Improving CNN-RNN Hybrid Networks for Handwriting Recognition

Kartik Dutta, Praveen Krishnan, Minesh Mathew and C.V. Jawahar
CVIT, IIIT Hyderabad, India
{kartik.dutta, praveen.krishnan, minesh.mathew}@research.iiit.ac.in and jawahar@iiit.ac.in

***Abstract*—The success of deep learning based models have centered around recent architectures and the availability of large scale annotated data. In this work, we explore these two factors systematically for improving handwritten recognition for scanned off-line document images. We propose a modified CNN-RNN hybrid architecture with a major focus on effective training using: (i) efficient initialization of network using synthetic data for pre-training, (ii) image normalization for slant correction and (iii) domain specific data transformation and distortion for learning important invariances. We perform a detailed ablation study to analyze the contribution of individual modules and present state of art results for the task of unconstrained line and word recognition on popular datasets such as IAM, RIMES and GW.**

***Index Terms*—Handwriting recognition, CNN-RNN network, Data augmentation, Image pre-processing.**

I. Introduction

Handwritten text recognition (HWR) is one of key problems studied by the document community due to its pervasiveness in the places where people interact, communicate and transact. With better algorithms and technologies, we move a step closer to address the problem of content level access to ancient historical books and manuscripts which were written by hand and digitized in the form of scanned images as part of modern digital library [1], [2] projects. Other than searching in historical databases, HWR technologies can benefit the way we organize and manage modern classrooms [3], [4], search from instructional videos [3], and performing data analytics on medical transcripts [5] etc. The major challenges in recognizing text from handwritten images comes from the inherent variability in data. Every individual has a different style of writing and moreover, depending on the various underlying factors, even the style of a single person also changes in different instances of writings. In the historical documents one has to also deal with different degradation artifacts which hugely reduce the performance of the recognition systems.

A HWR problem is typically formulated into two parts, where given an image (word or line image), the task of the recognizer is to predict the character string ($\mathbf{w}$) which is later given to a linguistic engine which constrains $\mathbf{w}$ to form a valid word. These constraints can be either from lexicon $\mathcal{L}$ or a language model $P(\mathbf{w})$. Although there are methods which don't apply linguistic constraints, they are relatively worse in performance and cannot reliably be given to end applications.

In recent times, there has been a tremendous improvements in recognition rates [6]–[8] owing to the success of underlying machine learning models using deep neural networks. The improvement in performance can be credited due to: (i) better architectures such as convolutional neural networks (CNN), recurrent neural networks (RNN), (ii) better learning schemes and regularizers, (iii) availability of large scale of annotated data, and (iv) increased computational capacity using GPUs. Traditionally, in the domain of text recognition HMMs [9] or RNNs (especially BLSTMs [10] or MDLSTMs [11]) along with CTC loss [12] have been popularly used due to its inherent ability to process temporal sequences. More recently, it was shown that a hybrid scheme of using convolutional recurrent architecture [13] where the convolutional layers are meant for feature extraction, which are subsequently given to a BLSTM network along with CTC loss, work better than other schemes. In this work, we further analyze the effectiveness of a CNN-RNN hybrid architecture by incorporating various ways of data augmentation and normalization schemes which helps the network to learn better invariances specific to handwritten images. We also present our analysis on the role of using synthetic data for pre-training such deep networks which greatly helps in improving the recognition performance. Fig. 1 presents recognition results from the proposed network for sample word images taken from IAM and RIMES dataset respectively.

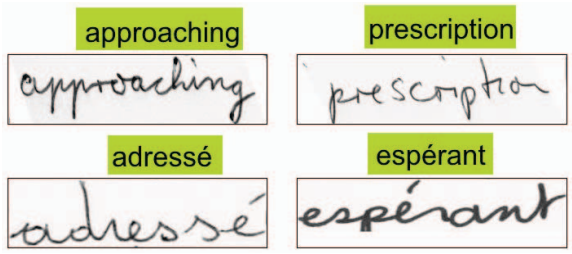

Fig. 1. Few sample qualitative results for word recognition on the IAM and RIMES dataset using the proposed method.

A. Related Works

Modeling HWR as a seq2seq problem using RNN's and CTC has been a widely used approach. With the popularity and robustness of CNNs, most works use convolutional layers as their feature extractors, while having different variations on the recurrent part of the network for transcription. Sueiras et al. [14] run a sliding window over the input image, where each patch is given to a convolutional feature extractor and later

978-1-5386-5875-8/18/$31.00 ©2018 IEEE
DOI 10.1109/ICFHR-2018.2018.00023

given to an encoder-decoder BLSTM network with attention for the transcription. Sun et al. [15] use a fully convolutional network as their feature extractor and use multi-directional (MDir) LSTM's as recurrent units. Pham et al. [16] use multi-dimensional LSTM's (MDLSTM) as their recurrent units while using convolutional layers as feature extractors. Chen et al. [17] use a multi-task network that is able to do both script identification and handwriting recognition simultaneously. They use a variation of the LSTM unit, referred to as SepMDLSTM.

Recently, Wigington et al. [18] use a network similar to [13], with BLSTM's as their recurrent layer. However they used novel normalization and image distortion strategies and achieved competitive results. Other than doing unconstrained and lexicon based decoding, a lot of works like [11], [19] use language model based decoding to reduce errors, especially in the line level recognition setting.

Another set of approach specifically for word-level recognition has been done using CNN architecture that evaluates whether a certain n-gram is present in a given portion of the image [20]. Krishnan et al. [21] also used a CNN to learn PHOC like attributes for images and embed the text represented through the PHOC features along with the embedded images into a common subspace. Both these methods are inspired on the PHOC representation proposed in [22].

A method by Toledo et al. [23] tries to combine both the above approaches for recognizing word level images, by first training a PHOCNet [24] for word attribute embedding and then embedding patches of word images into the attribute space. From the projections in the attribute space, a sequence is created and given to a recurrent network to perform transcription. Stuner et al. [25] use a cascades of LSTM's, rejecting a word during decoding if it is not close enough to a word in the lexicon after passing through the cascade and applying viterbi based decoding on the rejected words.

II. CNN-RNN Hybrid Network

Word recognition [10] is the problem of converting the handwritten content present in an image into machine understandable text. In this work we use a CNN-RNN hybrid architecture, first proposed by [13]. Fig. 2 illustrates the architecture that we use, which consists of a spatial transformer layer (STN) [26], followed by a set of residual convolutional blocks, proceeded by a stacked BLSTM (bidirectional LSTM) and ends with a linear layer for transcribing the labels. The STN network is an end-to-end trainable layer which performs geometric transformation on the input, so as to correct the distortions that are present in handwriting due to variable hand movements [27]. It can be used to correct various geometric transformations such as affine, thin plate spline, etc. The convolutional layers (ReseNet18 [6]) here are used for learning a sequence of feature maps, which are then passed on as input to the stacked BLSTMs. The last convolutional layer which obtain a feature map $F_l \in \mathbb{R}^{\alpha \times \beta \times \gamma}$ is given as an input to the recurrent layers (here BLSTMs) as a sequence of γ feature vectors, each $F_{l+1} \in \mathbb{R}^{\alpha \times \beta}$. We use the CTC [12] loss function

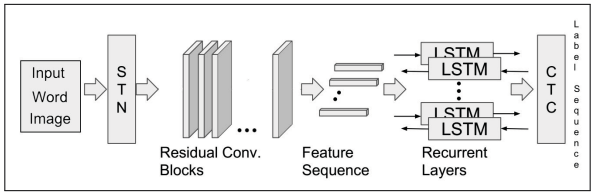

Fig. 2. Overview of the CNN-RNN hybrid network architecture used in this work. The various important components of the architecture are highlighted such as the spatial transformer network, residual convolutional blocks, bi-directional LSTM's and the CTC loss.

to train our network. It converts the predictions generated by the recurrent layers as a maximum probable sequence for the input. While testing the final classification layer, the SoftMax activation outputs the probability distribution over the class labels at each time step. In case of naive, lexicon free decoding, the output sequence is found by concatenating the most probable labels at each step and then removing recurring labels and blank labels. When a lexicon is available at the time of testing, the decoding can be constrained to output only the sequence of labels present in the lexicon.

III. Image Normalization and Augmentation

A. Pre-processing

We use the image slant and slope normalization technique proposed by [28] as an pre-processing step for both word and line level recognition. The algorithm works by shearing the image with an angle from a certain range of values and evaluating those transformations with respect to a function defined from the histograms of contours of nearly vertical strokes. The method requires no parameter tuning and is used directly on both isolated word level and line level images. The second row of Fig. 4 shows the output of our pre-processing step for a few sample word images.

B. Pre-training with Synthetic Data

A typical deep learning architecture contain millions of parameters to be learnt and thereby require a large amount of data in order to generalize well and prevent over-fitting. To overcome the lack of availability of real handwritten training data we use the IIIT-HWS dataset [29] for pre-training our isolated word recognition networks. The dataset is formed out of 750 publicly available Latin fonts and contains images with varying kerning level, stroke width and Gaussian noise. A vocabulary of size 90K words is used for creating the dataset. Fig. 3 shows few examples of natural looking synthetic images that were rendered through the mentioned pipeline in [29]. In order to pre-train our line recognition model, we followed a pipeline similar to the above, except that instead of rendering isolated words, we rendered lines, with the transcriptions being taken from the train set of the dataset that we were building our line recognition model.

C. Distortions and Transformations

Given a pre-trained network using synthetic data which gives a good initialization for the deep network, we now focus

Fig. 3. Examples of generated synthetic images in the IIIT-HWS dataset. The first two rows shows the same word rendered by different fonts. The last two rows shows different words being rendered by different fonts.

onto the various data augmentation schemes which supplements the real data in learning the desired invariances. While training a CNN network, it is a common practice to introduce artificial variations in data to make network robust to intra-class variations and prevent over-fitting. Popular techniques include: random crops, horizontal reflection, random flipping of pixels, and affine transformations such as scaling and translation. In this work, we use three types of augmentation schemes: (i) affine transformation, (ii) elastic distortion and (iii) multi-scale transformation both while training and testing. Under affine transformation we apply translation, scaling, rotation, and shearing. The second last row of Fig. 4 shows different possible variations while performing affine transformation to a word image. Here we restrict rotation to a random amount between $(+/-)5$ degrees, while shearing is restricted to $(+/-)0.5$ degrees along the horizontal direction which mimics the skew and cursiveness present in natural handwriting. We perform translation in terms of padding on all four sides, of upto 20 pixels in any direction, to simulate incorrect segmentation of words. We randomly apply a combination of the above 3 transformations to an input image.

Elastic distortion: Human handwriting has a high degree of oscillation due to the non-uniform hand muscle forces being exerted while writing. These variations can be captured to certain extend using elastic distortions which was first proposed in [30] for data augmentation of handwritten digits. We adapt a similar scheme for augmenting both word and line images. The basic idea is to generate a random displacement field which dictates the computation of new location to each pixel through interpolation. The displacement field is smoothed using a Gaussian filter of standard deviation σ and scaled using constant factor α. The last row of Fig. 4 shows different possible variations created for each word image while performing elastic distortion. We apply this distortion directly to word images as mentioned in [30], while in case of line images we apply the distortion in a sliding window fashion, with a window size of $25 \times h$, where h is the height of the line image.

Multi-scale transformations: The idea of multi-scale transformation is to learn to predict characters at multiple scales. The scale of a character is dependent on the context in which it occurs in a word. For example, a word image for "car" resized at a fixed height would have all characters in the same scale while another image corresponding to word "Car" would have different scale for the first character w.r.t to other characters. The above problem can also be generalized to ngrams occurring at different scales. To learn a scale invariant classification, we present the images at multiple scales, as shown in 3rd row of Fig. 4, allowing the network learn these while training. More recently in [18], Wigington et al. also presents a method which address this issue by normalizing the scale of all images present in a dataset using profile normalization. Our approach address the same issue with the help of data augmentation while training the network.

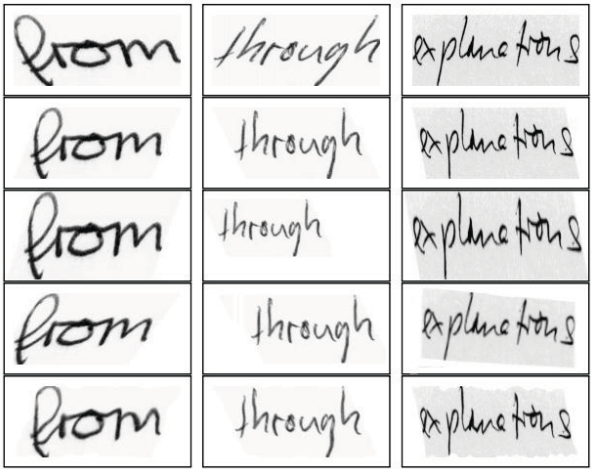

Fig. 4. Examples of various pre-processing and augmentation techniques used in the paper. The first row shows the original image. The 2nd row shows the images after pre-processing. The 3rd row shows the images after multi-scale augmentation is applied on the corresponding image from 2nd row. The 4th row shows the image after affine distortion is applied on the corresponding image from 2nd row. The final row shows the images after elastic distortion is applied on the corresponding image from 2nd row.

Test-time augmentation: We also performed test time augmentation similar to [20], where we generate $N = 25$ jittered images, by applying the pre-processing and data augmentation mentioned above, for each input test image, we average out the final layer probabilities for all the N images before the decoding step.

IV. Experiments

We use the character error rate (CER) and word error rate (WER) metrics to compare the various models. CER is defined as (where GT:ground truth and PT:predicted text):

$$CER = \frac{\sum_{i \forall samples} EditDistance(GT_i, PT_i)}{\sum_{i \forall samples} \#Chars(GT_i)}$$

and WER is defined as the number of words inserted, substituted or deleted, averaged over the total number of words present in the ground truth. In case of word level recognition, since there is no word alignment to be performed, the WER simplifies to the number of predicted words which don't match

TABLE I
THE LIST OF DATASETS USED IN THIS WORK.

Dataset	#Words	#lines	#Writers
IAM [31]	1,15,320	13,353	657
RIMES [32]	66982	12093	1300
GW [33]	4894	656	1

the ground truth, divided by the total number of words present in the ground truth.

We use the three most popular datasets used by handwritten document analysis community. Table I mentions different statistics about the datasets such as the number of words, lines and writers. Both IAM and RIMES dataset are modern collections, while the George Washington (GW) dataset is a historical collection.

IAM Handwriting Database [31]: It includes contributions from 657 writers, having a total of 13,353 handwritten lines, comprising of 115,320 words. The database is annotated at the sentence, line and word levels. We use the standard partition for training, testing, and validation provided along with the dataset. For the case of word level recognition, we use a lexicon made up of all the unique words in the dataset.

RIMES [32]: The ICDAR 2011 competition version of the RIMES database has contributions from over a thousand writers and has a total of 12,093 lines and 66,982 words. A train, val and test split was released for the isolated word recognition as part of the competition which we follow here. We use the lexicon that was released as part of the competition. The competition dataset also contained 11,333 training lines and 778 test lines. Since the competition release did not include a validation partition, we sampled 10% of the total training to create our validation set. Thus, the final dataset division consisted of the train, validation and test containing 10,203, 1,130 and 778 lines, respectively. For word level recognition, we use a lexicon that was released as part of the ICDAR 2011 competition.

For both IAM and RIMES dataset we don't consider punctuation or capital letters for recognition, similar to [20]. In line level recognition we follow the same setting as used in [19].

George Washington (GW) Database [33]: It contains 4894 word images written by George Washington and his associates in 1755. We use the first partition of the dataset for all our experiments, same as [23], resulting in 2433, 1293, 1168 word images for training, validation and testing respectively. Since the images provided were already normalized and binarized, no pre-processing was applied on the images of this dataset from our side. We use this dataset only for word level recognition and and keep all the punctuation and capital letters in the database, similar to the setting in [23]. In one experiment we use a lexicon made up of all the words in the dataset, while in another we use a lexicon made up of only the words in the train set.

A. Ablation study on IAM Dataset

Table II shows the recognition results of various variants of the CNN-RNN hybrid architecture on the test set of IAM dataset, considering word level segmentation. All of the decodings are performed in an unconstrained setting. The various models and their training strategies are mentioned below:

TABLE II
ABLATION STUDY OF THE CNN-RNN HYBRID ARCHITECTURE ON THE IAM DATASET CONSIDERING WORD LEVEL SEGMENTATION.

Method	WER	CER
CRNN-REAL [13]	22.86	11.08
CRNN-FULL	20.10	9.31
S-CRNN-FULL	18.3	7.82
S-D-CRNN-FULL	18.04	7.78
S-D-R-CRNN-FULL	16.19	6.34
S-D-R-CRNN-FULL-PP	15.79	5.98
S-D-R-CRNN-FULL-PP-DAUG	13.16	5.10
S-D-R-CRNN-FULL-PP-DAUG-TT	**12.61**	**4.88**

- CRNN-REAL is the original architecture of [13] and trained only on the IAM train set.
- CRNN-FULL is the original architecture of [13], first pre-trained on IIIT-HWS and then fine-tuned on IAM. Here we only use affine-transformations as mentioned in Section III-C for augmenting our data.
- S-CRNN-FULL integrates the STN layer as an initial layer of the original architecture of [13] and is trained using the same strategy as above.
- S-D-CRNN-FULL integrates the STN layer and we also apply dropout to the recurrent layers of the above architecture and is trained using the same strategy as above.
- S-D-R-CRNN-FULL integrates STN, dropout and residual learning. The network architecture now matches the architecture used in [34]. It is trained using the same strategy as above.
- S-D-R-CRNN-FULL-PP uses the same network architecture and training strategy as above, along with the preprocessing strategy mentioned in Section III-A.
- S-D-R-CRNN-FULL-PP-DAUG adds the data augmentation strategies mentioned in Section III-C to the above model.
- S-D-R-CRNN-FULL-PP-DAUG-TT uses test time augmentation as mentioned in Section III-C along with all of the attributes of the model above.

Table II empirically validates the benefit of using realistic synthetic data, as mentioned in Section III-B for pre-training purposes. The effectiveness of the STN layer in correcting distortions present in handwriting can also be seen. Since handwriting is usually cursive in shape, it is important to normalize for this by de-slanting, which we can observe from the above table as well. Table also shows that using multi scale transformation (to mimic variation in handwriting scales), elastic distortion (to simulate distortions due to hand movements) and test time augmentation lead to a significant reduction in error rates as compared to just using affine distortions based data augmentation. From the original CRNN-REAL model we progressively reduce the error and obtain an absolute reduction in our WER and CER by more than 55% and 44% respectively.

TABLE III
WORD RECOGNITION RESULTS ON THE IAM DATASET UNDER DIFFERENT EVALUATION SETTINGS SUCH AS THE KIND OF SEGMENTATION AND WHETHER A LEXICON WAS USED FOR MAKING THE PREDICTIONS. HERE FULL-LEXICON REFERS TO THE LEXICON CREATED FROM ALL THE DISTINCT WORDS IN THE DATABASE, WHILE TEST-LEXICON CONTAINS WORD ONLY FROM THE TEST SET.

Method	Seg.	Decoding	WER	CER
Krishnan et al. [35]	Word	Unconstrained	16.19	6.34
Wigington et al. [18]			19.07	6.07
Sueiras et al. [14]			23.8	8.8
This Work			**12.61**	**4.88**
Sun et al. [15]		Full-Lexicon	11.51	-
Wigington et al. [18]			5.71	3.03
Stuner et al. [25]			5.93	2.78
Poznanski et al. [20]			6.45	3.44
This Work			**4.80**	**2.52**
Sueiras et al. [14]		Test-Lexicon	12.7	6.2
Wigington et al. [18]			4.97	2.82
Krishnan et al. [21]			6.69	3.72
Krishnan et al. [35]			5.10	2.66
This Work			**4.07**	**2.17**
Pham et al. [16]	Line	Unconstrained	35.1	10.8
Puigcerver et al. [19]			18.4	5.8
Chen et al. [17]			34.55	11.15
Krishnan et al. [35]			32.89	9.78
This Work			**17.82**	**5.7**

TABLE IV
WORD RECOGNITION RESULTS ON THE RIMES DATASET UNDER DIFFERENT SETTINGS. HERE COMP. LEXICON REFERS TO THE LEXICON RELEASED AS PART OF THE ICDAR 2011 COMPETITION.

Method	Seg.	Decoding	WER	CER
Wigington et al. [18]	Word	Unconstrained	11.29	3.09
Sueiras et al. [14]			15.9	4.8
This Work			**7.04**	**2.32**
Wigington et al. [18]		Comp. Lexicon	2.85	1.36
Sueiras et al. [14]			6.6	2.6
Stuner et al. [25]			3.48	1.34
Poznanski et al. [20]			3.90	1.90
This Work			**1.86**	**0.65**
Pham et al. [16]	Line	Unconstrained	28.5	6.8
Chen et al. [17]			30.54	8.29
Puigcerver et al. [19]			**9.6**	**2.3**
This Work			14.70	5.07

B. Word Recognition Results

Table III, IV, V shows the quantitative word recognition results on the IAM, RIMES and GW dataset respectively, using the best model from Table II, fine-tuned on the train set of the respective dataset. Here we compare various methods under different settings such as:- (i) level of segmentation (words/lines), (ii) presence/absence of lexicon while decoding the output.

For almost all the settings on the IAM and RIMES dataset we report the state of the art results. This is despite the fact that works like [14], [15], [18], [35], etc. are broadly using a CNN-RNN hybrid network similar to ours. As Section IV-A shows this improvement can be attributed to using all the following techniques: pre-processing, data & test time augmentation, pre-training with synthetic data and using architectural improvements such as using STN and residual blocks in feature extraction.

TABLE V
WORD RECOGNITION RESULTS ON THE GEORGE WASHINGTON DATASET UNDER DIFFERENT SETTINGS. HERE FULL-LEXICON REFERS TO THE LEXICON CREATED FROM ALL THE DISTINCT WORDS IN THE DATABASE, WHILE TRAIN-LEXICON ONLY CONTAINS WORDS FROM THE TRAIN SET.

Method	Seg.	Decoding	WER	CER
Fischer [36]	Word	Unconstrained	-	20
Toledo et al. [23]			-	7.32
This Work			**12.98**	**4.29**
Almazán et al. [22]		Full Lexicon	-	17.40
This Work			**12.59**	**3.81**
Almazán et al. [22]		Train Lexicon	-	22.15
This Work			**29.54**	**12.29**

Despite the GW dataset being a single writer dataset we don't obtain nearly as good word recognition results for it as the other 2 datasets. This is explained by various factors. First, the small size of the train set in GW and the use of binarized images instead of grayscale images in the other datasets. Secondly, none of the extra characters present in GW (old English "G", punctuation, etc.) are part of our synthetic dataset. These characters also reduce the effectiveness of lexicon based decoding and we observe a lot of confusion between capital and lower-case letters in the case of lexicon based decoding. The train lexicon in GW has an out of vocabulary (OOV) rate of about 16%, which accounts for our worse performance when using the train lexicon.

C. Visualization

To better visualize the working of the convolutional layers of our network, in Fig. 5 we visualize the activations of a few channels from the second convolutional layer, where certain word images from the IAM dataset are given as input to the network. As we can see, certain channels in the second convolution layer activate on the foreground and background, while others activate on the horizontal or vertical ligatures in the handwritten image. Fig. 6 show the recognized output on a few sample word images from the IAM and RIMES dataset. Here we are decoding using an unconstrained setting. We can observe that most of the errors occur due to the lack of clarity in the original handwritten image regarding the shape of alphabets or due to improper segmentation.

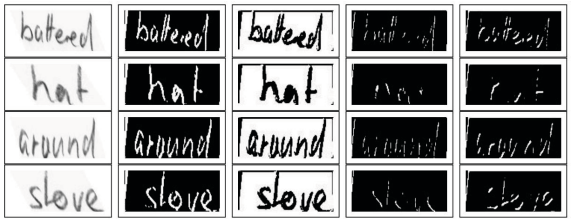

Fig. 5. Visualizations of the activations learnt by the 2nd convolutional layer of our network. The first column shows the pre-processed input image (taken from the IAM dataset). The second column shows the corresponding activations of a channel which acts as a textual edge detector. The third column shows a channel which activates on the image background. The fourth column shows a channel which acts like a vertical line detector, while the last column shows a channel which acts like a horizontal line detector.

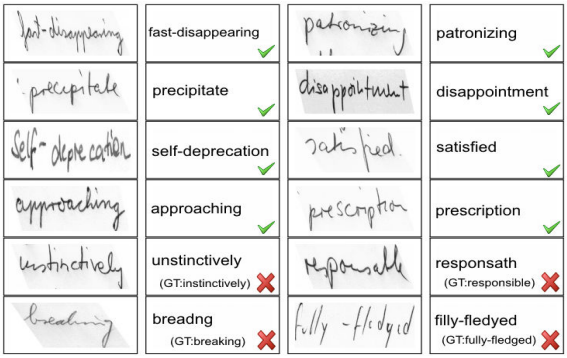

Fig. 6. Qualitative results of word recognition on the IAM dataset.

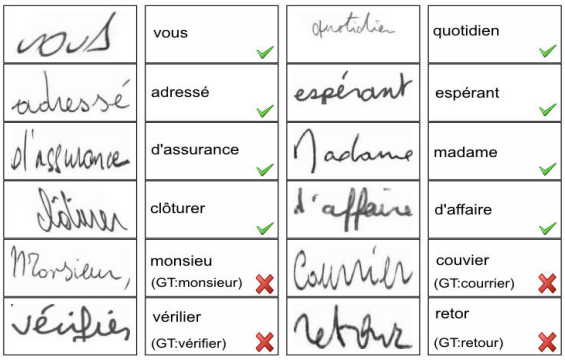

Fig. 7. Qualitative results of word recognition on the RIMES dataset.

V. Conclusion

In this work, we presented effective ways to train a CNN-RNN hybrid architecture using synthetic data and domain specific image normalization and augmentation. We also showed the individual contributions of each of these modules for improving the recognition rates at both line and word levels. In future, we would also like to integrate our line-level recognition model with language model based decoding so as to further enhance our recognition performance.

References

[1] J.-B. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant *et al.*, "Quantitative analysis of culture using millions of digitized books," *Science*, 2011.

[2] N. Balakrishnan, R. Reddy, M. Ganapathiraju, and V. Ambati, "Digital library of India: a testbed for Indian language research," *TCDL*, 2006.

[3] P. Krishnan and C. V. Jawahar, "Matching handwritten document images," in *ECCV*, 2016.

[4] R. J. Milewski, V. Govindaraju, and A. Bhardwaj, "Automatic recognition of handwritten medical forms for search engines," *IJDAR*, 2009.

[5] R. Milewski and V. Govindaraju, "Handwriting analysis of pre-hospital care reports," in *CBMS*, 2004.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[7] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning." in *AAAI*, 2017.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[9] U.-V. Marti and H. Bunke, "Using a statistical language model to improve the performance of an hmm-based cursive handwriting recognition system," in *IJPRAI*, 2001.

[10] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *PAMI*, 2009.

[11] P. Voigtlaender, P. Doetsch, and H. Ney, "Handwriting recognition with large multidimensional long short-term memory recurrent neural networks," in *ICFHR*, 2016.

[12] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.

[13] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *PAMI*, 2016.

[14] J. Sueiras, V. Ruiz, A. Sanchez, and J. F. Velez, "Offline continuous handwriting recognition using sequence to sequence neural networks," *Neurocomputing*, 2018.

[15] Z. Sun, L. Jin, Z. Xie, Z. Feng, and S. Zhang, "Convolutional multi-directional recurrent network for offline handwritten text recognition," in *ICFHR*, 2016.

[16] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour, "Dropout improves recurrent neural networks for handwriting recognition," in *ICFHR*, 2014.

[17] Z. Chen, Y. Wu, F. Yin, and C.-L. Liu, "Simultaneous script identification and handwriting recognition via multi-task learning of recurrent neural networks," in *ICDAR*, 2017.

[18] C. Wigington, S. Stewart, B. Davis, B. Barrett, B. Price, and S. Cohen, "Data augmentation for recognition of handwritten words and lines using a CNN-LSTM network," in *ICDAR*, 2017.

[19] J. Puigcerver, "Are multidimensional recurrent layers really necessary for handwritten text recognition?" in *ICDAR*, 2017.

[20] A. Poznanski and L. Wolf, "CNN-N-Gram for handwriting word recognition," in *CVPR*, 2016.

[21] P. Krishnan, K. Dutta, and C. V. Jawahar, "Deep feature embedding for accurate recognition and retrieval of handwritten text," in *ICFHR*, 2016.

[22] J. Almazán, A. Gordo, A. Fornés, and E. Valveny, "Word spotting and recognition with embedded attributes," *PAMI*, 2014.

[23] J. I. Toledo, S. Dey, A. Fornés, and J. Lladós, "Handwriting recognition by attribute embedding and recurrent neural networks," in *ICDAR*, 2017.

[24] S. Sudholt and G. A. Fink, "Phocnet: A deep convolutional neural network for word spotting in handwritten documents," in *ICFHR*, 2016.

[25] B. Stuner, C. Chatelain, and T. Paquet, "Handwriting recognition using cohort of LSTM and lexicon verification with extremely large lexicon," *CoRR, vol. abs/1612.07528*, 2016.

[26] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *NIPS*, 2015.

[27] W. Liu, C. Chen, K.-Y. K. Wong, Z. Su, and J. Han, "Star-net: A spatial attention residue network for scene text recognition." in *BMVC*, 2016.

[28] A. Vinciarelli and J. Luettin, "A new normalization technique for cursive handwritten words," *PR Letters*, 2001.

[29] P. Krishnan and C. V. Jawahar, "Generating synthetic data for text recognition," *arXiv preprint arXiv:1608.04224*, 2016.

[30] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis." in *ICDAR*, 2003.

[31] U.-V. Marti and H. Bunke, "The IAM-database: an english sentence database for offline handwriting recognition," *IJDAR*, 2002.

[32] E. Grosicki and H. El-Abed, "ICDAR 2011-french handwriting recognition competition," in *ICDAR*, 2011.

[33] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "Lexicon-free handwritten word spotting using character hmms," *PR*, 2012.

[34] K. Dutta, P. Krishnan, M. Mathew, and C. V. Jawahar, "Unconstrained handwriting recognition on devanagari script using a new benchmark dataset," in *DAS*, 2018.

[35] P. Krishnan, K. Dutta, and C. V. Jawahar, "Word spotting and recognition using deep embedding," in *DAS*, 2018.

[36] A. Fischer, "Handwriting recognition in historical documents," Ph.D. dissertation, University of Bern, 2012.