Towards Accurate Handwritten Word Recognition for Hindi and Bangla

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. Building accurate lexicon free handwritten text recognizers for Indic languages is a challenging task, mostly due to the inherent complexities in Indic scripts in addition to the cursive nature of handwriting. In this work, we demonstrate an end-to-end trainable CNN-RNN hybrid architecture which takes inspirations from recent advances of using residual blocks for training convolutional layers, along with the inclusion of spatial transformer layer to learn a model invariant to geometric distortions present in handwriting. In this work we focus building state of the art handwritten word recognizers for two popular Indic scripts – Devanagari and Bangla. To address the need of large scale training data for such low resources languages, we utilize synthetically rendered data for pre-training the network and later fine tune it on the real data. We outperform the previous lexicon based, state of the art methods on the test set of Devanagari and Bangla tracks of RoyDB by a significant margin.

Keywords: Handwriting Recognition, Lexicon Free, Indic Scripts

1 Introduction

The creation and dissemination of handwritten documents remains pervasive for humans as a personal choice of communication other than speech. Handwritten text recognition is the process of automatic conversion of such handwritten documents into machine encoded text. It has been a popular research area for many years due to various applications such as digitizing handwritten manuscripts [1], postal automation [2], matching documents [3], digitizing handwritten medical forms [4], etc. Most of these works have focused on Latin scripts, with very few works in the space of Indic scripts. In this work, we address the challenges of building a handwritten word recognizer for two popular Indic scripts – Devanagari and Bangla. A lot of handwritten documents in these scripts have been made available by scanning historical documents, ancient manuscripts and literary resources with cultural significance. Extracts from such Bangla and Devanagari manuscripts is shown in Figure 1.

Devanagari and Bangla are the two most popular Indian scripts, also being the fifth and sixth most popular language in the world [5]. Both these scripts

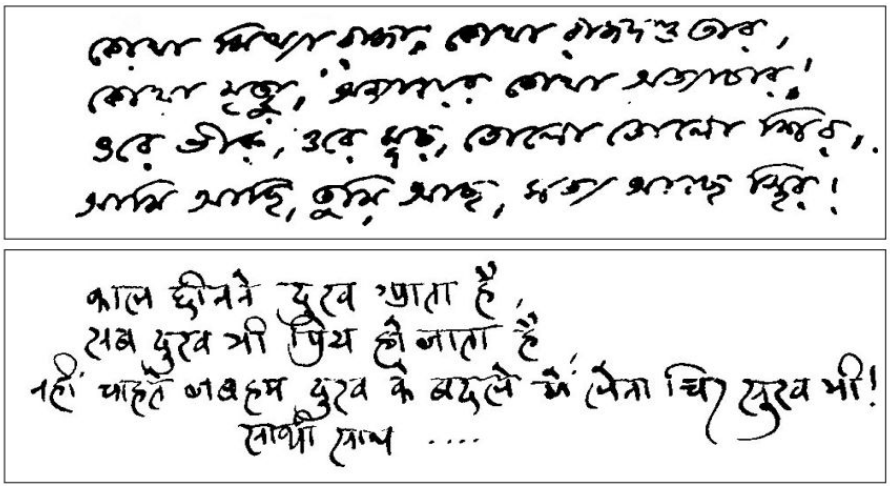

Fig. 1. The top box shows an excerpt from the Bangla poem "Namashkar" (1907) written by Rabindranath Tagore, while the bottom box shows an example of Devanagari writing from a Hindi poem written by Amitabh Bachchan.

are read left to right. Both scripts contain 11 vowels whereas the number of consonants is 38 for Devanagari and 39 for Bangla. Figure 2 shows the basic set of characters in both the scripts. As one can notice both the scripts contains a horizontal line running across the characters and words (for word images see fig. 4) which is referred as *Shirorekha*. The shape of the consonant is modified in case it is followed by a vowel in either script, and such a character is referred to a *modified* character or modifier. In certain cases, when a consonant follows one or more consonant(s), a new character gets formed which has an orthographic shape and is called a *compound* character. For more details about the Bangla and Devanagari scripts, the reader can look at [6]. Due to the presence of modifiers and compound characters, the number of distinct characters possible in both Bangla and Devanagari is far higher than Latin scripts (roughly 400 distinct characters in Bangla compared to only 62 in English), making word recognition for these Indic scripts more challenging as compared to English.

In addition to the script level challenges, a handwritten word recognizer for Indic scripts has to deal with challenges associated with writers with different styles and the cursive nature of the handwriting. The cursive way of writing [7] results in merging of adjacent characters, variable skew and modified shapes which further increases the complexity in recognition. Another important challenge is the lack of publicly available handwritten data in Indian languages. This inhibits training of modern deep learning architectures, which contain millions of parameters and require substantial data for generalization of features.

Most of the previous works [8–10] on recognizing Devanagari and Bangla scripts were limited to printed documents. In this work we focus on handwritten word images which are more challenging than machine printed words. Initial

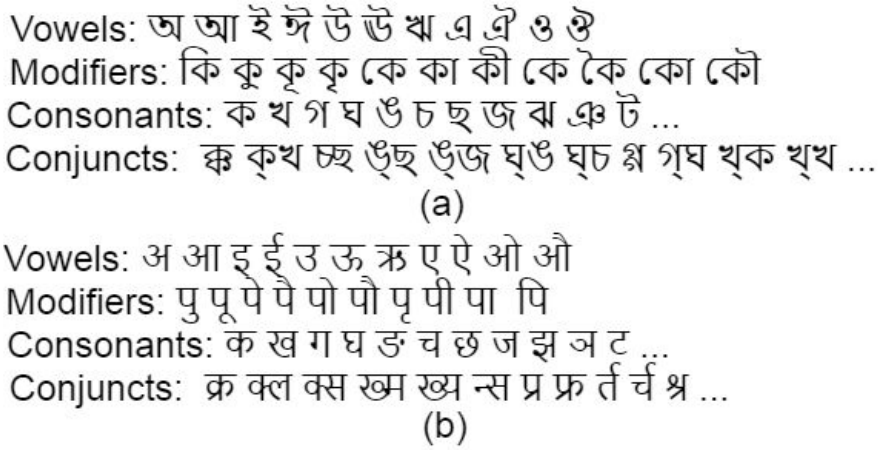

Fig. 2. Few examples of vowels, modifiers, consonants and conjuncts in (a) Bangla and (b) Devanagari

works in offline Bangla and Devanagari handwritten word recognition used Hidden Markov Models (HMM). HMM based approaches can be categorized as – (i) trained on recognizing the whole word or finding the holistic representation (ii) segmenting the image into different zones and using the HMM to recognize individual characters. Works such as [11, 12] for Bangla and [13, 14] for Devanagari, extract features from the entire word and use lexicon dependent HMM decoding to recognize the whole word. In [15] for Bangla and Devanagari word recognition, the authors first segment the word image into three regions, namely the upper, middle and lower zone, using image processing techniques such as skeletal analysis and shape matching. The upper and lower zones are recognized by using support vector machines while the middle zone was recognized using a HMM decoded with a lexicon of middle zone characters. Finally, the results from the recognizers in all three zones are combined. Both these methods are limited to lexicon based decoding. More recently, with the proliferation of deep learning based methods, modern text recognizers are built using a combination of convolutional neural networks (CNN) and recurrent neural networks (RNN) such as BLSTMs [16]. In [17], Garain et al. uses BLSTM's with CTC loss, while in [7] CNN's are used for feature extraction along with RNN's for sequence classification. Both these methods show results for offline Bangla handwritten recognition. While our work is similar to that in [7] in terms of basic architecture, our network is much deeper, uses residual connections and contains spatial transformation layer to better handle geometric distortions present in handwriting. Also, in [7] each modifier and compound character is treated as a unique character, while we model it as a sequence of characters.

In this work, we address the challenges associated with recognizing the handwritten word images of Devanagari and Bangla scripts, in an unconstrained setting, by using a CNN-RNN hybrid network. The main contributions of this work

are as follows- (i) To address the lack of data we pre-train our network on synthetic data created from fonts and fine tune the network of real world images, (ii) We use a network similar to the one used in [18], containing a spatial transformer layer, residual convolutional blocks and BLSTM layers along with the CTC loss function for sequence to sequence transcription and (iii) We present results in both lexicon-based and lexicon-free (unconstrained) setting. Our proposed method gives state of the results on both Bangla and Devanagari tracks of RoyDB [15].

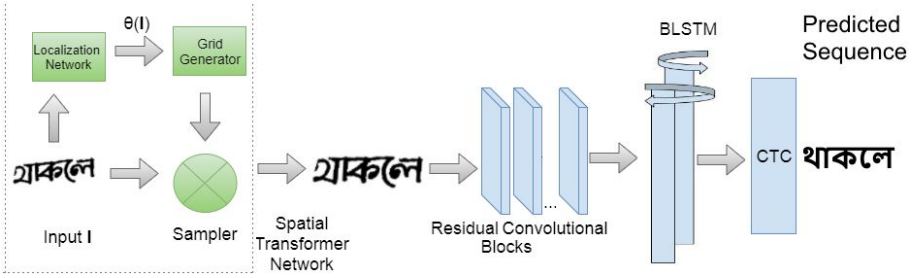

Fig. 3. Overview of the proposed CNN-RNN hybrid architecture. The various important components of the architecture are highlighted such as the spatial transformer network, residual convolutional blocks, BLSTM layers and the CTC loss function.

The paper is structured as follows: In section II, we discuss about the CNN-RNN hybrid architecture and its notable components. In section III, we discuss how we train our deep model and the result of applying our pre-trained model to both the Bengali and Devanagari track of the RoyDB dataset. Section IV concludes our work.

2 Methodology

Figure 3 illustrates the proposed deep architecture for lexicon free handwritten word recognition which consists of a spatial transformer layer (STN) [19], followed by a set of residual convolutional blocks, which is proceeded by a stacked bi-directional LSTM module and ends with a transcription layer for label prediction.

2.1 Synthetic Data

The availability of huge amount of data is crucial for successful training of deep architectures which typically contain millions of parameters. In [20, 21], a framework for rendering synthetic word images from standard fonts is proposed which practically enables building an nearly infinite vocabulary dataset. In this work, we follow a pipeline similar to [21] for rendering word images from Bangla and

Devanagari fonts to pre-train our deep network. The word is rendered onto the image in one of the following three ways: with a horizontal bottom text line or following a random curve or following a straight line. We apply varying amounts of kerning while rendering along with gaussian noise.

2.2 CNN-RNN Hybrid Architecture

In this work, we use the convolutional recurrent neural network (CRNN) [22] architecture along with STN layer [18, 23] proposed for scene text recognition. In our work we show that such a network can be adapted for robust recognition of handwritten word images for Indic scripts. A CRNN architecture consists of a set of convolutional layers, followed by a recurrent neural network units whose output is given to a transcription layer which is modeled using connectionist temporal classification [24]. In general, convolutional neural networks have been found to be excellent spatial feature extractors [25–27] with translation invariant properties while recurrent neural units can take a sequence of feature vectors of variable length and can perform sequence to sequence (seq-2-seq) transcription tasks. In our case, the input sequence of features is constructed from the feature maps of the last convolutional layer by concatenating column features across different channels. For example given the feature map of size $512 \times 5 \times 10$, it results in a sequence of 10 feature vectors with dimension $\mathbb{R}^{5\times512}$. Here we choose the column width to be single pixel. Given a sequence of feature vectors $f_1, f_2, \ldots, f_T$, we forward it to a stacked set of recurrent layers which is our case is a bi-directional LSTM [16] network. The BLSTM network considers both the forward and backward context (history) while making prediction from the label space at each time step. In our case, the label space consists of the basic (no modifiers and conjuncts) character set of the given language, plus a blank symbol. Finally, the CTC layer converts the predictions generated by the BLSTM output layer into a maximal probable label sequence for the target language. One of the key advantages of the above framework is that the input images can take varying input sizes, thus avoiding distortion in the aspect ratio, since both convolutional and recurrent layers can operate with variable size images and feature sequences respectively.

2.3 Spatial Transformer Network (STN)

The spatial transformer network [19] is an end-to-end trainable layer which can perform an explicit geometric transformation to the input. The transformation parameters are learnt through backpropagation. As seen in Figure 3 it has three main components, the localization network, the grid generator and the sampler. The localization network takes as input a feature map and outputs θ, the transformation parameters that is to be applied to the input feature map. The size of θ depends on the kind of transformations being modeled- affine, thin plate spline, etc. The localization network in itself is modeled as a neural network having a $|\theta|$ dimensional FC layer at the end. The grid generator generates a grid of coordinates in the input feature map corresponding to each pixel from

Table 1. Summary of the network configuration. The width, height and number of channels of each convolution layer is shown in square brackets, with the number of layers that are stacked together. After all but the last block, max pooling is applied. The width and height of the max pooling kernel are shown below each block. The number of units in each BLSTM layer is shown in square brackets, with the number of layers that are stacked together.

Block1 (2x2)	Block2 (2x2)	Block3 (1x2)	Block4 (1x2)	Block5	BLSTM
[3x3,64]x5	[3x3,128]x4	[3x3,256]x4	[3x3,512]x4	[3x3,512]x1	[256]x2

the output feature map. Finally, the sampler generates the output feature map using bilinear interpolation by sampling the pixels given by the grid generator after applying the learnt transformation. As [18, 23] show, this layer transforms the input feature map such that the geometric transformation is removed from the input and only the relevant part of the input is forwarded to subsequent layers. In the case of handwritten images the role of STN layer is particularly important to handle the variations caused due to the variable hand movements.

2.4 HW Word Recognition Network

Recent studies in deep learning [27, 28] on the Imagenet challenge have shown that deeper architectures lead to an improvement in classification accuracy. However, merely increasing the number of layers would bring challenges during training such as exploding/vanishing gradients, internal covariate shifts [29], and the degradation problem [30]. In our architecture, we take into account the best practices proposed in the recent literature to overcome these problems. We use smaller size 3×3 convolutional filters which enable us to obtain a bigger receptive field with lesser number of parameters. Batch Normalization [29] is applied before each convolutional layer to prevent internal covariate shifts. The convolutional filters are arranged into multiple residual blocks with skip connections as proposed in [31]. The residual layers enables to overcome the degradation problem which prevents the network to learn efficiently. Also, as [32] demonstrated that using dropout in the recurrent layers improved accuracy, we apply dropout with $p = 0.2$ at the BLSTM layers in our proposed model.

3 Recognition Experiments

In this work, we use the public benchmark **Indic Word Database / RoyDB [15]** to conduct experiments and validate our results. It has been compiled by 60 writers and consists of a Bengali and a Devanagari track. The Bengali track comprises of 17,091 binarized HW word images, while the Devanagari track comprises of 16,128 HW grayscale word images. On an average, an image in either track represents a word consisting of 4 characters. We use the word level annotations that is provided along with RoyDB and follow the training, validation

Table 2. Word recognition performance of the CNN-RNN hybrid model in comparison with state of the art methods on the test set of RoyDB.

Track	Method	WER	CER
Bangla	Roy et al. [15] – Lexicon	16.61	-
	Adak et al. [7] – Lexicon	14.58	-
	Ours – Lexicon	**4.30**	**2.05**
	Ours – Unconstrained	**10.71**	**3.49**
	Ours (Without Synth. Data)– Unconstrained	12.77	3.85
Devanagari	Roy et al. [15] – Lexicon	15.76	-
	Ours – Lexicon	**4.62**	**2.67**
	Ours – Unconstrained	**11.89**	**4.9**
	Ours (Without Synth. Data) – Unconstrained	14.09	5.53

and testing partition provided along with the dataset. We use the standard evaluation metrics of Character Error Rate (CER) and Word Error Rate (WER) to compare the various models. CER is defined as (where RT: recognized text and GT: ground truth text in the below equation)-

$$CER = \frac{\sum EditDistance(GT, RT)}{\#Characters}$$

i.e. the sum of insertions, deletions and substitutions in terms of characters required to transform RT to GT, divided by the number of characters in the ground truth. WER is the defined as the mean number of words wrongly transcribed.

Architecture Details: Table 1 lists the architecture of our network, along with the details of the convolution and recurrent layers used. For the localization network in our STN, we used three plain convolutional blocks and two linear layers. All the convolutional blocks have filter size, stride and padding of 3x3, 1 and 1 respectively. The number of channels in these layers were 64,64,128. 2x2 max pooling is applied before the first convolutional block and after each convolutional block as well. The first linear layer has 30 units and the second one has 6 units for learning the parameters of the affine transformation.

Pre-Training: To address the lack of enough training data, we use 1M word images, generated for both Bangla and Devanagari as described in Section 2.1, comprising of 50 different fonts for both the scripts. We use the train and validation corpus of RoyDB, for either track, as the rendering vocabulary to avoid undue advantages while comparing with other methods. In practice, the generation of synthetic data can span to nearly infinite vocabulary.

Data Augmentation and Fine-Tuning: To make the network invariant to affine transformations, we augmented each train image in RoyDB by applying a random amount of rotation (+/- 5 degrees), shearing (+/- 0.5 degrees along horizontal direction) and perform translation in terms of padding on all fours sides to simulate incorrect segmentation of words. After the model converges on

the synthetic dataset, we fine tune the network using the augmented train data of RoyDB in individual tracks.

Fig. 4. Recognition results for the CNN-RNN hybrid model on RoyDB dataset. First 4 rows show results for Bangla and the rest for Devanagari. Other than binarization and segmentation issues in the images, most of the errors are caused by ambiguities in the original handwritten image.

3.1 Results and Analysis

Table 2 shows the quantitative results of our model on both Bengali and Devanagari tracks of RoyDB, alongside the state of art methods in either track. We present our results in three different settings- (i) Lexicon based decoding, where the CTC layer selects a sequence from the test corpus lexicon with the highest likelihood during decoding. The methods proposed in [7,15] show results in this setting, (ii) Lexicon free or "unconstrained" setting in which decoding is not restricted to any target lexicon and finally (iii) we also present our results without using any synthetic data, in the lexicon free setting. In all these scenario we improve the state of the art results with significant margins. When compared to previous lexicon based methods [7, 15], we report an WER 4.30 and 5.13 for

Bangla and Devanagari respectively. In this work, we primarily focus on the unconstrained lexicon setting. In this setting, we report better results from lexicon based methods with a relative improvement in WER of about 26% in the Bangla and 22% in the Devanagari tracks. We also observe that our networks perform better even without using synthetic data when compared with previous methods. The use of synthetic data provides us an absolute additional reduction of WER by 2% in both scripts. Figure 4 shows the recognized outputs for few sample images from both tracks of RoyDB. Figure 5 visualizes the activations of a few channels from the 2nd convolution layer when an image is passed through the network.

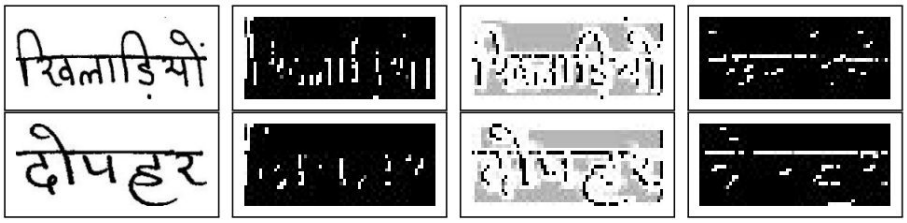

Fig. 5. Visualization of few channels from layer 2 activations. (From Left) first column shows the two input images. Second column shows the activation of a channel which detects vertical lines. The third column shows a channel which activates on the image background while the fourth column shows a channel which acts like a horizontal line detector.

3.2 Implementation Details

In all the experiments, the network is trained using stochastic gradient descent with the ADADELTA [33] optimizer. We initialize the parameters of the STN to represent the identity transformation. The input images are resized to 96x256. We used a batch size of 64 for training. Both Bangla and Devanagari models took around 15 hours to train on a single NVIDIA GTX 1080 Ti GPU. The character set for each language was taken from the set of unique characters in the respective Bangla and Devanagari track of the RoyDB database.

4 Conclusion

We demonstrate state of the art lexicon free handwritten word recognizers for Devanagari and Bangla scripts using a CNN-RNN hybrid model using modern architectural components. The idea of pre-training the network on synthetic data can pave way for solving text recognition problems for languages where datasets of adequate sizes are not available. As a future work, we would like to work with historical manuscripts and also relax the assumption of having pre-segmented words by incorporating automatic text localization.

Acknowledgement

This work was partly supported by IMPRINT scheme, Govt. of India. The authors would also like to thank Oishika, Sounak and Sreya for their help in verifying the results for Bangla.

References

1. Rath, T.M., Manmatha, R.: Word spotting for historical documents. IJDAR (2007)
2. Srihari, S.N., Kuebert, E.J.: Integration of hand-written address interpretation technology into the united states postal service remote computer reader system. In: DAS. (1997)
3. Krishnan, P., Jawahar, C.: Matching handwritten document images. In: ECCV. (2016)
4. Milewski, R.J., Govindaraju, V., Bhardwaj, A.: Automatic recognition of handwritten medical forms for search engines. IJDAR (2009)
5. Simons, G.F., Fennig, C.D.: Ethnologue: Languages of the world. SIL International (2017)
6. Pal, U., Chaudhuri, B.: Indian script character recognition: a survey. PR (2004)
7. Adak, C., Chaudhuri, B.B., Blumenstein, M.: Offline cursive bengali word recognition using CNNs with a recurrent model. In: ICFHR. (2016)
8. Chaudhuri, B., Pal, U.: A complete printed bangla OCR system. PR (1998)
9. Bansal, V., Sinha, R.: A complete OCR for printed hindi text in devanagari script. In: DAS. (2001)
10. Mathew, M., Singh, A.K., Jawahar, C.: Multilingual OCR for indic scripts. In: DAS. (2016)
11. Bhowmik, T.K., Parui, S.K., Roy, U.: Discriminative HMM training with GA for handwritten word recognition. In: ICPR. (2008)
12. Bhowmik, T.K., Roy, U., Parui, S.K.: Lexicon reduction technique for bangla handwritten word recognition. In: DAS. (2012)
13. Shaw, B., Bhattacharya, U., Parui, S.K.: Combination of features for efficient recognition of offline handwritten devanagari words. In: ICFHR. (2014)
14. Shaw, B., Parui, S.K., Shridhar, M.: Offline handwritten devanagari word recognition: A holistic approach based on directional chain code feature and HMM. In: ICIT. (2008)
15. Roy, P.P., Bhunia, A.K., Das, A., Dey, P., Pal, U.: Hmm-based indic handwritten word recognition using zone segmentation. PR (2016)
16. Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. PAMI (2009)
17. Garain, U., Mioulet, L., Chaudhuri, B.B., Chatelain, C., Paquet, T.: Unconstrained bengali handwriting recognition with recurrent models. In: ICDAR. (2015)
18. Liu, W., Chen, C., Wong, K.Y.K., Su, Z., Han, J.: Star-net: A spatial attention residue network for scene text recognition. In: BMVC. (2016)
19. Jaderberg, M., Simonyan, K., Zisserman, A., et al.: Spatial transformer networks. In: NIPS. (2015)
20. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Synthetic data and artificial neural networks for natural scene text recognition. (2014)

21. Krishnan, P., Jawahar, C.: Generating synthetic data for text recognition. arXiv preprint arXiv:1608.04224 (2016)
22. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. PAMI (2016)
23. Shi, B., Wang, X., Lyu, P., Yao, C., Bai, X.: Robust scene text recognition with automatic rectification. In: CVPR. (2016)
24. Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: ICML. (2006)
25. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. (2012)
26. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Reading text in the wild with convolutional neural networks. IJCV (2016)
27. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
28. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR. (2015)
29. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML. (2015)
30. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016)
31. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: ECCV. (2016)
32. Pham, V., Bluche, T., Kermorvant, C., Louradour, J.: Dropout improves recurrent neural networks for handwriting recognition. In: ICFHR. (2014)
33. Zeiler, M.D.: Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)