In-Context Learning Makes an Ego-evolving Scene Text Recognizer

Supplementary Material

7. Model Architecture

Figure 9 presents the detailed model architecture of E^{2}STR. We follow the paradigm established by Flamingo [1], where we perform cross attention between the vision outputs and the language outputs in each language model layer. The language outputs serve as queries and the vision outputs serve as keys and values. The detailed configures of the vision encoder and the language decoder are summarized in Table 7. For fair comparison, we provide MAERec [17] with the same language decoder with E^{2}STR-ICL (We name this modification as MAERec†). The comparison between MAERec† and E^{2}STR is shown in Table 8.

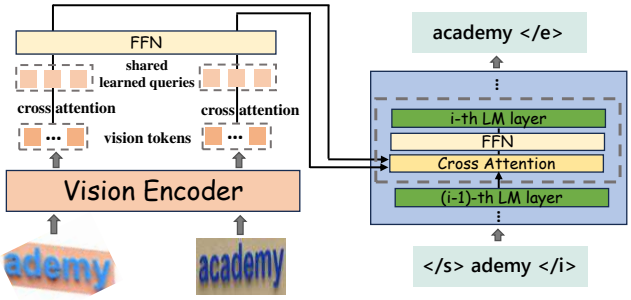

Figure 9. Detailed Model Architecture of E^{2}STR.

	Input Size	Patch Size	Embedding	Depth	Heads	Parameters
Vision Encoder	32x128	4x4	768	12	12	85M
Language Decoder	-	-	768	12	12	125M

Table 7. Model details of E^{2}STR.

	MPSC	EIST	IAM
MAERec	81.81	70.33	70.27
MAERec†	82.00	70.77	70.51
E^{2}STR-ICL	83.64	76.77	74.10

Table 8. Word Accuracy performance comparison between MAERec [17] and E^{2}STR-ICL. MAERec† refers to MAERec using the same vision encoder and the same language decoder with E^{2}STR-ICL.

8. Model Scalability

Table 9 presents the inference time change brought by the different number of in-context prompts. It is easy to find that the number of in-context prompts in E^{2}STR is scalable. For example, the inference time of E^{2}STR-ICL (where we select two prompts) is 0.094s. But When expanding the number of in-context prompts by 7 times (*i.e.*, 16 prompts), the inference time is only increased by 1.08 times (*i.e.*, 0.196s).

Prompts	0	1	2	4	8	16
Inference Time (s)	0.071	0.085	0.094	0.113	0.140	0.196

Table 9. Inference time change brought by the different number of in-context prompts.

Table 10 presents the inference time change brought by different sizes of the in-context pool. As we can see, when expanding the pool size by 4 times (*i.e.*, from 100 to 500), the inference time is only increased by 0.07 times (*i.e.*, from 0.094 to 0.101). As a result, our E^{2}STR-ICL is highly scalable in terms of both in-context pool size and the number of in-context prompts.

Pool Size	100	200	300	400	500
Inference Time (s)	0.094	0.096	0.097	0.099	0.101

Table 10. Inference time change brought by different sizes of the in-context pool.

	Prompt Domain			
	Non-context	MPSC	EIST	IAM
MPSC	81.26	**83.64**	83.00	82.96
EIST	69.66	70.30	**76.77**	70.00
IAM	69.51	72.17	71.70	**74.10**

Table 11. Performance change brought by the domain variation of the in-context pool. **Bold** values denote the best performance in a row.

9. Model Stability

Table 11 presents how the performance change when varying the domains of the in-context pool. As we can see, our E^{2}STR-ICL is stable to the change of the context prompts. On all three benchmarks, out-of-domain in-context pools still improve the performance, though the improvement is lower than in-domain in-context pools. Nevertheless, there still exists a very slim chance that E^{2}STR-ICL erroneously rectifies predictions due to misleading prompts. Shown in Figure 10, when certain areas of the prompt image is highly

	Training GPU Hours		MPSC	EIST	IAM	AVG
kNN	**415.6**	base	81.22	69.78	69.62	73.54
		ICL	82.06	70.95	71.00	**74.67**
ST	**131.2**	base	81.26	69.66	69.51	73.48
		ICL	83.64	76.77	74.10	**78.17**

Table 12. Comparisons between kNN and our ST-strategy during in-context training.

similar to the test image but the ground-truth is different, E^{2}STR-ICL may erroneously rectifies the prediction.

Figure 10. Examples of erroneous rectification brought by misleading prompts.

10. Visualization

We provide more examples of the cross attention visualization in Figure 11.

Figure 11. More examples of the cross attention visualization.