Handwritten Text Recognition using Deep Learning Methods

*

1[st] Hagar Hany Hassan
Computer Engineering Department
Bahçeşehir University
İstanbul, Türkiye
Hagar.hany.hassan@gmail.com

2[nd] Ayla GÜLCÜ
Software Engineering Department
Bahçeşehir University
İstanbul, Türkiye
ayla.gulcu@eng.bau.edu.tr

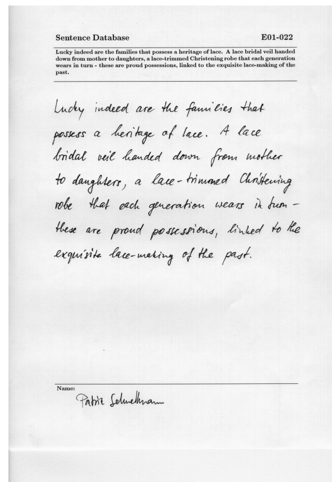
***Abstract*—Offline handwritten text recognition has been widely utilized in various fields including historical document analysis. Deep learning techniques have demonstrated their effectiveness in digitizing handwritten text as each technique is precisely designed to tackle a specific task or solve a particular problem. In this article, we use convolutional neural network for extracting distinct character features and a recurrent neural network for handling character combinations within sequential data. By combining these models, we create a hybrid deep neural network consisting of three CNN layers followed by a bidirectional LSTM layer. This architecture effectively encodes input images and generates character probability matrices with which the connectionist temporal classification operation computes the loss function. Extensive experimentation with various parameter values allowed us to optimize our model, which we evaluated on the IAM dataset, yielding a reasonably low error rate.**

***Index Terms*—Handwritten text recognition, convolutional neural networks, Recurrent neural networks, CTC loss**

I. Introduction

Recognizing people's handwriting is an overly complicated task since each person has a unique writing style. The cursive nature of manual writing, the gaps between characters in a word, the gaps between words, and the shape of a small or large vocabulary make the task of handwritten text recognition (HTR) difficult. Despite these difficulties, many researchers have been dealing with these problem because it can be used to extract data in many domains. Recognizing postal addresses, identifying names and amounts in bank checks are examples to these problems.

There are two types of HTR as online and offline text recognition. The former is easier than the latter since the handwritten text is written on any sensitive device, and the task of recognition is immediately implemented on the available spatial and temporal information. Offline handwritten recognition which is also called optical character recognition (OCR) [47], can be applied in several steps as follows: i) Digitizing the document by scanning, ii) Segmenting into multiple lines, words, or characters, iii) Applying OCR for character recognition, and iv) Error correction by applying some spelling checkers or lexicons.


Sentence Database **E01-022**

Lucky indeed are the families that possess a heritage of lace. A lace bridal veil handed down from mother to daughters, a lace-trimmed Christening robe that each generation wears in turn - these are proud possessions, linked to the exquisite lace-making of the past.

Lucky indeed are the families that
possess a heritage of lace. A lace
bridal veil handed down from mother
to daughters, a lace-trimmed Christening
robe that each generation wears in turn -
these are proud possessions, linked to the
exquisite lace-making of the past.

Name:

Fig. 1. A sample page from IAM Dataset

There are several handwritten text datasets in the literature that are publicly available to the researchers. IAM [52] is the most widely HTR dataset which contains 1500 pages of handwritten pages, 13 thousand lines and a total of 115 thousand words written by 657 different writers in English. CVL dataset [16] is another commonly used dataset which contains 1500 different pages written in different languages including English and German. RIMES Dataset [15] is another commonly used dataset for the task of HTR written in French with about 12,000 page samples written by 1,300 different writers . A sample page from IAM dataset is shown in Figure I

The methods for HTR can be divided into 2 major categories as Hidden Markov Models (HMM) [9], Artificial Neural Network-based models using Convolutional Neural Networks (CNN) [47] and Recurrent Neural Networks (RNN). There are also hybrid methods such as the one in [16] that use both HMM and Artificial Neural Networks. In several hybrid methods, different versions of RNNs are hybridized with CNNs.

2023 7th International Electromagnetic Compatibility Conference (EMC Turkiye) | 979-8-3503-4213-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/EMCTurkiye59424.2023.10287418

Gated Convolution-Recurrent Neural Network (GCRNN) [7], CNN-Long Short Term Memory (LSTM) [31], [54], CNN and Bidirectional LTSTM [39], GCNN + GRNN [34] and Multidimensional LSTM-CNN [37] can be given as examples to these models.

In this study, we develop an artificial neural network-based model for HTR and then test its performance on IAM word dataset. Our model consists of a several CNN layers stacked on top of each other which is then followed by a multi-layer Bidirectional LSTM. This architecture allows us to extract useful features from images and process the character sequence effectively by considering both forward and backward information. We use the Connectionist Temporal Classification (CTC) operation to determine the best match between predicted and target output, which helped compute the loss function. Overall, our approach was successful in achieving the desired task.

II. Literature Review

Handwritten Text Recognition (HTR) is a well-studied problem in artificial intelligence, with several approaches developed for digitizing handwritten text. Two main categories of methods used for HTR are Hidden Markov Models (HMM) and Artificial Neural Network-based models.

HMMs utilize probabilities to analyze and describe sequential data in various domains such as bioinformatics, music, DNA sequences, video, speech, and text. HMM can be divided into two main components: the hidden state sequence and the observable sequence. Researchers like Bunke et al. [9], Bishop [5], Plötz & Fink [38], Doetsch et al. [13], and Toselli et al. [51] have found HMM to be a valuable tool for analyzing and modeling sequential data. However, Espana et al. [15] and Gemello et al. [59] explored a hybrid approach of HMM with Artificial Neural Networks (ANN) to address issues such as removing slope and slant from handwritten text lines and normalizing text image size. A weakness of HMM in handwritten text recognition is its assumption that the current state depends solely on the previous state, limiting its ability to sequence long text or consider contextual information.

Another commonly used model for HTR is the Convolutional Neural Network (CNN), with particular attention given to the Residual Network (ResNet) architecture. ResNet-18, as used by Singh and Karayev [47], and X Chi et al. [60], addresses the problem of vanishing gradients and improves training performance for very deep networks. ResNet-18 comprises several residual blocks, with each block consisting of two convolutional layers and a shortcut connection. The network gradually reduces spatial dimensions, increases the number of channels, and produces final predictions through a global average pooling layer and a fully connected layer. ResNet also includes other variants such as ResNet-34, ResNet-50, ResNet-101, and ResNet-152, with increased depth and complexity requiring more computational resources. ResNet is commonly employed for languages with alphabets different from English, as demonstrated by Jayakanthan et al. ([61] in Tamil language character recognition and Mhapsekar et al. [62] in Devanagari language character recognition. ResNet-50 has also been utilized in Chinese language character recognition by Chen et al. [63] and Zhang et al. [65].

CNN, along with various types of Recurrent Neural Networks (RNN), is a common architecture for Handwritten Text Recognition (HTR). Typically, researchers utilize RNN layers after several CNN layers (around 3-5 layers) to process sequential data. Both RNN and Hidden Markov Models (HMM) share the idea that the previous state helps predict the current state. However, RNN has the advantage of longer sequence "memory." A specific type of RNN called Long Short-Term Memory (LSTM) is particularly effective in processing longer sequential data. Researchers such as Balci et al. [3], Wigington et al. [54], and Yugandhar Manchala [31] have shown a preference for the combination of CNN followed by LSTM for HTR.

Several researchers have also employed CNN followed by Bidirectional Long Short-Term Memory (BiLSTM), which incorporates information from both past and future contexts. Examples include [39] and [34]. Another variant of LSTM is Multidimensional Long Short-Term Memory (MD-LSTMs), which processes multidimensional data such as images and videos, incorporating LSTM cells for each dimension. Pham et al. [37] utilized the CNN-MDLSTM model and achieved good results.

III. METHODOLOGY

In this section, the architectural details of the proposed deep learning model are discussed. In the first section, we discuss CNNs briefly. Then, we discuss RNNs in the second section.

A. Convolutional Neural Networks (CNN)

CNNs are capable of detecting simpler or more complicated patterns, such as lines, curves, characters, words, faces, and objects [53] in images. An example CNN architecture consists of several layers as shown in Figure 2. As in [44], these layers can be briefly explained as follows:

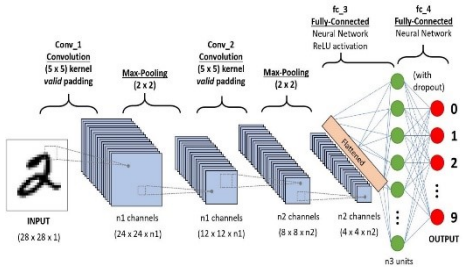


Fig. 2. An example of Convolutional Neural Network architecture [26].

Convolutional layer is the first layer in which several $m \times m$ filters are convoluted over an image. Each filter provides an activation map of size depending on some parameters like the filter size, the padding method, stride value.

In order to achieve nonlinearity, an activation function is applied to the out out of each neuron. Rectified Linear Unit (RELU) is the most common CNN activation functions that

keeps the positive values and replaces the negative ones with zeros [27].

Pooling layer is placed after the convolutional layer in order to replace the previous layer's output at specific locations by deriving a summary statistic of the nearby outputs, reducing image size by focusing more on data details. Two types of pooling functions such as max-pooling and average pooling are used in CNNs. Last layer in a CNN is usually a fully connected layer in which the neurons have full connectivity with all neurons in the preceding and succeeding layers.

B. Recurrent Neural Networks (RNNs)

RNNs are specialized classes of aANNs which process sequential data such as information from speech or text [19]. They produce an output based on all previous information of a sequence. That is, the prediction at a particular time uses information from all earlier inputs in the sequence, not only the last input. They aim to learn from the current input along with what they have learned from the inputs they received previously. The output of the RNNs determine a probability of the subsequent input based on the probability of the previous input. Over time, the network is updated to produce more accurate results.

RNNs suffer from vanishing gradients problem which makes them inefficient for large sequences. In order to overcome this problem, a special type of RNNs have been proposed. Long Short Term Memory (LSTM) is an extension of RNN which has a memory that is especially important for learning from previous experiences over long distances [45]. The three gates in LSTM, orget gate, input gate, and output gate, affect the information flow in a given block.

There are also Bi-directional LSTM (BiLSTM) cells that basically have two LSTM cells working in opposite directions, forward and backward. Actually, the concept of Bi-directional RNNs were introduced earlier in [46] in order to produce an output based on information from both the past and the future. When using bidirectional, inputs will be processed in two directions: one from the present to the future and the other from the future to the present. This method differs from unidirectional in that information from the future is preserved in the LSTM that runs backward. Combining the two hidden states allows for preserving data from the present and future at any time. An example of Bidirectional RNN architecture is shown in Figure 3. The figure also illustrates the main working mechnanism in LSTM or another RNN unit, Gated Recurrent Unit (GRU).

C. Gradient Descent Optimization Algorithm

Optimization algorithms manipulate the learning parameters to control the cost function. By adjusting the weights and biases based on the updated learning parameters, the cost function of the neural network can either decrease or increase. The goal of gradient descent is to converge to the global optimum by taking small steps towards the target through multiple iterations. Gradient descent can be categorized into

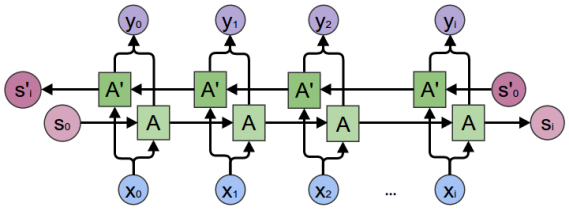


Fig. 3. Bidirectional Recurrent Neural Networks architecture.Mehdi, Muhammad Sarim. (2020). [65]

three types: batch gradient descent, stochastic gradient descent, and mini-batch gradient descent. Batch gradient descent processes the entire training set at once, while mini-batch gradient descent handles a smaller batch of training examples. Gradient descent incorporates various optimizers, such as momentum [66] and adaptive moment estimation (Adam) [67]. Momentum in SGD enables smaller oscillations in the vertical direction and faster movement in the horizontal direction towards the global optimum. Adaptive moment estimation (Adam) combines the momentum and RMsprop algorithms, offering a more advanced optimization technique.

D. Connectionist Temporal Classification (CTC)

CTC is used for the tasks such as handwritten text or speech recognition. In these tasks, each input is different from person to person and from time to time, so the amount of space for any character in any handwritten text does not seem the same or equal. The CTC algorithm takes the probability distribution over label sequences and finds the most suitable label for the input sequence, which is given in order to compute the loss function between the ground truth text and the output from the model [17]. This algorithm uses the blank label to separate between the same characters found in a row and the existing label in any frame, meaning there is no character there. The alignments allowed by CTC are the same length as the input. CTC allows any alignment which maps to output after merging repeats and removing blank symbols. The work of this algorithm is to go from probabilities at each time step of the input sequence to the probability of an output sequence by finding the most probable alignments which can be mapped correctly to the given input.

The mechanism of this algorithm is as follows: Each time step shows the probability of a character at its position (frame). A path is a sequence of frames with the characters' probabilities in their corresponding frame [44]. Each label or input from the earlier stage has multiple paths, which are defined by a Finite State Machine (FSM). The FSM can create several valid paths that map to a specific labeling, such as l =" ab," as shown in Figure 4, by putting a blank label at the beginning and end frame or between characters. The probability of a path is given by multiplying the probability of each character in its frame. The function collapse maps a path to labeling by removing the blank label and dropping the duplicated characters. Finally, the probability of a labeling l is given by summing all the valid paths that map to this labeling.

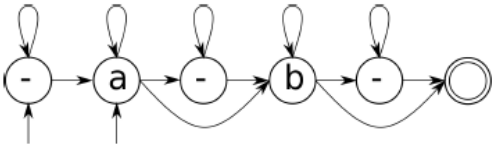


Fig. 4. An example of the FSM [44].

Examples of paths for a text using CTC: “to” ‘ -t-o—’, ‘ ttttttttt-ooo-’, ‘to’, … “hello” ‘h-elllll-ll-ooo’, ‘hel-lo’, …. “a” ‘aa’, ‘a-’, ‘-a’, …

IV. IMPLEMENTATION

In this section, we discuss the implementation details of the proposed algorithm. In this study, we used different software packages like Keras and Pandas. we trained the proposed model on Google Collab using python language.

A. Dataset

The IAM Dataset [52] is the most commonly used dataset for recognizing handwritten text. It consists of 1,539 pages with 13,353 lines and 115,320 words written by 657 authors, showcasing various writing styles. The dataset includes forms with a header displaying a digitally created text to be replicated, a lower section where writers rewrite the text in their handwriting, and a signature at the bottom. We have used words in this dataset. We split the dataset into training, validation, and testing subsets. 90% of the samples were used for training, 5% for validation, and the remaining 5% for testing. This provides distinct sets for validation and testing in addition to ensuring that we have enough data for model training. Additionally, we built a character vocabulary using Keras’ StringLookup layer to process labels at the character level. We resize the images 128 $\times$ 32 while maintaining the aspect ratio in order to to avoid distortion. We finally normalize the pixel values by converting the image data type to tf.float32 and dividing it by 255.0, which scales the pixel values between 0 and 1. This assumes that the pixel values in the input images are in the range of 0 to 255.

B. Proposed Model Architecture

We have used the model architecture illustrated in Figure 5. We employed a Convolutional Neural Network (CNN) as the first part. It had five layers, with each layer applying a filter kernel of size 3 $\times$ 3 to the input image. The number of filters increased in each layer, and max pooling was applied to summarize regions and reduce dimensionality. ReLU function and Batch Normalization were used to ensure non-linearity and normalize outputs, respectively. The output of the CNN block was a 23$\times$265 feature map.

The second part of the architecture consisted of three hidden layers of Bidirectional Long Short-Term Memory (BiLSTM) to capture sequence features. Each layer had a different number of hidden units, and both forward and backward passes were utilized. The output of this block was a 32$\times$64 probability distribution for each partition in the input image.

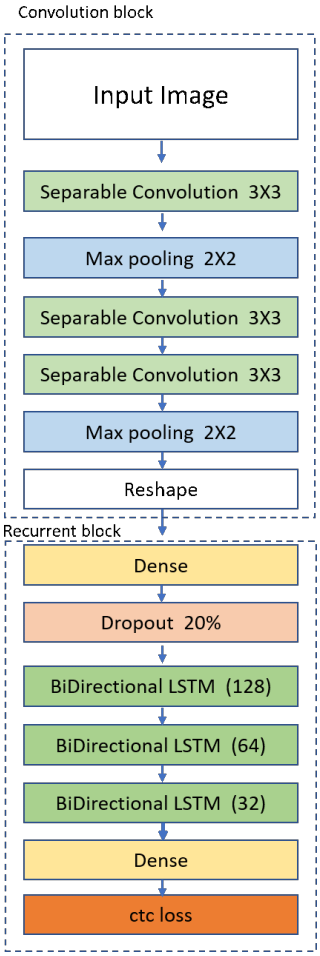


Fig. 5. The architecture of the proposed model.

Finally, we used a dilated convolution with a filter kernel of size 1$\times$1 and 80 filters to map the output sequence to 80 characters, including a blank character for the CTC (Connectionist Temporal Classification) task. The architectural details of the complete model is given in Table I.

C. Evaluation Criteria

HTR models are usually evaluated using both Character Error Rate (CER) and Word Error Rate (WER) criterion. These are used to measure the accuracy of the HTR system and are calculated by the Lowenstein Distance (Levenshtein, 1966) [29] between the ground truth and the predicted value produced by the model. To calculate CER, the ground truth text is taken and for the generated text, the number of character level modifications including subtraction, removal or deletion needed to achieve the exact desire word is counted. WER is calculated similarly, but it is applied on the word level instead of the character level. Accuracy on the other hand is the percentage of correctly recognized characters, words, or sentences.

The number of words in the training and test set is given in Table II. We have created three different dataset configurations,

TABLE I
ARCHITECTURAL DETAILS OF THE PROPOSED MODEL

Type	Description	Output size
Input	Grey value of word image	128 x 32 x 1
Conv1 + Pool	Kernel 3X3, Pool 2 X 2	128 x 32 x 32
Conv2	Kernel 3 X3	64 x 16 x 64
Conv3 + Pool	Kernel 3 X3, Pool 2 X 2	32 x 8 x 64
Collapse	Reshape	(32 x 512)
Dense1	Dense (fully connected) layer with 64 units	(None, 32, 64)
Dropout	Dropout layer	(None, 32, 256)
bidirectional	Bidirectional LSTM layer with 128 units	(None, 32, 128)
bidirectional_1	Bidirectional LSTM layer with 64 units	(None, 32, 64)
dense2	Dense (fully connected) layer with num_classes+1 units	(None, 32, 79)

TABLE II
DISTRIBUTION OF THE WORDS IN TRAIN AND TEST SPLITS

Training	Validation	Total	Percentage
109,554	5,766	115,320	95% Train, 5% Test
39,984	9,997	50,000	80% Train, 20% Test
7,985	1,997	10,000	80% Train, 20% Test

each of which having different number of words in training and test sets.

V. RESULTS AND DISCUSSION

In this section, we present the experimental results obtained by the proposed model which contains CNN layers followed by Bi-LSTM layers. Table III shows the test results obtaned on different dataset configurations. The table also illustrates the hyper-parameters like the batch-size and the optimization algorithm used in the proposed model. The first configuration is comprised of the whole IAM word dataset of 115,320 word. Adam optimizer with a learning rate (lr) of 0.001 resulted in a CER of 10.72% and a WER of 26.45%.

For the second dataset configuration which contains 50,000 word images (half of the previous one), we used RMSprop optimizer with a lr of 0.0003. The CER and WER values were 15.12% and 35.76%, respectively. Finally, the third configuration uses roughly 10% of all the available words. We used the RMSProp optimizer and experimented with different learning rates ranging from 0.0001 to 0.0003. The model achieves 22.95% and 45.87%for CER and WER, respectively.

The results suggest that the proposed model is able to achieve low error rates especially under high data regime, as the model benefits from a more diverse and extensive dataset. Furthermore, smaller batch sizes contribute to better performance by allowing for more frequent updates and fine-tuning of the model. Higher learning rates, specifically 0.001,

TABLE III
CER AND WER RESULTS AFTER 95 EPOCHS FOR DIFFERENT DATASET CONFIGURATIONS AND WITH DIFFERENT MODEL HYPER-PARAMETERS

Training size	Batch-size	lR	Optimizer	CER%	WER%
115,320	100	0.001	Adam	10.72	26.45
50,000	30	0.0003	RMSProp	22.95	45.87
10,000	30	0.0001	RMSProp	14.5	34.9

TABLE IV
VALIDATION RESULTS OF DIFFERENT MODELS ON IAM DATASET.

Model	CER%	WER%
CRNN + CTC	10.7 %	26.4%
MDLSTM-RNN [37]	10.4%	10.1%
MDLSTM-RNN (No Language Model) [6]	10.1%	36.5%
FPHTR Resnet18 [47]	28.9%	38%
FPHTR Resnet18 + Synth. Aug. [47]	9.0%	16.5%
FPHTR Resnet34 + Synth. Aug. [47]	8.8%	14.0%

result in lower error rates, indicating the importance of a well-adjusted learning rate that enables faster convergence. Finally, the choice of optimizer plays a role, with the Adam optimizer outperforming RMSProp in terms of both CER and WER. Overall, these results highlight the significance of dataset size, batch size, learning rate, and optimizer selection in achieving accurate and reliable handwriting recognition.

We conducted a comprehensive performance comparison of our proposed model with other models mentioned in the literature. The results of these models, along with our proposed model, are presented in Table IV. It is important to note that all models were evaluated on the IAM words dataset, except for FPHTR ResNet18 and ResNet18 + Synth. Aug., which utilized paragraph-based images, while ResNet34 employed line-based images.

When considering the Character Error Rate (CER), our model demonstrates a slight improvement over the MDLSTM-RNN model [37], although it falls short compared to the MDLSTM-RNN model without a language model [6]. However, in terms of Word Error Rate (WER), our model surpasses both models.

It is noteworthy that our model, similar to ResNet [47], utilizes Convolutional Neural Networks (CNN). However, our model achieves superior performance compared to ResNet. In a related study by Sumeet et al. [47], synthetic augmentation was employed to demonstrate the impact of data size on performance. Their findings highlight that larger training data leads to better performance. Consequently, their results using ResNet18 and ResNet34 + Synth. Aug. outperform other models in terms of CER.

A. Conclusion

Offline Handwritten Text Recognition (HTR) has been drawing attention of many researchers over the years due to its importance. For example, employees of a postal office

need to recognize the handwritten texts and then to manually type the text on the computer quickly. These processes make the employee struggle and waste time. This situation does not only occur in postal offices but also in many areas such as recognizing text in historical documents. In this paper, we focus on HTR in English language. We build a hybrid deep learning model which contains two stages. The first stage includes three layers of convolution layers. The second stage involves three layers of bidirectional long-short-term memory followed by dilated convolutional layer for projecting the dimension into the target dimension. We evaluate our two-stage model on the IAM dataset using CER and WER evaluation criteria. The experimental result shows that the proposed model yields in satisfactory results with low error rates when compared to the other complex models in the literature.

While HTR has been explored for languages like English, Bangla, Arabic, and Chinese, the Turkish language remains relatively unexplored in this context. The future work aims to enhance the HTR model by incorporating Turkish word recognition and making the hybrid model language-independent. This will involve modifying the model's structure and model hyper-parameters to achieve accurate transcription of Turkish characters and enable recognition of complete text instead of single words. Overall, this research paper highlights the importance of HTR and presents a hybrid deep learning model with promising results. The future work aims to expand the model's capabilities to recognize Turkish text and optimize its performance through different models. Moreover, we plan to develop composite models that will be applied not only to words or sentences but to the whole page images in order to perform line and word segmentation and text recognition simultaneously.

REFERENCES

[1] Ahlawat, S., Choudhary, A., Nayyar, A., Singh, S., & Yoon, B. (2020). Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN). Sensors., 20(12), 3344.

[2] Ali, A., & Mallaiah, S. (2021). Intelligent handwritten recognition using hybrid CNN architectures based-SVM classifier with dropout. . Journal of King Saud University – Computer and Information Sciences

[3] Balci, B., Saadati, D., & Shiferaw, D. (2017). Handwritten text recognition using deep learning. CS231n: Convolutional Neural Networks for Visual Recognition. Stanford University, Course Project Report, Spring, pp. 752–759.

[4] Bartlett, P., Hazan, E., & Rakhlin, A. (2008). Adaptive online gradient descent. In Proceedings of the NIPS, Vancouver, BC, Canada.

[5] Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.

[6] Bluche, T. (2016). Joint Line Segmentation and Transcription for End-to-End Handwritten Paragraph Recognition. 29th Conference on Neural Information Processing Systems.

[7] Bluche, T., & Messina, R. (2017). Gated convolutional recurrent neural networks for multilingual handwriting recognition. 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), 646–651.

[8] Bluche, T., Louradour, J., Knibbe, M., Moysset, B., Benzeghiba, F., & Kermorvant, C. (2014). The A2iA Arabic handwritten text recognition system at the OpenHaRT2013 evaluation. in International Workshop on Document Analysis Systems (DAS).

[9] Bunke, H., Roth, M., & Schukat-Talamazzini, E. G. (1995). Offline cursive handwriting recognition using hidden Markov models. Pattern Recognition, 28, 1399–1413.

[10] Cho, K., van Merrienboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches—arXiv preprint arXiv:1409.1259.

[11] Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

[12] Cui, H., & Bai, J. (2019). A new hyperparameters optimization method for convolutional neural networks. . Pattern Recognition. Letter., 828–834.

[13] Doetsch, P., Kozielski, M., & H., N. (2014). Fast and robust training of recurrent neural networks for offline handwriting recognition. Proceedings of International Conference on Frontiers in Handwriting Recognition, ICFHR., 279–284.

[14] E. Grosicki and H. El-Abed, "ICDAR 2011 - French Handwriting Recognition Competition," 2011 International Conference on Document Analysis and Recognition, Beijing, China, 2011, pp. 1459-1463, doi: 10.1109/ICDAR.2011.290.

[15] Espana-Boquera, S., Castro-Bleda, M. J., Gorbe-Moya, J., & Zamora-Martinez, F.(2010). Improving offline handwritten text recognition with hybrid HMM/ANN models. IEEE transactions on pattern analysis and machine intelligence, 33(4), 767-779.

[16] Florian Kleber, Stefan Fiel, Markus Diem and Robert Sablatnig, CVL-Database: An Off-line Database for Writer Retrieval, Writer Identification and Word Spotting, In Proc. of the 12th Int. Conference on Document Analysis and Recognition (ICDAR) 2013, pp. 560-564, 2013. pdf doi

[17] Graves, A. (2012). Supervised Sequence Labeling with Recurrent Neural Networks. Springer.

[18] Graves, A., Fernández, S., & Gomez, F. (2006). Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. International Conference on Machine Learning, 369–376.

[19] Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2015). LSTM: A Search Space Odyssey. IEEE Transactions on Neural Networks and Learning Systems, 2222 – 2232.

[20] Haykin, S. (2009). Neural networks and learning machines. New York: Prentice Hall/Pearson.

[21] Hwang, K., & Sung, W. (2016). Character-level incremental speech recognition with recurrent neural networks. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 5335,5339.

[22] Ingole, M. M., & Tighare, K. (2021). Neural Network Based Handwritten Character Recognition. International Journal of Scientific Research in Science and Technology.

[23] Ioffe, S. (2012). Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. in Proceedings of the 31st International Conference on Neural Information Processing Systems, 1942–1950.

[24] Jarrett, K., Kavukcuoglu, K., Ranzato, M., & LeCun, Y. (2009). What is the best multi-stage architecture for object recognition? . In Proceedings of the IEEE 12th International Conference on Computer Vision (ICCV), Kyoto, Japan.

[25] Joshi, R., & Shah, D. D. (2021). Approach to Avoid Resource Exhaustion Caused by Editing Tools for Automating Effects Using Noise Inducing Procedures in Deep Learning. International Journal of Intelligent Communication, Computing and Networks.

[26] Kamalanaban, E., Gopinath, M., & Premkumar, S. (2018). Medicine box: Doctor's prescription recognition using deep machine learning. International Journal of Engineering and Technology(UAE)., 7, 114–117.

[27] Kashif, M. (2021). Urdu Handwritten Text Recognition Using ResNet18. . arXiv preprint arXiv:2103.05105.

[28] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. Journal: Communications of the ACM.

[29] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady.

[30] Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. in Proc. ICML, 30(1).

[31] Manchala, S. Y., Kinthali, J., Kotha, K., Kumar, J. J. K. S., & Jayalaxmi, J. (2020). Handwritten text recognition using deep learning with Tensorflow. International Journal of Engineering and Technical Research, 9(5).

[32] Manmatha, R., & Srimal, N. (2002). Scale Space Technique for Word Segmentation in Handwritten Documents. International Conference on Scale-Space Theories in Computer Vision,, pp. 22–33.

[33] Marti, U.-V., & Bunke, H. (2002). The iam-database: An English sentence database for offline handwriting recognition. International Journal on Document Analysis and Recognition. , 39–46.
[34] Neto, D. S., Flor, A. B., D., B. A., & Baptista, E. B. (2020). HTR-Flor: A Deep Learning System for Offline Handwritten Text Recognition. IEEE 2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI) - Recife/Porto
[35] Nirmalasari, D. A., Suciati, N., & Navastara, D. A. (2021). Handwritten Text Recognition using Fully Convolutional Network. In IOP Conference Series: Materials Science and Engineering, 1077(1), 012030.
[36] Nurseitov, D., Bostanbekov, K., Kanatov, M., Alimova, A., Abdallah, A., & Abdimanap, G. (2021). Classification of Handwritten Names of Cities and Handwritten Text Recognition using Various Deep Learning Models. arXiv preprint arXiv:2102.04816.
[37] Pham, V., Bluche, T., Kermorvant, C., & Louradour, J. (2014). Dropout improves recurrent neural networks for handwriting recognition. In Proceedings of the 14th Int. Conf. on Frontiers in Handwriting Recognition, Heraklion, Greece.
[38] Plötz, T., & Fink, G. A. (2009). Markov models for offline handwriting recognition: a survey. International Journal on Document Analysis and Recognition (IJDAR)., 12(4), 269-298.
[39] Puigcerver, J. (2017). Are multidimensional recurrent layers really necessary for handwritten text recognition? 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), 67–72.
[40] Rabby, A. S., Haque, S., Shahinoor, S. A., Abujar, S., & Hossain, S. A. (2018). A universal way to collect and process handwritten data for any language. Procedia computer science, 143, 502-509.
[41] Ryu, J., Koo, H. I., & Cho, N. I. (2014). Language-independent text-line extraction algorithm for handwritten documents. IEEE Signal processing letters, 21(9), 1115-1119.
[42] S´anchez, J.-A., Romero, V., Toselli, H. A., & Vidal, E. (2016). Icfhr2016 competition on handwritten text recognition on the read dataset. 15th International Conference on Frontiers in Handwriting Recognition (ICFHR), 630–635.
[43] Sanasam, I., Choudhary, P., & Singh, K. M. (2020). Line and word segmentation of handwritten text document by mid-point detection and gap trailing. Multimedia Tools and Applications, 79(41), 30135-30150.
[44] Scheidl, H. (2018). Handwritten text recognition in historical documents.
[45] Schmidhuber, J., & Hochreiter, S. (1997). Long Short-Term Memory. Neural Computation, 1735–1780.
[46] Schuster, M., & Paliwal, K. K. (1997). Bidirectional Recurrent Neural Networks. IEEE TRANSACTIONS ON SIGNAL PROCESSING, 45(11).
[47] Singh, S. S., & Karayev, S. (2021). Full Page Handwriting Recognition via Image to Sequence Extraction. . arXiv preprint arXiv:2103.06450.
[48] Surinta, O., Holtkamp, M., Karabaa, F., Van Oosten, J. P., Schomaker, L., & Wiering, M. (2014). A path planning for line segmentation of handwritten documents. . In 2014 14th International Conference on Frontiers in Handwriting Recognition. , 175-180.
[49] Tappert, C. C., & Cha., S.-H. (2007). English Language Handwriting Recognition Interfaces. Pace University, New York, NY, USA.
[50] Tieleman, T., & Hinton, G. (2012). Lecture 6.5–rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.
[51] Toselli, A. H., & Vidal, E. (2015). Handwritten text recognition results on the Bentham collection with improved classical n-gram-hmm methods. In Proceedings of the 3rd International Workshop on Historical Document Imaging and Processing (HIP@ICDAR).
[52] U. Marti and H. Bunke. The IAM-database: An English Sentence Database for Off-line Handwriting Recognition. Int. Journal on Document Analysis and Recognition, Volume 5, pages 39 - 46, 2002.
[53] Venkatesan, R., & Li, B. (2018). Convolutional Neural Networks in Visual Computing. A Concise Guide, CRC Press.
[54] Wigington, C., Tensmeyer, C., Davis, B., Barrett, W., Price, B., & Cohen, S. (2018). Start, follow, read End-to-end full-page handwriting recognition. In Proceedings of the European Conference on Computer Vision (ECCV), 367-383.
[55] Yani, M., Irawan, B., S, S., M.T., Setiningsih, C., S.T., & M.T. (2019). Application of Transfer Learning Using Convolutional Neural Network Method for Early Detection of Terry's Nail. Journal of Physics Conference Series.
[56] Zhao, J. (2021). In silico design of small molecular libraries via Reinforcement learning. Uppsala University, Disciplinary Domain of Medicine and Pharmacy, Faculty of Pharmacy, Department of Pharmaceutical Biosciences.
[57] Jiaxi, Z. (2021). In silico design of small molecular libraries via Reinforcement learning.
[58] Zinkevich, Weimer, M., Smola, M., & Li, L. (2010). Parallelized stochastic gradient descent. . NIPS, 2595–2603.
[59] Gemello, Roberto & Mana, Franco & Loquendo, Dario. (2008). Hybrid HMM/Neural Network based Speech Recognition in Loquendo ASR.
[60] X. Chi, S. Huang and J. Li, "Handwriting Recognition Based on Resnet-18," 2021 2nd International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE), Zhuhai, China, 2021, pp. 456-459, doi: 10.1109/ICBASE53849.2021.00091.
[61] Jayakanthan, R., Kumar, A. H., Sankarram, N., Charulatha, B. S., & Ramesh, A. (2020). Handwritten tamil character recognition using ResNet. International Journal of Research in Engineering, Science and Management, 3(3), 133-137.
[62] Mhapsekar, M., Mhapsekar, P., Mhatre, A., & Sawant, V. (2020). Implementation of residual network (ResNet) for devanagari handwritten character recognition. In Advanced Computing Technologies and Applications: Proceedings of 2nd International Conference on Advanced Computing Technologies and Applications—ICACTA 2020 (pp. 137-148). Springer Singapore.
[63] Chen, L., Peng, L., Yao, G., Liu, C., & Zhang, X. (2019, September). A modified inception-ResNet network with discriminant weighting loss for handwritten chinese character recognition. In 2019 International Conference on Document Analysis and Recognition (ICDAR) (pp. 1220-1225). IEEE.
[64] Zhang, R., Wang, Q., Lu, Y. (2017, November). Combination of ResNet and center loss based metric learning for handwritten Chinese character recognition. In 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR) (Vol. 5, pp. 25-29). IEEE.
[65] Mehdi, Muhammad Sarim. (2020). Trajectory Prediction for ADAS. 10.13140/RG.2.2.11891.35369.
[66] Jiao, M., Wang, D., Qiu, J. (2020). A GRU-RNN based momentum optimized algorithm for SOC estimation. Journal of Power Sources, 459, 228051.
[67] Jais, I. K. M., Ismail, A. R., Nisa, S. Q. (2019). Adam optimization algorithm for wide and deep neural network. Knowledge Engineering and Data Science, 2(1), 41-46.