Curriculum Learning for Printed Text Line Recognition of Ligature-based Scripts

Adnan Ul-Hasan*, Faisal Shafait† and Marcus Liwicki*
*Department of Computer Science, University of Kaiserslautern, Kaiserslautern, Germany.
Email: adnan@cs.uni-kl.de, marcus.liwicki@dfki.de
†NUST School of Electrical Engineering and Computer Science, Islamabad, Pakistan.
Email: faisal.shafait@seecs.edu.pk

***Abstract*—This paper introduces a novel curriculum learning strategy for ligature-based scripts. Long Short-Term Memory Networks require thousands or even millions of iterations on target symbols, depending upon the complexity of the target data, to converge when trained for sequence transcription because they have to localize the individual symbols along with the recognition. Curriculum learning reduces the number of target symbols to be visited before the network converges. In this paper, we propose a ligature-based complexity measure to define the sampling order of the training data. Experiments performed on UPTI database show that the curriculum learning using our strategy can reduce the total number of target symbols before convergence for printed Urdu Nastaleeq OCR task.**

***Index Terms*—Curriculum learning, Urdu Nastaleeq OCR, UPTI database, LSTM Networks**

I. Introduction

Recurrent Neural Networks (RNN) are becoming the standard in the field of Optical Character Recognition (OCR). In the last decade, various RNN architectures, internal working details, and application areas have been published. Due to their powerful context-aware processing, they have successfully been used as sequence learning machines for transcription and other document analysis tasks [1], [2], [3].

However, the high performance of RNN is achieved on the cost of complex computations and slow convergence times. Louradour and Kermorvant [4] nicely described two main reasons for such behaviours for OCR tasks: One is the architectural issues (exploding and vanishing gradient problems), and second is the requirement for using RNNs as sequence learners instead of single character learners.

The first problem has been addressed by using a modified computation scheme, termed as Long Short-Term Memory (LSTM) [5] where multiplicative gates (similar to computer memory cells with read, write and refresh functions) replace the traditionally used activation units of Neural networks. This scheme has been used to overcome the vanishing gradient problem; thus making RNNs trainable using standard back-propagation.

The second issue has been resolved by using a Connectionist Temporal Classification (CTC) [6] layer at the output of standard LSTM network. This layer allows direct sequence learning without the need to segment text-line images into corresponding words or characters. But using this scheme implies that the network has to learn the location of individual characters in addition to classify them. This, in turn, increases the process's complexity and training takes many iterations until convergence. There are many efforts reported in the literature to speed-up the training process. Broadly, such efforts can be divided into two main categories. One is to improve RNN/LSTM architecture or to develop parallel architectures [7]. Second is to develop methods to efficiently train RNNs/LSTM networks [8], [9].

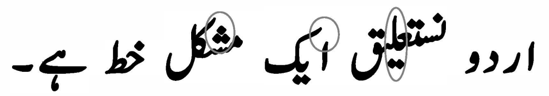
Curriculum Learning is one such solution to train neural networks efficiently. The concept behind curriculum learning is not new: Skinner [10] coined the term "shaping" in context of gradual learning in humans and other animals. Elman [11] suggested that starting with small steps and following a predefined schedule could help in the efficient training of artificial neural networks. The basic idea is that the gradual increase in the number of concepts and complexity of concepts will speed-up the learning process. Bengio et al. [12] showed that the use of curriculum learning strategies improve generalization and that the networks converge faster. It should be noted that there is no single curriculum strategy that applies to every case. On the contrary, these strategies vary from one application domain to another; e.g., for images, noise could be considered as the *easiness* criteria; for text, the number of words in a text-line could be the criteria for *easiness*, and so on and so forth.

The main contribution of this paper is that we propose a novel curriculum learning strategy for Nastaleeq-like cursive scripts that have thousands of ligatures[1]. We introduce a complexity score and start with the simple ligatures, gradually increasing the complexity during the training. We report the results of applying curriculum learning techniques to printed Urdu Nastaleeq OCR using 1D-LSTM networks. Please refer to Section II and Section III for further details.

A. Related Work

There are many applications reported in the literature where the idea behind curriculum learning (starting with small concepts and gradually increasing the complexity) has been applied. Bengio et al. [12] applied this approach to shape recognition and language modelling. In the shape-recognition

[1]Ligature is defined as the glyph formed by connecting different characters in Urdu language.



اردو نستعلیق ایک مشکل خط ہے۔

Fig. 1. Urdu Nastaleeq script is read from right-to-left. When writing, numbers are written and read from left-to-right, otherwise, the script is written from right-to-left. The gray circles show the problem of *kerning*, where part of neighbouring character overlaps an individual character.

experiments, they found that using curriculum yields better generalization than no-curriculum settings. For the language modelling experiments, they started training with $5,000$ most frequent words, and then with each pass, they added $5,000$ more samples.

Elman [11] applied this concept on grammar prediction. They first train the network with $10,000$ simple sentences. After 5 epochs of training, they remove this dataset and introduce a new dataset in which 25% sentences are complex and rest are simple. This process continues until all examples of complex sentences are used. This network mastered the complex set of sentences completely, whereas it failed to do so when presented with complex examples from the very beginning.

Louradour and Kermorvant [4] used curriculum learning for handwriting recognition task. They used the length of a text line as a criterion for easyness and they defined probability distribution based on this criterion. Then they draw the text-lines based on their probability according to a certain schedule. They reported that for some databases, generalization significantly improved. However, their experiments showed better convergence for all databases.

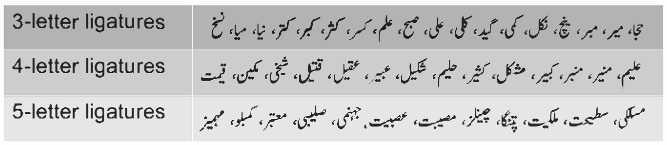
This paper is further organized as follow. In the next Section, details about our novel curriculum learning strategy are given. Section III describes the experimental evaluation. Experimental results and analysis are described in Section IV and Section V concludes the paper with future directions.

II. Curriculum Learning for Printed Urdu Nastaleeq Script

A. Properties of Nastaleeq Script

Nastaleeq is a cursive script even in the printed form. It is the main script of writing in large parts of Pakistan, India, Afghanistan and Iran, and is known to a population of over 200 million people. There have been very little efforts found [13], [14], [15], [16] to digitize documents in Nastaleeq albeit there exists a rich heritage of literature in the aforementioned countries. One main reason for unavailability of standard approaches for Urdu Nastaleeq OCR is the complexity of the script itself. Few challenges that this script poses are overlapping ligatures (non-trivial segmentation), variability of shapes of characters depending upon their position in the word/ligature (the shape in a particular position also depends upon connecting characters) and huge number of unique ligature. For more details about the OCR challenges presented by Urdu Nastaleeq script, please see [14]. An example of Urdu Nastaleeq script is shown in Fig. 1.

Since, Urdu Nastaleeq is a cursive script, a simple curriculum strategy of using number of characters in a text-line (as



3-letter ligatures	حجا، میر، مبر، پنچ، نکل، کمی، گید، کلی، علی، صبح، علم، کسر، کثر، کبر، کتر، نیا، میا، نخ
4-letter ligatures	علیم، منیر، منبر، کبیر، مشکل، کثیر، حلیم، شکیل، عبیر، عقیل، قتیل، شیخی، مکین، قیمت
5-letter ligatures	مسلکی، سطحیت، ملکیت، پتنگا، چینلز، مصیبت، عصبیت، جہنمی، صلیبی، معتبر، کمبلو، مہمیز

Fig. 2. Examples of 3-, 4- and 5-letter ligatures of Urdu Nastaleeq script. Note that increase in the number of characters in a ligature increases the ligature's shape complexity. The main body in most cases shifts upwards and right.

done in [4]) as the measure of its simplicity (or complexity) may not be useful. We claim that a more useful strategy could be to use number of ligatures and number of characters in a ligature as the measure of simplicity/complexity. There are 24k ligatures [17] in Urdu and they can comprise up to 13 characters, though ligatures consisting of more than seven characters are uncommon. Mostly, they consist of three, four or five characters. In Nastaleeq script, as the number of characters increase in a ligature, its shape tends to shift upward right (diagonally) and the overlapping between ligature's characters also increases. Some examples of ligatures with different numbers of characters are shown in Fig. 2.

B. Curriculum Learning Approach

In the curriculum learning setting, the idea is to start with less and simpler concepts and then gradually increase both number and complexity of concepts. Starting with simpler ligatures (those consist of less number of characters) will allow the LSTM network to avoid dealing with complex shapes at start. In case of text-line's length as simplicity/complexity measure, complex ligatures (those consist of many characters) will be present from the start and then LSTM network has to deal with them along with simpler shapes.

Keeping the above-mentioned points in mind, we have devised a novel curriculum to speed-up convergence in case of Urdu Nastaleeq OCR. We defined the complexity of a ligature in term of *complexity score* (CS_t) of a training text-line as defined in Equation 1.

$$CS_t = a_t + b_t^2 + c_t^3 \tag{1}$$

where,

$a_t \rightarrow$ number of ligatures consisting of single letter (isolated charactrers),
$b_t \rightarrow$ number of ligatures consisting of two letters,
$c_t \rightarrow$ number of ligatures consisting three or more characters respectively.

This score is calculated on the transcription of a training text-line image. The complexity score (CS_t) is then reciprocated to compute the *simplicity* of a text-line.

$$simplicity_t \in [0,1] = \frac{1}{CS_t} \tag{2}$$

This simplicity value is then used to draw a particular text-line from a probability distribution (given by Equation 1 in [4]). Similar to that equation, we parametrized the probability distribution with λ as follows:

$$P_\lambda = \frac{1}{N_\lambda}(simplicity_t)^\lambda \tag{3}$$

TABLE I
NO. OF CHARACTERS BROWSED FOR EACH VALUE OF λ FOR FIRST FOUR EPOCHS. AFTERWARDS, THE NUMBER OF TARGET SYMBOLS FOR $\lambda = 0$ WILL KEEP ADDING IN EACH OF FURTHER EPOCHS.

λ	**3 or more ligatures**	**4 or more ligatures**	**5 or more ligatures**	**Length of text-line**	**Baseline: No curriculum**
3	126, 322	166, 285	165, 208	037, 628	303, 663
2	217, 516	236, 734	240, 421	163, 189	303, 663
1	279, 672	283, 050	286, 050	278, 067	303, 663
0	304, 040	304, 844	303, 490	304, 390	303, 663

where

$$N_\lambda = \sum_t (simplicity_t)^\lambda$$

is used to define complexity scores in term of probability.

It must be noted that we could have used a different criterion for calculating the complexity score like given in Equation 4:

$$CS_t = a_t + W * b_t + X * c_t \tag{4}$$

where W and X (with $X \gg W$) are weights for ligatures with more no. of characters. However, we found that the use of Equation 1 is more appropriate as it assigns more weight to ligatures with high number of characters correctly.

III. EXPERIMENTAL EVALUATION

To evaluate our claim that using ligatures-complexity as the measure for complexity of a text-line, we performed two sets of experiments. In the first setting, the number of characters in a text-lines were used as a measure of complexity (same as used by [4]). Text-lines were assigned a simplicity score based on the following equation:

$$simplicity_t = \frac{1}{max(m, |Z_t|)} \tag{5}$$

where $|Z_t|$ is the number of characters in length of transcription and $m > 0$ is a constant to avoid numerical problems. Here we chose m to be 2, as a word in a Urdu single text-line contains a minimum of two characters.

In the second set of experiments, we used the number of characters in a ligature as a measure of complexity of a text-line. The *simplicity* score was calculated using Equation 2. The procedure for training is the same as described in [4]. We changed the number of variables in Equation 1 to see the effect of including four- or five-characters ligatures in the calculations of complexity scores. The number of variables increase to four in Equation 1 in the first scenario and to five in the second one.

For both sets of experiments, we kept the initial value of $\lambda = 3$ and then we decreased it to 0 linearly in the first four epochs of training. Based on the value of λ, each training text-line's probability value is determined and the samples were drawn according to that distribution for 4 different values of λ. Table I gives the statistics of the number of target symbols browsed for the first four epochs for each experiment. In the following epochs, the number given for $\lambda = 0$ gets added to the total number of *so far* browsed target symbols.

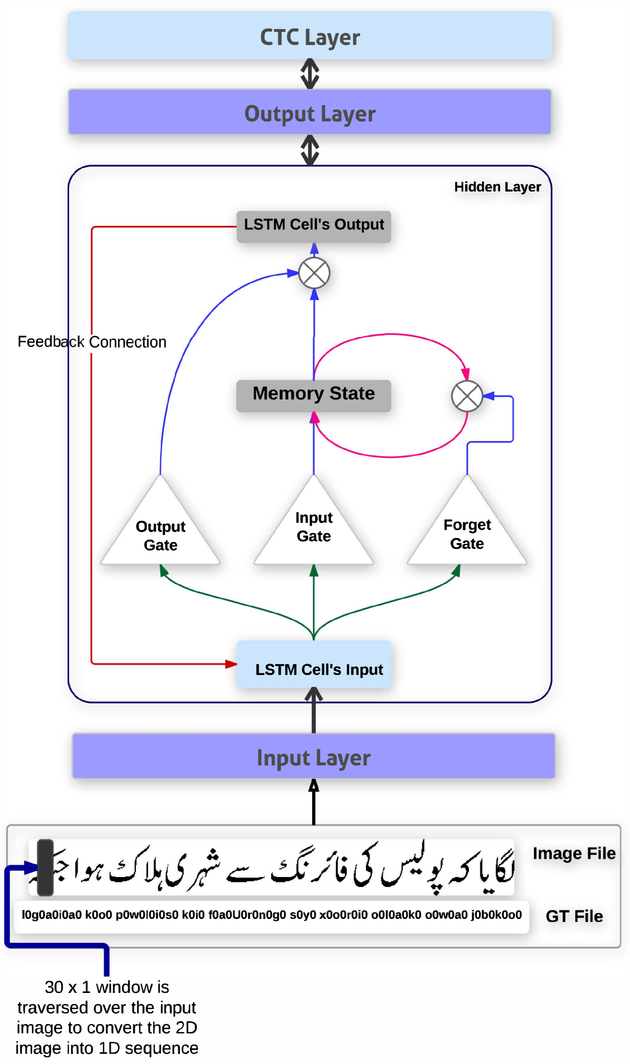

CTC Layer
Output Layer
Hidden Layer
LSTM Cell's Output
Feedback Connection
Memory State
Output Gate
Input Gate
Forget Gate
LSTM Cell's Input
Input Layer
لگایا کہ پولیس کی فائرنگ سے شہری ہلاک ہوا جبکہ
Image File
l0g0a0i0a0 k0o0 p0w0l0i0s0 k0i0 f0a0U0r0n0g0 s0y0 x0o0r0i0 o0l0a0k0 o0w0a0 j0b0k0o0
GT File
30 x 1 window is traversed over the input image to convert the 2D image into 1D sequence

Fig. 3. Simplified 1D-LSTM architecture. The Hidden layer consists of LSTM memory blocks. Each memory cell is connected to its surroundings with **input** and **output** gates. The input gate allows the input to be read, output gate allows outputs to be written. In addition **forget** gate allows retention of the information within the memory cell. The CTC layer aligns the output activations with the ground-truth sequence [6]. The input image is traversed by a $X \times 1$ (X being the height of the image) window to convert the 2D image into a 1D sequence.

A. System Architecture

We used 1D-LSTM architecture to train the LSTM network (same as described in [13]). In 1D-LSTM networks, a sliding window of 1-pixel width and of height equal to the image height traverses the input text-line image to convert it into a one dimensional sequence. The height of image is termed as "depth" of the 1D sequence and each slice of image thus obtained is called a "frame". It is necessary to make depth of each image equal, so that the individual frames in each image are equal. This process of making heights equal is called "normalization", and it is essential for such types of networks.

For this work, we used the same normalization method used in [13], that is heights were made equal by just re-scaling the image to a particular height (30 in our case) keeping the height-to-width ratio same. The LSTM network architecture

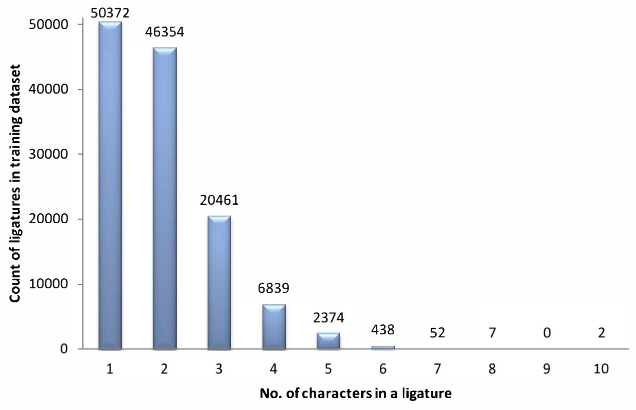


Fig. 4. Frequency distribution of ligatures in training set of UPTI dataset as a fucntion of constituent letters. 1 letter ligatures are infact isolated characters. 2-letter and 3-letter ligatures have far more freuency than other ligatures in this database. Please note that first two bars are clipped to show smaller count ligartures.

is shown in Fig. 3. It consists of one input layer, one hidden layer containing 100 LSTM cells and one output layer. CTC layer [6] is used in transcription task at output layer to make LSTM network trainable on full text-lines.

In the curriculum setting, a sample of training image were drawn for each value of λ, and the LSTM network was trained using stochastic gradient descent learning. As mentioned earlier, the value of λ was initially set to 3 and then it was decreased to 0 linearly during the first four epochs. The value of $\lambda = 0$ corresponds to the situation, where each text-line was drawn with equal probability. For 1D-LSTM training, a sliding window of 30×1 traversed over the input image and each frame is fed to the LSTM network for training. The network learns to classify each frame into one of the target classes (reject class included) based on the context of that frame. We used the bidirectional mode of LSTM architectures, where there are basically two hidden layers, one traversing the input from right-to-left and the second from left-to-right. Both of these layers are connected to a single output layer.

B. Database

We used the freely available Urdu Printed Text Images (UPTI) database [18] for the evaluation of our curriculum strategy. This database consists of $10,063$ synthetically generated text-line images. These text-lines contains many degradations like jitter, elastic elongation, sensitivity and threshold applied on them. The whole database was divided into training (46%), validation (34%) and test (20%) categories. Fig. 4 shows the distribution of training ligatures as a function of number of constituent letters. One-letter ligatures are basically isolated characters. They are shown here just to show the relative frequency of isolated characters in the training database. This distribution reveals an interesting fact about the UPTI database and that is that more than 95% ligatures consist of 4 or less characters and isolated characters occur about 40%. So, we can say that this database contains a very small proportion of very complex ligatures (containing 7 or more characters).

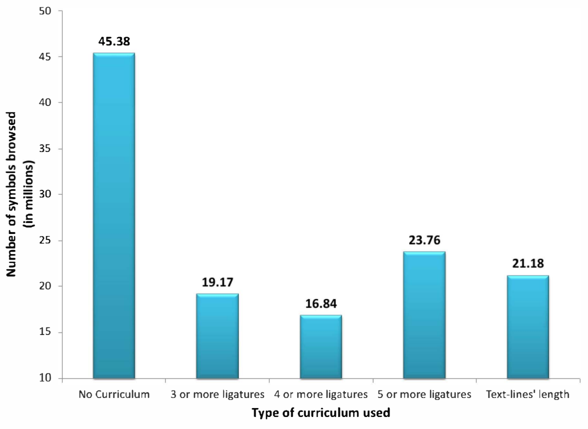


Fig. 5. No. of total target symbols browsed in each type of curriculum used. Note that without any curriculum learning, it took a lot of training symbols to be browsed before converging.

C. Performance Metric

Since we are interested in the convergence speed, we use the number of characters observed in each epoch (same as that used in [4][2]). Specifically, we observed the Character Error Rate (CER) against the number of characters observed in each epoch in terms of well-known edit distance criterion.

IV. Results and Discussion

In each epoch, the number of observed characters increases, so to compare different approaches, we plot the respective training errors against the number of training samples observed in each epoch. Total number of target symbols browsed during the training for each experiment type are plotted in Fig. 5. The progression of learning for increasing number of target symbols in each epoch is shown in Fig. 6. The Character Error Rate (CER) on the validation set is plotted against the number of target symbols browsed in each epoch.

It is evident from the plots that curriculum learning indeed speeds-up the convergence. However, curriculum learning failed to improve the validation errors in our experiments. The network was able to converge with smaller number of target labels when a curriculum is provided based on a complexity score. It is clear that choosing the right complexity measure for curriculum learning is very important. It is also apparent that for defining the curriculum, the ligature-based complexity measure is better than the length of text-line as complexity measure. The latter not only takes longer to converge than ligature-based criterion, but also the generalization error was high. For Urdu Nastaleeq printed OCR task, we found that faster convergence is achieved when we considered 3-characters ligatures separately in Equation 1 among different complexity measures for ligature-based curriculum, that is we lumped the count of 4- or more-characters ligatures together. The generalization error does not differ a lot in this case as well. As mentioned in Section III-B, more than 95% ligatures

[2]We didn't use the normalized Negative Log-Likelihood or normalized Edit distance measure because we are not comparing the effect training on ligatures (words) in our experiments.

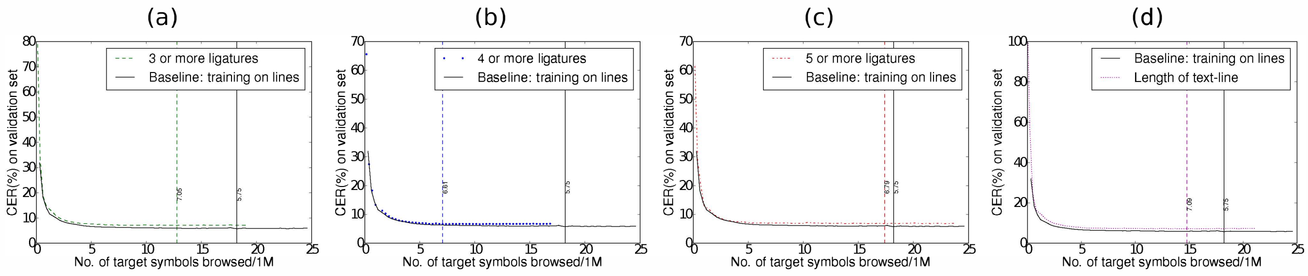


Fig. 6. Convergence curves for Curriculum learning for Urdu printed OCR. The dotted lines show the best validation error. Fig. (a) shows the case when 3 or more characters ligatures are lumped together, Fig. (b) shows the case when 4 or more characters ligatures are lumped together, Fig. (b) shows the case when 5 or more characters ligatures are lumped together Fig. (d) shows the case when length of text-lines are considered as the complexity measure. See Equation 1 for explanation of (a), (b) and (c) and Equation 5 for (d).

in the training data consist of 4-letters or less. This means that when we assign weights to these ligatures separately, the complexity scores assigned and consequently the probability distribution was realistic. This has resulted in better performance of the network when the complexity score is estimated with 4 or more lettered ligatures lumped together.

In the current curriculum strategy, the generalization error does not improve at all. A possible justification for this could be the fact that we started the training with simple ligatures in the curriculum learning. In Nastaleeq script, when complexity of ligatures increases, the characters tends to shift right and upwards (diagonal). This means that the vertical position of a character in a frame changes as the ligature's complexity increases, thus making it difficult for LSTM network to recognize the characters. A possible solution to this issue is to vertically normalize the frame for each character irrespective of its position in a ligature.

V. Conclusions

In this paper, we introduce a new curriculum learning strategy for Urdu Nastaleeq script. This method significantly reduces the number of target symbols browsed during the LSTM-based learning for printed Urdu Nastaleeq OCR task. The suggested technique may equally be applied to other cursive scripts like Arabic Naskh having many ligatures. We demonstrated that if the input text-lines are sorted according to the simplicity/complexity of ligatures therein, the LSTM networks can converge faster in term of number of target labels visited in each iteration. This work however can be extended into multiple directions. First, other cursive languages like Arabic and Persian can take the benefit of ligature-based complexity measure. Furthermore, this idea can be extended to Latin languages where word-based complexity measure can be defined instead of ligatures. Instead of training on individual words first as done by [4], if text-lines are sorted on the amount of words and on the basis of characters per words, then in our opinion, it will yield better results.

References

[1] V. Frinken, A. Fischer, M. Baumgartner, and H. Bunke, "Keyword Spotting for Self-training of BLSTM NN based Handwriting Recognition Systems," *Pattern Recognition*, vol. 47, no. 3, pp. 1073–1082, 2014.
[2] T. M. Breuel, A. Ul-Hasan, M. Al Azawi, F. Shafait, "High Performance OCR for Printed English and Fraktur using LSTM Networks," in *ICDAR*, Washington D.C. USA, aug 2013.
[3] E. Indermühle, V. Frinken, and H. Bunke, "Mode detection in online handwritten documents using blstm neural networks," in *ICFHR*, Bari, Italy, 2012.
[4] J. Louradour and C. Kermorvant, "Curriculum Learning for Handwritten Text Line Recognition." in *DAS*, France, 2014.
[5] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[6] A. Graves, S. Fernndez, F. J. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks." in *ICML*, 2006, pp. 369–376.
[7] F. Weninger, J. Bergmann, and B. Schuller, "Introducing CURRENNT - the Munich Open-Source CUDA RecurREnt Neural Network Toolkit," *Journal of Machine Learning*, vol. 15, 2014.
[8] I. Sutskever, "Training Recurrent Neural Networks," Ph.D. dissertation, Dept. of Computer Science, Univ. of Toronto, 2012.
[9] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, "On the Importance of Initialization and Momentum in Deep Learning," in *ICML*, USA, Jun. 2013, pp. 1139–1147.
[10] B. Skinner, "The behavior of Organisms: An experimental analysis," *Appleton-Century-Crofts.*, 1938.
[11] J. L. Elman, "Learning and Development in Neural Networks: The Importance of Starting Small," *Cognition*, vol. 48, no. 1, pp. 71–99, 1993.
[12] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum Learning," in *ICML*, Montreal, Quebec, Canada, 2009, pp. 41–48.
[13] A. Ul-Hasan, S. B. Ahmed, S. F. Rashid, F. Shafait, and T. M. Breuel, "Offline Printed Urdu Nastaleeq Script Recognition with Bidirectional LSTM Networks." in *ICDAR*, Washington D.C. USA, aug 2013.
[14] S. Naz, K. Hayat, M. I. Razzak, M. W. Anwar, S. A. Madani, and S. U. Khan, "The optical character recognition of Urdu-like cursive scripts." *Pattern Recognition*, vol. 47, no. 3, pp. 1229–1248, 2014.
[15] Q. Akram, S. Hussain, F. Adeeba, S. Rehman, and M. Saeed, "Framework of Urdu Nastalique Optical Character Recognition System," in *Conference on Language and Technology*, Karachi, Pakistan, 2014.
[16] Q. Akram, S. Hussain, A. Niazi, U. Anjum, and F. Irfan, "Adapting Tesseract for Complex Scripts: An Example for Urdu Nastalique," in *DAS*, France, 2014.
[17] M. Ijaz and S. Hussain, "Corpus Based Urdu Lexicon Development," in *Conference on Language Technology*, Peshawar, Pakistan, 2007.
[18] N. Sabbour and F. Shafait, "A Segmentation-free Approach to Arabic and Urdu OCR," in *SPIE Document Recognition and Retrieval XX, DRR13*, San Francisco, CA, USA, Feb. 2013.