Supplementary material for
Towards the Unseen: Iterative Text Recognition by Distilling from Errors

Ayan Kumar Bhunia[1] Pinaki Nath Chowdhury[1,2] Aneeshan Sain[1,2] Yi-Zhe Song[1,2]
[1]SketchX, CVSSP, University of Surrey, United Kingdom.
[2]iFlyTek-Surrey Joint Research Centre on Artificial Intelligence.
{a.bhunia, p.chowdhury, a.sain, y.song}@surrey.ac.uk.

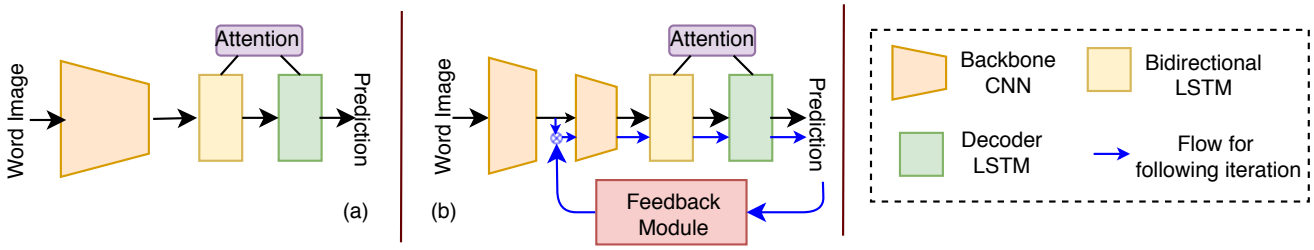

Figure 1. (a) State-of-the-art text recognition framework [16] (b) Character sequence prediction designed through iterative feedback network, that fuses previously predicted character sequences with rich visual features for subsequent prediction.

A. Relative Performance Gain

We introduce a meta-framework that could be incorporated on the top of most of the SOTA text recognition frameworks. Along with seminal work like **ASTER** by Shi *et al*. [16], we consider some latest text recognition frameworks like **Show, Attend and Read** (SAR) [12] and **SCATTER** [14] as our baseline, upon which we validate our iterative framework. SAR extends the work of Shi *et al*. using 2D attention mechanism to eliminate the rectification network. On the other side, SCATTER additionally couples multiple BLSTM encoders for richer context modelling, followed by a gating mechanism to balance between context rich information and backbone CNN features. Here, we additionally add the results using lexicons for different datasets under both for DS and CS setups. Furthermore, we add graphs to focus on by what margin our framework can provide gain over respective baseline SOTA frameworks under both DS and CS setups. Moreover, recognising rarely/unseen words being the major focus of this work, it can seen clearly seen that we achieve a rather significant performance gain on both STR (5-7%) and HTR (10-12%) tasks under DS set up. Nevertheless, we obtain a reasonable gain under CS set up unanimously over all the studied baseline SOTA frameworks.

B. Feedback Module vs Language Model

Instead of the iterative approach towards refining the text prediction, we could have used a Language model (LM) over the model's prediction. For a fair comparison, we use a state-of-the-art RNN-LM [7] trained from text corpus (librispeech) at character level [1] that aims to predict the next likely character. This could be fused with the text recognition decoder using two popular state-of-the-art methods introduced in [6] that usually integrate external LM for machine-translation and speech recognition tasks. In *Shallow Fusion*, weighted sum of predicted scores from text recognition decoder and LM are used for final prediction. *Deep Fusion* on the other hand fuses the hidden states of those two together followed by a FC layer. Please refer to [6] for more details. As seen from Table 3, our method performs better in both CS and DS setups in comparison to these LM integrations. Integrating external Language Model for discrete word recognition is a separate direction of research altogether. Even if it is incorporated as an off-the-shelf choice, performance is limited, as claimed in a recent independent

Table 1. Comparison of unconstrained WRA for **novel words** not encountered during training (**DS setup**). $t = 0$ signifies no feedback.

Methods	IIIT5K			SVT	IC13	IC15	SVTP	CUTE80	IAM		RIMES	
	50	1K	None	None	None	None	None	None	L	0	L	0
Shi *et al.* [16] (t=0) No-Feedback	97.4	93.4	84.3	84.2	82.6	65.7	74.4	61.6	71.7	54.3	73.4	59.7
Baseline Seq-SCM	97.6	93.7	85.6	84.1	83.7	65.5	75.8	63.4	74.5	57.6	77.9	63.7
Baseline Deterministic-Feedback	97.7	94.4	87.9	86.8	85.9	70.4	78.6	64.7	76.0	59.9	80.3	69.7
Shi *et al.* [16] **+ CVAE-Feedback** (t=1)	97.8	96.2	90.6	88.7	89.3	72.2	79.6	65.1	78.3	64.5	83.7	70.4
Shi *et al.* [16] **+ CVAE-Feedback** (t=2)	97.9	96.8	**90.8**	**88.9**	**89.4**	**72.6**	**79.6**	**66.1**	78.4	**64.8**	83.9	**70.5**
Shi *et al.* [16] **+ CVAE-Feedback** (t=3)	97.8	96.4	90.7	88.8	89.5	72.5	79.6	65.8	78.4	64.6	83.6	70.3
Relative Gain (t=0 vs t=2)	0.5↑	3.4↑	6.5↑	4.7↑	6.8↑	6.9↑	5.2↑	4.5↑	6.7↑	10.5↑	10.5↑	10.8 ↑
Show, Attend and Read [12] (t=0) No-Feedback	97.7	93.8	85.8	86.5	84.7	68.4	82.2	71.8	74.9	57.9	77.3	62.8
Show, Attend and Read [12] **+ CVAE-Feedback** (t=2)	98.1	96.9	**91.5**	**90.5**	**91.2**	**74.8**	**87.1**	**75.0**	81.1	**68.0**	84.9	**73.0**
1pt **Relative Gain** (t=0 vs t=2)	0.4↑	3.1↑	5.7↑	4.0↑	6.5↑	6.4↑	4.9↑	3.2↑	6.2↑	10.1↑	7.6↑	10.2 ↑
SCATTER [14] (t=0) No-Feedback	97.6	93.5	84.7	86.9	84.3	71.8	82.6	69.3	75.6	59.0	77.4	62.9
SCATTER [14] **+ CVAE-Feedback** (t=2)	98.0	96.8	**91.1**	**90.9**	**90.9**	**77.7**	**87.3**	**72.7**	81.5	**68.7**	85.0	**73.1**
1pt **Relative Gain** (t=0 vs t=2)	0.4↑	3.3↑	6.4↑	4.0↑	6.6↑	5.9↑	4.7↑	3.4↑	5.9↑	9.7↑	7.6↑	10.2 ↑
1pt												

Table 2. Comparison of unconstrained WRA on standard evaluation protocol (**CS setup**). $t = 0$ signifies no feedback.

Methods	IIIT5K			SVT	IC13	IC15	SVTP	CUTE80	IAM		RIMES	
	50	1K	None	None	None	None	None	None	L	0	L	0
Shi *et al.* [16] (t=0) No-Feedback	99.3	98.5	93.2	93.1	91.6	75.9	78.2	79.3	91.2	82.3	93.5	88.9
Baseline Seq-SCM	99.3	98.5	93.2	93.0	91.8	75.8	78.5	79.9	91.8	82.9	93.7	89.3
Baseline Deterministic-Feedback	99.6	98.8	93.5	93.6	92.7	77.1	79.6	65.1	93.1	86.9	94.9	92.0
Shi *et al.* [16] **+ CVAE-Feedback** (t=1)	99.4	98.6	94.0	93.5	93.1	78.4	80.4	82.5	93.1	86.9	94.9	92.0
Shi *et al.* [16] **+ CVAE-Feedback** (t=2)	99.6	98.8	**94.9**	**93.7**	**93.7**	**78.8**	**80.9**	**82.9**	93.7	**87.5**	95.2	**92.7**
Shi *et al.* [16] **+ CVAE-Feedback** (t=3)	99.5	98.7	94.6	93.6	93.5	78.5	80.7	82.7	93.6	87.2	94.9	92.4
Relative Gain (t=0 vs t=2)	0.3↑	0.3↑	1.7↑	0.6↑	2.1↑	2.9↑	2.7↑	3.6↑	2.5↑	5.2↑	1.7↑	3.8 ↑
Show, Attend and Read [12] (t=0) No-Feedback	99.4	98.7	94.8	91.2	93.7	78.6	86.0	89.5	92.7	85.9	93.9	90.2
Show, Attend and Read [12] **+ CVAE-Feedback** (t=2)	99.6	98.9	**96.3**	**91.9**	**95.4**	**81.4**	**88.5**	**91.0**	94.3	**89.7**	95.2	**93.0**
1pt **Relative Gain** (t=0 vs t=2)	0.2↑	0.2↑	1.5↑	0.7↑	1.7↑	2.8↑	2.5↑	1.5↑	1.6↑	3.8↑	1.3↑	2.8↑
SCATTER [14] (t=0) No-Feedback	99.3	98.5	93.6	92.7	93.8	82.0	86.5	87.0	92.8	86.0	94.0	90.5
SCATTER [14] **+ CVAE-Feedback** (t=2)	99.6	98.8	**95.2**	**93.2**	**95.7**	**84.6**	**88.9**	**89.7**	94.6	**90.3**	95.6	**93.2**
1pt **Relative Gain** (t=0 vs t=2)	0.3↑	0.3↑	1.6↑	0.5↑	1.9↑	2.6↑	2.4↑	2.7↑	1.8↑	4.3↑	1.6↑	2.7↑
1pt												

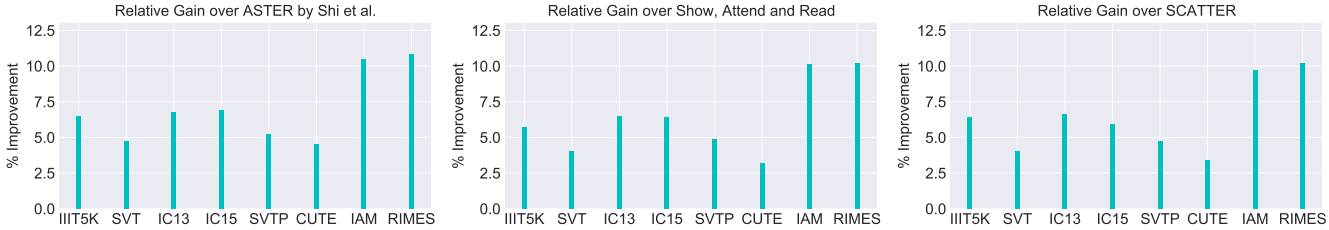

Figure 2. Performance gain in DS setup

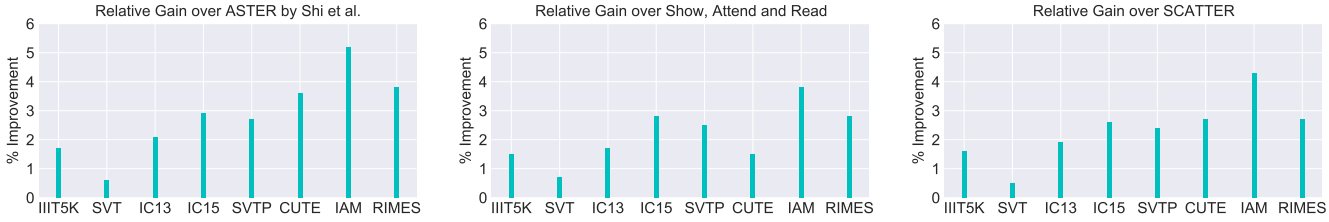

Figure 3. Performance gain in CS setup

work [11]. This can be attributed to the following factors: (i) LM has been used extensively in speech recognition tasks for refining a model's prediction, where data is present at the sentence-level, whereas our focus lies in discrete word recognition. While at sentence level there is enough context to refine word predictions using LM, it cannot harness similar extent of context information for discrete words. (ii) The language model corpus is significantly different from the one used for training word-

Table 3. Comparative study with different Language Model (LM) integration methods

Methods	Conventional Setup & Disjoint Setup															
	IIIT5K		SVT		IC13		IC15		SVTP		CUTE80		IAM		RIMES	
	CS	DS	CS	DS	CS	DS	CS	DS	CS	DS	CS	DS	CS	DS	CS	DS
Shi *et al.* [16]	93.2	84.3	93.1	84.2	91.6	82.6	75.9	65.7	78.2	74.4	79.3	61.6	82.36	54.4	88.9	59.7
[16] + Shallow	93.3	84.3	92.9	84.2	91.5	82.5	75.9	65.7	78.0	74.5	79.3	61.6	82.30	54.3-	88.7	59.7
[16] + Deep	93.5	85.6	93.3	85.3	92.3	85.1	76.5	67.4	78.9	76.5	81.2	62.9	83.67	57.5	89.9	63.6
[16] **+ CVAE-Feed.** (t=2)	94.9	90.8	93.7	88.9	93.7	89.4	78.8	72.6	80.9	79.6	82.9	66.1	87.5	64.8	92.7	70.5

image recognition system. This leads to a biased incorrectness [7]. (iii) LM being an independent post processing step, not only ignores rich visual features from the input image, but is also unaware of the error distribution of the model. On the contrary, our model revisits the rich visual features iteratively after every prediction, considering the error distribution while training. Furthermore to align with the evaluation standards for *unconstrained word recognition* we site all results in our work **using greedy decoding only – no LM based post-processing**. By greedy decoding, it means we only take the model's output without any post-processing as our final result. We conduct a further experiment to compare with the classic N-Gram LM model [11] for the CS (DS) setup, this gives 76.2 (66.5)& 82.43 (55.56) on IC15 & IAM respectively, which is again worse than ours.

C. Optimal Performance at t=2

In existing literature involving iterative pipelines, one simply stops when the gain diminishes, and is usually found empirically [3, 9, 22]. In our case, performance saturated at $t_{optimal} = 2$ (degraded by 0.0-0.3% at t=3). A similar phenomenon where performance degrades in later iterations is also reported in iterative pose-estimations [19] and image generation [9]. We speculate that this could be attributed to the randomness associated with the latent space of VAE, where the feedback module unknowingly adds noise to the convolutional feature maps.

D. Why is training with less unique words advantageous?

The disjoint training setup serves as an evaluation protocol for testing our model's accuracy on unseen words. The resulting superiority of our model in such scenario establishes a confidence of fair result over datasets running low in unique words. Consequently, any model can be trained by our algorithm using datasets having lesser unique words to deliver a satisfactorily high accuracy. This in turn alleviates the challenge of acquiring rarely available large datasets (apart from English) for training text recognition models. Furthermore, even with a small set of unique words, one can generate multiple instances of the same unique word by asking different people to write the same word, without incurring any additional annotation cost. For instance the word 'hello' written by 10 different users would provide 10 different word images, without any extra time cost on annotating them. This greatly simplifies the collection and annotation processes during dataset formation.

E. Trade-off between increased time cost *vs* additional performance gain

Any state-of-the-art iterative approaches found in the computer vision literature [19, 3, 9, 22, 13, 9, 4] do incur an extra computational expenses, be it text rectification [21] or in our case text-recognition. The consensus within the community is however that the extra computational expenses can be ignored w.r.t the additional performance gain – in our case, this would be a very significant gain of 5-7% and 10-12% under unseen scenarios in STR and HTR datasets respectively. Of course, one does need to carefully assess the extra computational burden – this is something we already did in Table 5, and in our case, the extra time cost is in the milliseconds which is inline with prior works.

F. Probabilistic model being better than deterministic one

One candidate word would have multiple possible erroneous alternatives. As knowledge from such error distributions needs to be distilled into the feedback module, uncertainty handling is very important yet lacking in any deterministic pipeline. Furthermore, the feedback module is in a cross-modal setting where information needs to be transferred from discrete character space to continuous affine transformation parameter space (Section 3.2) of conditioning layer. In such scenarios, variational models (probabilistic) have generally been proven to be more effective [17, 23, 24] because of the region estimation of latent space, as opposed to the point estimation used in deterministic methods. Consequently, we employ a CVAE to explicitly model the prior about possible error distribution which results in a better performance. We have compared our framework with a deterministic baseline (*Deterministic-Feedback*), and our probabilistic model outperforms its deterministic counterpart.

G. Effectiveness of feedback module

A feedback module is central to any iterative framework like ours – it completes the information flow from iteration $i-1$ to i. In our case, it propagates knowledge of predicted character sequence from an earlier iteration to the next. Our feedback mechanism is a novel conditional variational autoencoder, which is capable of distilling knowledge from error distributions.

Please note that the feedback module is trained from two data sources - (i) it learns correcting the model's prediction from iteration $i-1$ to i, (ii) Using an auxiliary decoder in the feedback module, we are trying to reconstruct the correct word (e.g. 'hello') from its erroneous alternative ('nello'). This error distribution (e.g. containing {'hello', 'nello'}) is pre-collected using SOTA methods that basically has the knowledge of which other ways the word 'hello' could be wrongly recognised as, based on its appearance. Thus it imparts a knowledge about the erroneous possible alternatives– such that the model is inclined to predict 'hello' instead of its erroneous alternatives (e.g. 'nello'). In other word, we encourage the model 'not to do such mistakes'.

Please refer to Section section 3.2 for detailed descriptions. We conducted a series of ablative experiments to verify the effectiveness of the feedback module in Section 4.3 & Table 4. We have mostly evaluated its effectiveness by (i) altering the designs of the autoencoder (Tab. 3), and removing the error distribution (Tab. 4), and (ii) comparing to a language model alternative.

H. Reason behind obtaining much better performance for unseen words

One needs to be careful to understand that the definition of "unseen" word is different from "unseen" as used in zero shot recognition. In particular, say for example one word `'kingdom'` had never been encountered by the model during training. Since any sequence-to-sequence learning model has a significant extent of vocabulary dependency, recognising unseen character sequence is comparatively difficult for the model than if the word had appeared during training. Furthermore, the word `'kingdom'` might be unknown, but not the individual characters `'k', 'i, 'n, 'g, 'd, 'o, 'm'`. The model has the knowledge of individual character from training. In other words, "unseen" words are the "difficult cases", but this is not analogous to the notion of "unseen class" of zero-shot learning. Rather our model learns the fine-grained character details better, and due to its iterative design along with knowledge gathered from error distribution – it can predict better via rectification for those "difficult cases".

I. Mechanism for 'unseen' words

In short, our model assumes words as character sequences, and it follows that "*unseen*" words are sequences that were not observed during training (while the *individual* character themselves would have). The iterative design coupled with knowledge distilled from error distributions, gives our model the best chance of finding the right combinations even if they are "unseen". In other words, through iterative look-back mechanism the model is encouraged to become less biased/overfitted on the trained character sequence. Instead it should rely on fine-grained character level details for correct prediction.

J. Where output of T_A should be modulated:

Interesting point! We do not have any explicit labels to modulate the feature-map, rather this is implicitly learned via back-propagating gradients based on loss (Eq. 4) computed at following (i+1) prediction. Similar ideas can be found in the rectification network of [16]

K. Comparison with RandText by Yue *et al*. [20]:

Please note that while unseen words in Yue *et al*. [20] consists of random character sequences with no context information, ours is more generic and realistic [18] i.e., consisting of unseen *plausible* words. Additionally our motivation is very different from [20]. While we try to rectify false predictions by re-visiting the visual feature and modelling textual error distribution, [20] balances the available contextual and positional information dynamically in a single pass. Neither the code or `RandText` dataset was unavailable.

L. Is gain due to extra computation?

We additionally validated this point by replacing the backbone feature extractor in [16, 12, 14] with ResNet-101 to match the Flops of their feedback counterpart. The resulting heavier variants reached $54.3\%(65.9\%)$ [16], $58.6\%(69.5\%)$ [12] and $59.7\%(72.3\%)$ [14] in DS setting for IAM(IC15) respectively. This proves that our performance gain was not tied to extra computations.

M. Design specific novelty of CVAE:

Unlike [17] and [23], ours is a (a) cross-modal VAE that transfers knowledge from discrete character space to continuous feature-map space, (b) modified design (L397-431) that enables learning from textual error distribution – both contributions being specifically designed for text recognition.

N. No leakage of information through error distribution

In the DS setup, SOTA [15, 5, 2, 16] methods (used to collect the error distribution) are **re-trained** using a subset of the training set. The subset is created by removing all words (having same character sequence) occurring in the evaluation set to ensure no "leak" of information under DS setup through error distribution. This ensures that the error distribution produced by other SOTA methods, which also do not see the words occurred in the evaluation set in the training phase.

O. Comparison with SOTA:

We follow the training setup in [16, 12, 14]. Accordingly, training set of SAR [12] consists of [8], [10], SynthAdd, and public real dataset. SCATTER [14] uses [8], [10] and SynthAdd. For fairness, we compare with any SOTA after adding our feedback module.

P. More insights about comparison with LM model trained on librispeech:

Replacing librispeech with words from word-image recognition datasets scored $83.2\%(76.3\%)$ and $54.7\%(66.2\%)$, using [16]+Deep Fusion in the CS and DS settings on IAM (IC15) dataset respectively. This drop is for the limited vocabulary of words in word-image datasets, when compared to librispeech (testing-words excluded), specially for DS setup. We already have a stronger LM competitor.

References

[1] End-to-end automatic speech recognition systems - pytorch implementation. https://github.com/Alexander-H-Liu/End-to-end-ASR-Pytorch. Accessed on: 04-03-2021. 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 *ICCV*, 2019. 5

[3] João Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *CVPR*, 2016. 3

[4] Xinlei Chen, Li-Jia Li, Li Fei-Fei, and Abhinav Gupta. Iterative visual reasoning beyond convolutions. In *CVPR*, 2018. 3

[5] Zhanzhan Cheng, Yangliu Xu, Fan Bai, Yi Niu, Shiliang Pu, and Shuigeng Zhou. Aon: Towards arbitrarily-oriented text recognition. In *CVPR*, 2018. 5

[6] Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. On using monolingual corpora in neural machine translation. *arXiv preprint arXiv:1503.03535*, 2015. 1

[7] Jinxi Guo, Tara N Sainath, and Ron J Weiss. A spelling correction model for end-to-end speech recognition. In *ICASSP*, 2019. 1, 3

[8] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *CVPR*, 2016. 5

[9] Minyoung Huh, Shao-Hua Sun, and Ning Zhang. Feedback adversarial learning: Spatial feedback for improving generative adversarial networks. In *CVPR*, 2019. 3

[10] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Reading text in the wild with convolutional neuralnetworks. *IJCV*, 2016. 5

[11] Lei Kang, Pau Riba, Mauricio Villegas, Alicia Fornés, and Marçal Rusiñol. Candidate fusion: Integrating language modelling into a sequence-to-sequence handwritten word recognition architecture. *arXiv preprint arXiv:1912.10308*, 2019. 2, 3

[12] Hui Li, Peng Wang, Chunhua Shen, and Guyu Zhang. Show, attend and read: A simple and strong baseline for irregular text recognition. In *AAAI*, 2019. 1, 2, 4, 5

[13] Ke Li, Bharath Hariharan, and Jitendra Malik. Iterative instance segmentation. In *CVPR*, 2016. 3

[14] Ron Litman, Oron Anschel, Shahar Tsiper, Roee Litman, Shai Mazor, and R Manmatha. Scatter: selective context attentional scene text recognizer. In *CVPR*, 2020. 1, 2, 4, 5

[15] Canjie Luo, Lianwen Jin, and Zenghui Sun. Moran: A multi-object rectified attention network for scene text recognition. *Pattern Recognition*, 90, 2019. 5

[16] Baoguang Shi, Mingkun Yang, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Aster: An attentional scene text recognizer with flexible rectification. *IEEE T-PAMI*, 2018. 1, 2, 3, 4, 5

[17] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *NeurIPS*, 2015. 3, 5

[18] Zhaoyi Wan, Jielei Zhang, Liang Zhang, Jiebo Luo, and Cong Yao. On vocabulary reliance in scene text recognition. In *CVPR*, 2020. 4

[19] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *CVPR*, 2016. 3

[20] Xiaoyu Yue, Zhanghui Kuang, Chenhao Lin, Hongbin Sun, and Wayne Zhang. Robustscanner: Dynamically enhancing positional clues for robust text recognition. In *ECCV*, 2020. 4

[21] Fangneng Zhan and Shijian Lu. Esir: End-to-end scene text recognition via iterative image rectification. In *CVPR*, 2019. 3

[22] Fangneng Zhan, Shijian Lu, and Chuhui Xue. Verisimilar image synthesis for accurate detection and recognition of texts in scenes. In *ECCV*, 2018. 3

[23] Tiancheng Zhao, Ran Zhao, and Maxine Eskénazi. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *ACL*, 2017. 3, 5

[24] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. Pluralistic image completion. In *CVPR*, 2019. 3