Towards the Unseen: Iterative Text Recognition by Distilling from Errors

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

Abstract

Visual text recognition is undoubtedly one of the most extensively researched topics in computer vision. Great progress has been made to date, with the latest models starting to focus on the more practical “in-the-wild” setting. However, a salient problem still hinders practical deployment – prior state-of-arts mostly struggle with recognising unseen (or rarely seen) character sequences. In this paper, we put forward a novel framework to specifically tackle this “unseen” problem. Our framework is iterative in nature, in that it utilises predicted knowledge of character sequences from a previous iteration, to augment the main network in improving the next prediction. Key to our success is a unique cross-modal variational autoencoder to act as a feedback module, which is trained with the presence of textual error distribution data. This module importantly translate a discrete predicted character space, to a continuous affine transformation parameter space used to condition the visual feature map at next iteration. Experiments on common datasets have shown competitive performance over state-of-the-arts under the conventional setting. Most importantly, under the new disjoint setup where train-test labels are mutually exclusive, ours offers the best performance thus showcasing the capability of generalising onto unseen words (Figure 1 offers a summary).

1. Introduction

Text recognition being a longstanding problem in computer vision plays a pivotal role in a diverse set of applications, ranging from OCR systems [4, 42, 48, 54], navigation and guiding board recognition [10], to more recent ones such as visual question answering [5]. With the advance of deep learning [50, 62, 10, 43], recognition accuracy have notably increased over traditional methods [36]. Research focus has thus shifted to the more practical “in-the-wild” setting in an attempt towards ubiquity. Of these, irregular scene text recognition [50, 10, 58, 60] has gained considerable attention, yet the focus is placed on irregular-image rectification process [62, 58] other than the core recognition problem itself.

In this paper, we continue this push towards practicality, albeit with a different perspective – we importantly focus on the understudied problem of unseen (or rarely seen) word recognition, where no (or limited) word image of a particular character sequence is present during training. Our motivation is straightforward – humans can recognise a word image, even when it falls beyond the scope of known vocabulary. In fact, robustness of a text recognition framework largely depends on its performance on rarely or unseen words [52]. Note that unlike the conventional zero-shot [59] setting where the transfer happens on class-level, here the combination of characters is “unseen” although the characters individually have come up in training. The fact that the sequences not being encountered during training is what makes this task challenging. Our solution for this “unseen” problem is intuitive: (i) we leverage an iterative framework with a feedback mechanism to give the model a chance to re-visit its false predictions, and (ii) we explicitly ask such feedback to encapsulate useful information that would help the model to correct itself at the following iteration.

Our first contribution is therefore an iterative framework, where characters predicted in the previous iteration provide clues through a feedback loop [17] to enhance performance in the subsequent iterations. This is fundamentally different to current state-of-the-arts [50, 10, 29], most of which adapt...
a feed-forward framework consisting of three-components (feature-extracting backbone, bidirectional-RNN encoder, and attention-based recurrent decoder). Despite the attention mechanism, its single-pass nature still dictates wrong predictions, thereby leaving no chance for the model to recover. To this end, our iterative design enables the revision of incorrect intermediate predictions in its subsequent steps, via a novel cross-modal (i.e., text prediction to image feature-maps) feedback mechanism. The key to our success lies with how feedback is progressed at each iteration. A naive solution might be to apply an independent spelling correction network [13, 56] chained serially to a basic text recognition model. Apart from not being end-to-end trainable, this also ignores the intermediate visual features from the recognition network, ultimately bisecting the feedback loop. We on the other hand advocate that earlier word predictions (text labels) should be fed back cross-modal to the main text recognition network and directly modulate the visual feature maps at the next iteration. That is, the feedback module triggers a mapping from the discrete predicted label space, to a continuous space of affine transformation parameters (akin to [41]) which are consequently used to condition visual features (hence closing the feedback loop).

Simply knowing how feedback works is not enough – we still need to devise what information should be fed back to give a model its best shot at rectifying itself. For this, we resort to distilling knowledge from textual error distributions collected from state-of-the-art text recognition models – this is akin to humans who use prior experiences to help them to make corrections. For example, ‘hello’ might be wrongly predicted as ‘nello’ or ‘bello’ due to partial structural similarity of ‘h’ with ‘n’ or ‘b’. By distilling such error distributions into the feedback module (during training only), the model will gain knowledge of correct character associations. Our second contribution is therefore designing the feedback module via a conditional variational auto-encoder (CVAE) [51] that learns from such error distributions. More specifically, we augment the vanilla CVAE with an auxiliary decoder that tries to directly reconstruct the correct word, given any incorrect prediction at each iteration. Note that deterministic alternatives such as typical feedback networks [47, 17] or spelling correction (prediction refining) networks, [13, 56] would not work well since they do not model the uncertainty among multiple erroneous alternatives, dictating a variational formulation like ours.

Our contributions are: [a] We for the first time propose an iterative approach to specifically tackle the “unseen” text recognition problem. [b] We design a conditional variational autoencoder to act as a feedback module, which works cross-modal to propagate predicted text labels from an earlier iteration to condition the visual features from the main network. [c] Our novel cross-modal feedback module is trained by distilling knowledge learned from textual error distributions that model multiple erroneous character sequences to a given candidate word.

Experiments confirm our framework to be capable of adopting unseen words better than state-of-the-art frameworks on various public scene-text recognition and handwriting recognition datasets. Further ablative studies demonstrate the superiority of our iterative framework over naive spell checking [56, 13] and language model alternatives [22], and that the proposed feedback module can be plug-and-play with more than one base network.

2. Related Works

Text Recognition: With the rising applicability of deep learning methods, Jaderberg et al. [20] employed convolution neural network for text recognition, but such methods were constrained to dictionary words. Although this limitation was eliminated by Jaderberg et al. [18], it still needed huge resources for character level localisation. Sequence discriminative training using connectionist temporal classification (CTC) [11] coupled with recurrent neural networks, dealt with the need for character level localisation. This lead to an end-to-end trainable convolutional recurrent neural network for reading texts [48]. This was further enhanced by incorporating the idea of attention [3] for text recognition [26, 49]. Usually a two-fold process is implemented [58, 49, 50, 62], where an irregular image is first passed through a rectification network, and then followed by a text recognition network. Ideas like 2D attention mechanism, focusing attention network (FAN) [9] has been explored recently along-with the possibility of enhancing text reading accuracy using synthetically generated large datasets [63]. Despite such extensive research, reading irregular curved texts had not been explored much in details until recently in [10], which describes how arbitrarily oriented texts can be read by extracting four directional features from the 2D input image. Baek et al. [2] has conducted a comparative study where different popular text recognition architectures were trained in a similar setting. Hence it can be observed that although recent works in text reading have emphasised on designing a better rectification network [62, 58], all such researches [31, 57] essentially used an off-the-shelf recognition module. On the other side, handwriting recognition poses a tougher challenge owing to a free flow nature [4] of writing. Poznanski et al. [42] employed a ConvNet to estimate an n-gram frequency profile along with a large dictionary having true frequency profiles for recognition.

Feedback Mechanism: Carreira et al. [7] augments input space based on a corrective signal output manifold, improving on human pose estimation and generalising to instance segmentation tasks [28]. While Wei et al. [55] used a ConvNet followed by a similar module having a larger receptive field, Newell et al. [37] developed an hour-glass network
design, stacked together for merging information across all input scales. Zamir et al. [61] proposed a novel network architecture aligned with a feedback notion, functionally akin to ResNet architecture. Others include instance segmentation [28], few-shot learning [64], object detection [8], super resolution [35] and image generation [47,17]. However, we here introduce cross-modal iterative feedback for text recognition with a conditional variational autoencoder that enables modelling of a prior knowledge of linguistically correct character-sequences.

Error Correction: Earlier efforts to utilise this idea of refining character-sequence prediction can be found in automatic speech recognition (ASR) community. In works such as Rozovskaya et al. [46] and Hans et al. [15] preposition errors were detected using classifiers. Grammatical error correction was approached using statistical machine translation methods by Ng et al. [38]. Recently, the idea of using recurrent encoder-decoder architecture in conjunction with attention mechanism was presented by Xie et al. [56] to address complex orthographic errors. Later this idea was adopted by Guo et al. [13] for tasks related to ASR.

3. Methodology

Instead of generating results in a single feed forward pass [50,30,58,65], we propose an iterative approach towards text recognition. As our work solely contributes to the text recognition part, we have refrained from detailing on the initial rectification network. We have used an off-the-shelf rectification network based on Spatial Transformer Network [21] and Thin Plate Splines [2] from Shi et al. [50].

3.1. Text Recognition Module

The rectified image received from the rectification network [50] is fed into this text recognition network $T$ aiming to produce a character sequence $Y = \{y_1, y_2, ..., y_K\}$, where $K$ denotes the variable length of text. Given an image $I \in \mathbb{R}^{H \times W \times C}$, the convolutional feature extractor tries to learn rich visual information and produce a feature map of size $\mathbb{R}^{H' \times W' \times D}$, where $H'$, $W'$, and $D$ are the height, width and number of channels in the output feature map respectively. That output is reshaped into a sequence of feature vectors $B = [b_1, b_2, ..., b_L]$, where $L = H' \times W'$, and $b_i \in \mathbb{R}^D$. Every $D$ dimensional feature vector $b_i$ encodes a particular local image region based on its receptive fields. Thereafter a bidirectional-LSTM is employed to capture the long range dependency in both the directions, thus alleviating the constraints of limited receptive fields. It outputs an updated feature sequence of same length, denoted by $H \in \{h_1, h_2, ..., h_L\}$. Following this $y_k$ is decoded based on three factors viz; bidirectional-encoder output $H$, the previous internal state $s_{k-1}$, and the character $y_{k-1}$ predicted in the last step. At time step $k$, a recurrent decoder network generates an output vector $o_k$ and a new state vector $s_k$, defined as: $(o_k, s_k) = RNN(s_{k-1}, [g_k, E(y_{k-1})])$ where $g_k$ is the glimpse vector [50] that encodes the information from specific relevant parts of the encoded feature $H$ to predict $y_k$; $E$ is an embedding layer, and $\cdot$ signifies a concatenation operation. Here, $g_k$ is computed as: $g_k = \sum_{i=0}^{L} \left( \frac{\exp(a_{k,i})}{\sum_{j=0}^{L} \exp(a_{k,j})} \right) h_i$ and, attention score $a_{k,i} = v^T \tanh(W_s s_{k-1} + W_a h_i + b_a)$, where $v, W_s, W_a, b_a$ are the learnable parameters. Finally the current step character $y_k$ is predicted by: $p(y_k) = \text{softmax}(W_o o_k + b_o)$ where $W_o$ and $b_o$ are trainable parameters.

3.2. Cross-Modal Variational Feedback

Overview: Unlike previous attempts of devising mostly deterministic feedback modules [17, 47, 61], we propose a Cross-Modal Variational Feedback network instilling benefits of variational autoencoder (VAE) [25], a powerful class of probabilistic models. Let us consider a text recognition network $T$ that has been dissected into two parts, namely $T_A$ and $T_B$ where the particular position of dissection is obtained empirically as described in section 4.3. The prior knowledge from the resultant discrete character space is modelled using a Feedback Network $F$ by predicting affine transformation parameters. Such parameters modulate the output activation map of $T_A$, as $\Psi = T_A(1) \in \mathbb{R}^{H \times W \times T'}$, via a designed feedback conditioning layer $\Phi$. In other words, for iteration step $t$, the feedback network takes the preceding iteration’s output $Y^{t-1}$ as its input and predicts the transformation parameters $\mathcal{P}'$ as its output. As a result it learns the mapping: $F: Y \rightarrow \mathcal{P}'$, such that feedback conditioning layer $\Phi$ modulates $\Psi$ based on $\mathcal{P}'$. This can be depicted as $\Psi' = \Phi(\Psi; \mathcal{P}')$ where $\Psi'$ being fed to $T_B$ predicts $Y'$ with higher precision.

Feedback Conditioning Layer: The primary objective of this layer is the propagation of prior knowledge from previous feed-forward pass prediction to $T$. This action couples the rich visual information extracted by $T_A$, with the prior feedback signal coming from earlier prediction. A noteworthy mention in this context would be the work by Perez et al. [41], where a general purpose conditioning layer FiLM has been designed based on simple feature-wise affine transformation operation for visual reasoning. Works involving visual-question-answering [41], image style transfer [16] and semantic image synthesis [39] have been seen to endorse a similar idea as well. Based on some prior or conditional information, the intermediate activation map $\Psi$ could be modulated as: $\Psi = \Psi \circ \gamma + \omega$, where $\gamma, \omega$ are global affine transformation parameters. They both have dimensions $\mathbb{R}^{T'}$, and are usually predicted by a network receiving the conditional information as input. In lieu of transforming each channel globally, we allow local transforma-
We assume that posterior of \( z \) depends on the actual ground-truth label \( \hat{Y} \) instead of \( \hat{P} \), so \( q_\psi(z|\hat{P}, \hat{Y}) \approx q_\psi(z|\hat{Y}, \hat{Y}) \) which effectively makes the latent space \( z \) aware of exact ground-truth character sequence \( \hat{Y} \).

(iii) Inspired by the auxiliary task approach towards improving primary task objective [66], we aim to decode the ground truth character sequence \( \hat{Y} \) directly from \( z \) by using an auxiliary character sequence decoder. Firstly, this approach tackles the problem of vanishing latent variable [6] and provides better gradient to regularize the learning of feedback module. Secondly, we discover a choice of training our feedback module (refer Figure 2) via relativistic information of prediction (ground-truth pairs) (say \( G \)) generated from other state-of-the-arts, instead of depending on \( \hat{T} \) predicted \( Y \) alone. This helps the module in learning an associative relation between candidate correct words and closely related erroneous instances. Loosely speaking ‘hello’ might be predicted as ‘nello’ or ‘bello’ due to a partial structural similarity of ‘h’ with ‘n’ or ‘b’. Learning this sense from error distribution imparts the model a semantic knowledge of required character association to form a valid word. Additionally, using text-only data \( G \) alleviates the issue of limited availability of image-paired datasets during training.

Hence, we adapt Eqn. 1 for our feedback network to generate a prior knowledge \( \hat{P}^* \), instead of transformation parameters only. This prior knowledge has two components. One encapsulates relationship between \( Y' \) and \( \hat{Y} \), while the other generates \( \hat{P} \) and injects \( Y \) into \( \hat{T} \) for next prediction. Therefore adopting the aforementioned variational lower bound expression (Eqn. 1), and assuming conditional independence of those two knowledge components (given \( z \) and \( Y \) denoted by \( p(\hat{P}^*|z, \hat{Y}) = p(\hat{P}|z, \hat{Y})p(\hat{Y}|z, \hat{Y}) \), we get a modified lower bound as:

\[
\tilde{\mathcal{L}}'(Y, \hat{P}^*; \theta, \psi) = -KL(q_\psi(z|\hat{Y}, \hat{Y})||p_\theta(z|Y)) + \\
E_{q_\psi}[\log p(\hat{P}|z, Y)] + E_{q_\psi}[\log p(\hat{Y}|z, Y)]
\]  

(ii) Ideally speaking, \( \hat{P} \) is supposed to adjust the activation map \( \hat{Y} \) in a way that predicts \( \hat{Y} \) when passed through a fixed \( \hat{T} \). In other words, for time-step \( t \), given \( \hat{P}^t \) predicted by the feedback network with a conditioning on \( Y^{t-1} \), the modulated feature map \( \hat{Y}^t \) is fed into \( \hat{T} \) to obtain the output of next iteration \( Y' \). Therefore, \( Y' = \hat{T} \hat{Y}^t \) (\( \hat{Y}^t = \hat{Y}(\hat{Y}; \hat{P}^t) \)). Thus, minimizing the cross-entropy loss between \( Y' \) and \( \hat{Y} \) is equivalent to maximizing the likelihood of \( \hat{P} \).

(i) Training of the network consists of 2 steps: (i) Predicting \( Y \) (green arrow); (ii) Refining \( Y \) to \( Y' \) using feedback mechanism. Feedback module excluding \( F_P \) can be trained using additional text-only data \( G \) (red). However, when trained using prediction \( Y' \), the entire path is followed (red+blue). Inference in the model consists of predicting \( Y' \) using any arbitrary number of correction steps.
Network Components: Ignoring the time-step notation, let us consider any word image $I$ having ground-truth $Y$, that undergoes the first forward-pass of $T$ predicting $Y$ and $Y'$ in current and successive post-feedback iterations respectively. During training, we obtain an embedding representation of both $Y$ and $Y'$ through a shared encoder network $F_{enc}$: $Y_{enc} = F_{enc}(Y)$ and $Y_{enc} = F_{enc}(Y')$. Moving on we have two independent branches: A posterior network $F_{post}$ to estimate parameters of posterior distribution, and a prior network $F_{prior}$ to do the same for prior distribution. Following this, we get: $\mu_{post}, \sigma_{post} = F_{post}(Y_{enc}, \tilde{Y}_{enc})$ and $\mu_{prior}, \sigma_{prior} = F_{prior}(Y_{enc})$. A latent variable $z$ is sampled from the posterior (or prior during testing) distribution and merged with $Y_{enc}$ before feeding it to the auxiliary ground-truth character sequence decoder $F_{aux}$ and transformation parameter prediction sub-module $F_P$. Therefore, we have, $P = F_P([Y_{enc}, z])$, and $\tilde{Y} = F_{aux}([Y_{enc}, z])$ where $P$ encapsulates both $\Gamma$ and $\Omega$, and $\tilde{Y}$ is output from the auxiliary decoder $F_{aux}$. Therefore, we can predict the next iteration as: $Y' = T_B(\tilde{Y})$, where $\Psi = \Phi(\Psi; P)$. Please refer to Figure 2 for clarity.

Learning Objectives: Our baseline text recognition network $T$ (parameters $\theta_T$) is trained using Cross-Entropy (ce) loss summed over the ground-truth output sequence $Y$

$$L^\theta_T = L_{ce}(Y, \hat{Y}) = -\sum_{k=1}^K y_k \log P(y_k | I, y_{k-1}) \quad (3)$$

Feedback module $F$ is to be trained from two input sources: (a) by using the prediction $Y$ obtained from $T$. In cases where the forward-pass prediction $Y$ is accurate the very first time, a diverged prediction value in the next iteration is highly undesirable. Hence, along with the lower bound (Eqn. 2), we impose a monotonically decreasing constraint $L_c$. This enforces the loss value (Eqn. 3) related to the current iteration $\{Y', \hat{Y}\}$ to be lesser than its previous $\{Y, \hat{Y}\}$, thus converging predictions to a higher precision. Hence, we optimize all the parameters $\theta_F$ of feedback module using:

$$L^{\theta_F} = \lambda_1 L_{ce}(Y', \hat{Y}) + \lambda_2 L_{ce}(\hat{Y}, \hat{Y}) + \lambda_3 L_{KL} + \lambda_4 L_c$$

where, $L_c = \max(0, L_{ce}(Y', \hat{Y}) - L_{ce}(Y, \hat{Y})) \quad (4)$

(b) By using pre-collected text-only data $G$ generated from existing text-recognition methods $[30, 10, 2, 50]$. Doing so develops a semantic sense of associative relation between candidate correct words and erroneous alternatives. Keeping $F_P$ fixed, rest of the feedback module (parameters denoted by $\theta_F$) is optimised solely using auxiliary decoder reconstruction loss and KL divergence loss as:

$$L^{\theta_F'} = \lambda_2 L_{ce}(\hat{Y}, \hat{Y}) + \lambda_3 L_{KL} \quad (5)$$

During testing we exclude both $F_{aux}$ and $F_{post}$, where iterative prediction is done using the transformation parameter predicted by $F_P$ (see Figure 2).

Comparative Discussions: While state-of-the-art text recognition frameworks model a conditional distribution $p(Y|I)$, our modelling objective is $p(Y'|Y_{t-1}, I)$. This could be decoupled into two marginal distributions related via a transformation parameter space $P'$ (more strictly prior knowledge $P'$). Assuming the obvious conditional independence, we reformulate it as: $p(Y'|Y_{t-1}, I) = p(Y'|I, P') \times p(D|Y_{t-1})$, where $p(Y'|I, P')$ is our modified feed forward text recognition model and $p(P'|Y_{t-1})$ resembles our feedback network.

Algorithm 1 Training algorithm of the proposed framework

1: Input: Image and ground-truth pairs $D$; prediction and ground-truth pairs $G$.
2: Initialise hyper params: $\alpha_1, \alpha_2$ be learning rate for $T, F$ respectively.
3: Initialise model params: $\theta_T, \theta_F$ ($\theta_F' \subset \theta_F$ excluding $F_P$).
4: while not done training do
5: Sample a mini-batch $D_t$ from $D$.
6: Update $\theta_T := \theta_T - \alpha_1 \nabla_{\theta_T} (L^{\theta_T})$ \ding{51} Eqn. 3.
7: Update $\theta_F := \theta_F - \alpha_2 \nabla_{\theta_F} (L^{\theta_F})$ \ding{51} Eqn. 4.
8: Sample a mini-batch $G_t$ from $G$.
9: Update $\theta_F' := \theta_F' - \alpha_2 \nabla_{\theta_F'} (L^{\theta_F'})$ \ding{51} Eqn. 5.
10: end while
11: Output: $\theta_T, \theta_F$.

4. Experiments

Datasets: We train our method using an approach similar to $[62, 58, 2, 10, 50, 30]$, on synthetic datasets such as Synth90k $[19]$ and SynthText $[14]$ that hold 8 and 6 million images respectively. Evaluation is done on: IIIT5K-Words, Street View Text (SVT), SVT-Perspective (SVP), ICDAR 2013 (IC13), ICDAR 2015 (IC15), and CUTE80. IIIT5K-Words $[34]$ presents randomly picked 3000 cropped word images. Street View Text $[53]$ contains 647 images, mostly being blurred, noisy or having low resolution. SVT-Perspective $[44]$ offers 645 samples from side-view angle snapshots having perspective distortion. ICDAR2013 $[24]$ presents 1015 words while ICDAR2015 $[23]$ has 2077 images, 200 of which are irregular. CUTE80 $[45]$ distinguishes itself by presenting 288 cropped high quality curved text images. Handwritten Text Recognition (HTR) and Scene-Text Recognition (STR) both share a common objective in terms of recognition, which is usually handled by similar network architecture. Hence, we validate our results on two different public HTR datasets. The evaluation setup described in $[4]$ is employed on two large standard datasets viz, IAM $[33]$ containing 1,153,320 words and RIMES having 66,982 words. For IAM we use the same partition for training, validation and testing as provided. For RIMES, we follow the partition released by ICDAR 2011 competition.
Table 1. Comparison of unconstrained WRA for novel words not encountered during training. \( t = 0 \) signifies no feedback.

| Methods                            | HIT@5K | SVT  | IAM  | TIMES |
|------------------------------------|--------|------|------|-------|
| Shi et al. [50] (t=0) No-Feedback  | 84.1   | 84.2 | 82.6 | 86.5  |
| Baseline Seq-SCM                    | 85.6   | 84.1 | 83.7 | 65.5  |
| Baseline Deterministic-Feedback     | 87.9   | 86.8 | 85.9 | 75.8  |
| Shi et al. [50] + CVAE-Feedback (t=1) | 90.6 | 88.7 | 89.3 | 72.2  |
| Shi et al. [50] + CVAE-Feedback (t=2) | 90.8 | 88.9 | 89.4 | 72.6  |
| Shi et al. [50] + CVAE-Feedback (t=3) | 90.7 | 88.8 | 89.4 | 72.5  |

| Relative Gain (t=0 vs t=2)         |        |      |      |       |
|------------------------------------|--------|------|------|-------|
| Snow, Attend and Read [27] (t=0) No-Feedback | 5.71 | 4.01 | 6.57 | 6.47 |

4.1. Implementation Details

Network Design: Keeping text rectification and recognition networks similar to Shi et al. [50], we implement our framework using open sourced codes [2, 30] in PyTorch [40]. A bi-directional LSTM of hidden size 256 is used in designing \( F_{\text{enc}} \) of our feedback module which accepts a one layer MLP embedded 128-dimensional representation of discrete character sequences. For both posterior \( F_{\text{post}} \) and prior \( F_{\text{prior}} \) networks, we use 2 layer MLPs with \( \tanh \) non-linearity. The latent variable \( z \) has a size of 256. The auxiliary decoder \( F_{\text{aux}} \) is a one layer LSTM decoder whose initial hidden state is initialized by applying a FC-layer on the concatenated representation of \( Y_{\text{enc}} \) and sampled \( z \). The parameter prediction network \( F_{\text{p}} \) is a convolutional decoder network inspired from [67]. The first layer is a fully-connected layer implemented through a 1x1 convolution that maps to a tensor of size equal to the last CNN layer of \( T \) via a reshaping operation. Thereafter we introduce a sequence of residual decoder blocks which upsamples feature maps to higher spatial dimensions in the reversed order of down-sampling followed by \( T \). In other words, if a ResNet encoder block in \( T \) halves the height of feature map, the corresponding ResNet decoder block [67] in \( F_{\text{p}} \) will double it. This strategy essentially formulates intermediate feature maps of a decoder ResNet block \( F_{\text{p}} \), whose spatial size is similar to the corresponding layer in the encoder ResNet block \( T \), thus predicting required affine transformation parameters of similar dimensions as well. Since we need to predict both \( \Gamma \) and \( \Omega \), we double the number of convolutional filters in the last layer of \( F_{\text{p}} \) and split the output channel-wise to obtain \( \Gamma \) and \( \Omega \) individually.

Training Details: It has been observed in practice, that warming up individual components in the initial phase, followed by a joint training operation provides better stability than training the entire framework at one go. Text recognition network \( T \), along with rectification network, is trained using ADADELTA optimizer having a learning rate of 1.0 and a batch size of 64. Meanwhile Feedback module \( F \) is trained via Adam optimizer with a learning rate 0.001, and gradient clipping at 5. In the warm-up phase, firstly, text recognition and rectification networks are trained from a union of MJSynth and SynthText datasets, for 600K iterations. The rectification network is frozen thereafter. Then, to capture a linguistic prior from text-data, the feedback module \( F \) is trained independently from \( G \) for 300K iterations, thereby ignoring the \( F_{\text{p}} \) part (Eqn. 5). Finally, keeping \( T \) fixed, we train the complete feedback module using \( Y \) (Eqn. 4), with the same training specifications for 300K iterations. Now, for the joint training (see algorithm 1), after updating \( T \), \( F \) is updated with the prediction \( Y \) from \( T \), as input. Thereafter, keeping \( F_{\text{p}} \) fixed, \( F \) is updated using \( G \). During this training, we reduce the learning rate of \( T \) to 0.01 which continues for 600K iterations. Owing to corresponding data sizes in HTR, iterations for warm-up phases are 100K, 50K and 50K respectively, while for joint training it is 100K. For both STR and HTR, we resize the image to 32x100 and train our model in a 11 GB Nvidia RTX 2080-Ti GPU. \( \lambda_1 \), \( \lambda_2 \), \( \lambda_3 \) and \( \lambda_4 \) has been assigned the values of 1, 1, 50 and 0.5 respectively for the same purpose. In order to generate the text-only data \( G \), we follow an approach similar to [68]. The entire training dataset is divided into train-validation split in a cross validation setting, where top ten beam search decoded hypotheses are collected from different state-of-the-art (SOTA) models [30,10,2,50], to harness (using open-source code) the error information from closest possible erroneous alternatives. Such a collection may consist of correctly or incorrectly predicted words. This requires the feedback module to learn a language prior for fixing a wrongly predicted word, as well as for a one-to-one character-sequence mapping for correctly predicted words.

4.2. Performance Analysis

Baselines: Aligned to our iterative approach we explore two alternatives as baselines. Seq-SCN: Inspired from Automatic Speech Recognition community [56], a naive baseline could be designed where we train an independent Spelling Correction Module (SCM) based on sequence-to-sequence architecture from text-only training data [13] consisting of paired model-hypothesis and corresponding ground-truth. Deterministic-Feedback: Here, we simply replace the CVAE based feedback module by a deterministic encoder(bi-LSTM)-decoder (parameter prediction network) architecture along with Shi et al. [50].
**New Evaluation Setup:** Here we design a new disjoint train-test split (DS) in addition to conventional train-test split (CS). While training we remove all words from MJSynth and Synth90K whose truth-pairs appear in any of the mentioned STR testing datasets. Additionally we ensure, $G$ does not contain any information from the testing set. Due to size constraint in STR datasets, we split it such that all word-image pairs corresponding to one particular ground-truth character sequence must collectively fall in either one of training or testing set, thus ensuring disjointedness. This evaluation protocol grades our model’s recognition performance on word images whose ground-truth character sequences never appeared in the training dataset. We use it to verify generalising capability of our model for unseen or rarely seen words. Furthermore, superiority of our model in this scenario confirms a fair result on datasets having less unique words. As large datasets are rarely available for text recognition (specifically handwriting) apart from English, it is ideal to use a dataset created by collecting multiple instances of a small set of unique words which thus need to be annotated just once per unique word. To do so any model needs to learn character-specific fine-grained details from a small available set of unique words, and generalise onto other unseen character sequences. Please note that we re-train SOTA models [30, 10, 2, 50], used to collect the error distribution, by ensuring no words from evaluation set appear in their training (ensuring no “leak” of information), while collecting error distribution for DS setup.

**Result Analysis:** Along with the designed baselines, we incorporate our iterative design on top of three popular state-of-the-art (SOTA) feed-forward text recognition frameworks — a) Shi et al. [50], b) Show, Attend and Read [27] c) SCATTER [29]. Show, Attend and Read [27] extends [50] by including 2D attention and SCATTER [29] couples multiple BLSTM layers for richer context modelling. We follow similar training protocol in [50, 27, 29] respectively. Nevertheless, ours is a meta-framework and could be added on top of most SOTA frameworks. Table 1 (highest scores are in red) depicts unconstrained word recognition accuracy (WRA) on unseen words (DS setup).

**Comparison with Baselines:** Seq-SCM performs inferior to our method as the rich visual feature is not harnessed while refining the prediction. At times it fails to copy an already accurate prediction, leading to a lower accuracy in certain datasets. Deterministic-Feedback being an iterative framework, performs better than other baselines at $t = 2$, however, it lags behind our design since any uncertainty handling potential is absent. [ii] **Significant improvement under DS setup:** From Table 1, improvement due to our iterative pipeline is quite evident in the DS scenario over three SOTA baselines [50, 27, 29]. Similarly, in handwritten dataset (Table 1), the performance drop in DS setting is way more severe than its STR counterpart. This signifies that HTR poses a greater challenge due to its free-flow nature of writing. Improvement against Shi et al. [50] reaches to 10.5% and 10.8%, without lexicon information (unconstrained) in HTR dataset of IAM and RIMES respectively, while for STR dataset of IC15 it secures a 6.9% rise in DS setup.

**Additional Observations:** Improvement at $t=2$ w.r.t $t=0$ is shown as relative gain where our method outperforms fairly in CS setup, and largely in DS setup (Table 2). In fact optimal WRA is seen at iteration $t = 2$, which then diminishes. Contrary to a possible impression, that our feedback module might remember the erroneous pairings from $G$, the improvement on “unseen” words empirically validates against it. Figure 3 shows qualitative results.

### Table 2. Comparison with SOTA results using standard setup [2].

| Methods               | HIT5K | IC13 | IC15 | SVTP | CUTE80 | IAM |
|-----------------------|-------|------|------|------|--------|-----|
| Yang et al. [38]      | 94.4  | 93.9 | 78.7 | 80.8 | 87.5   | -   |
| Luo et al. [10]       | 91.2  | 92.4 | 68.8 | 76.1 | 77.4   | 82.1|
| Cheng et al. [10]     | 91.7  | 92.3 | 68.2 | 73.0 | 76.6   | -   |
| Zhang et al. [65]     | 83.8  | -    | -    | -    | -      | -   |
| Back et al. [2]       | 87.9  | 92.3 | 71.8 | 79.2 | 74.0   | -   |
| Lyu et al. [32]       | 94.0  | 92.7 | 76.3 | 82.3 | 86.8   | -   |
| Zhan et al. [62]      | 93.3  | -    | 76.9 | 79.6 | 83.3   | -   |
| Shi et al. [9]        | 81.9  | -    | -    | -    | -      | 80.3|
| Cheng et al. [9]      | 87.4  | 93.3 | 70.6 | -    | -      | -   |
| @ Shi et al. [50]     | 93.4  | 91.8 | 76.1 | 78.5 | 79.5   | -   |
| @ Li et al. [27]      | 95.0  | 94.0 | 78.8 | 86.4 | 89.6   | -   |
| @ Liman et al. [39]   | 93.7  | 93.9 | 82.2 | 86.9 | 87.5   | -   |
| Shi et al. [50] (t=0) | 93.2  | 91.6 | 75.9 | 78.2 | 79.3   | 82.3|
| Li et al. [27] (t=0)  | 94.8  | 93.7 | 78.6 | 86.0 | 89.5   | 85.9|
| Liman et al. [25] (t=0)| 93.6  | 93.8 | 82.0 | 86.5 | 87.0   | 86.0|
| Baseline Seq-SCM      | 94.8  | 91.8 | 75.8 | 78.5 | 79.9   | 82.9|
| Deterministic-Feed.   | 93.5  | 92.7 | 77.1 | 79.6 | 80.5   | 85.6|
| [50] + CVAE-Feed. (t=2)| 94.9  | 93.7 | 78.8 | 80.9 | 82.9   | 87.5|
| [29] + CVAE-Feed. (t=2)| 96.3  | 95.4 | 81.4 | 88.5 | 91.0   | 89.7|

**4.3. Further Analysis and Insights**

**Ablation Study:** We have done a thorough ablation study to justify the contribution of every design choice on both
IAM (HTR) and IC15 (STR) datasets using Shi et al. [50] as baseline. [i] Significance of auxiliary decoder: On removing this part along with its respective loss function from Eqn. 4 we see a performance drop of 2.19%(1.98%) and 5.01%(2.94%) in the CS and DS setting for IAM (IC15) dataset respectively, thus confirming its contribution. [ii] Significance of monotonically decreasing constraint: Removing it, destabilizes by 5.47%(3.87%) in DS setup, for IAM (IC15) datasets respectively, thus confirming its importance. [iii] Significance of using error distribution: Discarding data inclusion from $G$ (error distribution), leads to a performance drop of 2.17%(1.87%) in conventional train-test split, and a further drop of 4.9%(2.48%) in disjoint setup, for IAM (IC15) datasets. Please refer to Table 4 for more analysis. [iv] Finding optimal block for feedback: We evaluated the performance by providing a feedback signal into every ResNet block of backbone feature extractor of $T$ at a time. Table 3 shows a complete analysis on both IAM and IC15 datasets in CS and DS setups which shows the optimum result to be obtained by supplying feedback signal to Block-3 of ResNet convolutional architecture. Moreover, local transformation is seen to outperform the global one. [v] Performance with varying text length: It is often observed that any text recognition framework struggles to recognize lengthy words. Owing to its iterative refining approach, along with a modelled linguistic prior, our method shows a considerably higher performance in comparison to no-feedback baseline on increasing character sequence length, as shown in Figure 3. [vi] Computational cost: Finally, we want to notify that benefit of any iterative pipeline [7, 8, 28] does incur extra computational expenses, be it text rectification [62] or in our case text-recognition. A thorough study on complexity and speed analysis (Intel(R) Xeon(R) W-2123 CPU @ 3.60GHz) in Table 5 reveals that $T_g$ takes most time due to its sequential decoding operation, whereas $F$ adds minimal burden. Please refer to supplementary as well.

Table 3. WRA of using feedback after a specific ResNet Block (abbreviated as ‘Blk’) such as Block 1 (16 × 50 × 64), Block 2 (8 × 25 × 128), Block 3 (4 × 25 × 256), Block 4 (2 × 25 × 256), Block 5 (1 × 25 × 256) as described in ASTER [50].

| Methods | Conventional Setup (CS) & Disjoint Setup (DS) |
|---------|-----------------------------------------------|
|          | Blk_1 | Blk_2 | Blk_3 | Blk_4 | Blk_5 |
| CS       |       |       |       |       |       |
| DS       |       |       |       |       |       |

Feedback module vs Language Model: We could have substituted our iterative approach towards refining text prediction with a Language model (LM). For fairness, we use a state-of-the-art RNN-LM [13] trained from text corpus (librispeech) at character level [1] that aims to predict the next likely character. This could be fused with the recognition decoder using two state-of-the-art methods introduced in [12] via Shallow Fusion (weighted sum of predicted scores) and Deep Fusion (fusing their hidden states). Our method performs better in both CS and DS setups in comparison to these LM integrations (Table 6). The limited performance of off-the-shelf LM [22] can be attributed to: (i) LM is mostly used in speech recognition tasks, where data is present at sentence-level which provides enough context. For distract word recognition (our focus) however, LM cannot harness such extent of context information. (ii) The LM corpus is significantly different from that used for training word-image recognition system. This leads to a biased incorrectness [13]. (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 cite all results in our work using greedy decoding only – no LM based post-processing.

Table 5. Complexity and speed analysis against no. of parameters and flops (Multiply-Add), for both individual component (left) and varying no. of iterations (right), using CPU time during inference.

| Networks | Parameters | Multiply-Add | CPU | Iteration | CPU |
|----------|------------|--------------|-----|-----------|-----|
| $T_g$    | 3.4M       | 1461M        | 8.23ms | t=0       | 28.64ms |
| $T_g$    | 23.7M      | 1482M        | 12.82ms | t=1       | 43.60ms |
| F        | 24.4M      | 1897M        | 2.19ms | t=2       | 58.66ms |

Table 6. Comparison with different LM integration methods.

| Methods | Conventional Setup & Disjoint Setup |
|---------|------------------------------------|
|          | CS | DS | CS | DS | CS | DS | CS | DS | CS | DS |
|          | IAM | Rimes | IAM | CUTE95 | CUTE96 | IAM | Rimes |
| [50]     | 84.7 | 69.9 | 85.5 | 60.8 | 86.7 | 61.9 | 86.2 | 60.8 | 84.5 | 58.3 |
| [40]     | 84.0 | 87.6 | 86.7 | 64.2 | 87.5 | 64.8 | 87.3 | 63.9 | 85.6 | 63.2 |
| Global   | 97.2 | 91.5 | 79.9 | 65.7 | 79.3 | 61.6 | 82.36 | 54.4 | 88.9 | 55.7 |
| Local    | 93.3 | 84.3 | 75.9 | 65.7 | 79.3 | 61.6 | 82.30 | 54.3 | 88.7 | 59.7 |
| [10] + Deep | 91.5 | 85.6 | 76.5 | 64.7 | 81.2 | 62.9 | 83.67 | 57.5 | 89.9 | 63.6 |
| [40] + CV-AE-Feed (t=2) | 94.9 | 90.8 | 78.8 | 72.6 | 82.9 | 66.1 | 87.5 | 64.8 | 92.7 | 70.5 |

5. Conclusion

Here we have proposed a novel iterative approach towards text recognition. Using a conditional variational autocoder (CVAE) as a feedback module, the knowledge of predicted character sequences is passed from the previous iteration, into the main recognition network, improving subsequent predictions. Our feedback network learns to use the error distribution among multiple character sequences that are closely related to a candidate word. Experiments on various STR and HTR datasets show our network to outperform others on the conventional setting, and more significantly on the more practical disjoint (unseen) setting.
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. 8

[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. 2, 3, 5, 6, 7

[3] Zhanzhan Cheng, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In ICLR, 2015. 2

[4] Ayan Kumar Bhunia, Abhirup Das, Ankan Kumar Bhunia, Perla Sai Raj Kishore, and Partha Pratim Roy. Handwriting recognition in low-resource scripts using adversarial learning. In CVPR, 2019. 1, 2, 5

[5] Ali Furkan Biten, Ruben Tito, Andres Mafla, Lluis Gomez, Marçal Rusiñol, Ernest Valveny, CV Jawahar, and Dimosthenis Karatzas. Scene text visual question answering. In CVPR, 2019. 1

[6] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In CoNLL, 2016. 4

[7] João Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In CVPR, 2016. 2, 8

[8] Xinlei Chen, Li-Jia Li, Li Fei-Fei, and Abhinav Gupta. Iterative visual reasoning beyond convolutions. In CVPR, 2018. 3, 8

[9] Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, and Shuigeng Zhou. Focusing attention: Towards accurate text recognition in natural images. In ICCV, 2017. 2, 7

[10] Zhanzhan Cheng, Yangliu Xu, Fan Bai, Yi Niu, Shiliang Pu, and Shuigeng Zhou. Aon: Towards arbitrarily-oriented text recognition. In CVPR, 2018. 1, 2, 5, 6, 7

[11] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In ICML, 2006. 2

[12] Cuglar 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. 8

[13] Jinxi Guo, Tara N Sainath, and Ron J Weiss. A spelling correction model for end-to-end speech recognition. In ICASSP, 2019. 2, 3, 6, 8

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

[15] Na-Rae Han, Martin Chodorow, and Claudia Leakcock. Detecting errors in english article usage by non-native speakers. Natural Language Engineering, 2006. 3

[16] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In ICCV, 2017. 3

[17] Minyoung Huh, Shao-Hua Sun, and Ning Zhang. Feedback adversarial learning: Spatial feedback for improving generative adversarial networks. In CVPR, 2019. 1, 2, 3

[18] Max Jaderberg, Andrea Simonyan, Karen Vidaldi, and Andrew Zisserman. Deep structured output learning for unconstrained text recognition. In ICLR, 2015. 2

[19] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. In NeurIPS Deep Learning Workshop, 2014. 5

[20] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Reading text in the wild with convolutional neural networks. IJCV, 2016. 2

[21] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and koray kavukcuoglu. Spatial transformer networks. In NeurIPS, 2015. 3

[22] 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, 8

[23] 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 ICDAR, 2015. 5

[24] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Motas, Jon Almazan Almazan, and Lluis Pere De Las Heras. Icdar 2013 robust reading competition. In ICDAR, 2013. 5

[25] Diederik P. Kingma and Max Welling. Auto-encoding variational bayses. In ICLR, 2014. 3

[26] Chen-Yu Lee and Simon Osindero. Recursive recurrent nets with attention modeling for OCR in the wild. In CVPR, 2016. 2

[27] 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. 6, 7

[28] Ke Li, Bharath Hariharan, and Jitendra Malik. Iterative instance segmentation. In CVPR, 2016. 2, 3, 8

[29] Ron Litman, Oron Anschel, Shahar Tsiper, Roee Litman, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, et al. Icdar 2015 competition on robust reading. In ICDAR, 2015. 3

[30] Canjie Luo, Lianwen Jin, and Zenghui Sun. Moran: A multi-scale approach to scene text detection. In CVPR, 2016. 2

[31] Canjie Luo, Yuanzhi Zhu, Lianwen Jin, and Yongpan Wang. What is wrong with scene text recognition model comparisons? dataset and model analysis. In ICCV, 2019. 2, 3, 6, 7

[32] Pengyuan Lyu, Zhicheng Yang, Xinhang Leng, Xiaojun Wu, Ruiyu Li, and Xiaoyong Shen. 2d attentional irregular scene text recognizer. arXiv preprint arXiv:1906.05708, 2019. 7

[33] Pengyuan Lyu, Zhicheng Yang, Xinhang Leng, Xiaojun Wu, Ruiyu Li, and Xiaoyong Shen. 2d attentional irregular scene text recognizer. arXiv preprint arXiv:1906.05708, 2019. 7
