

Unconstrained Off-Line Urdu Handwriting Recognition



By

Nauman Riaz

MS-CS 2020 329811 SEECS

Supervisor

Dr. Faisal Shafait

Department of Computing

A thesis submitted in partial fulfillment of the requirements for the degree of Masters
of Science in Computer Science (MS CS)

In

School of Electrical Engineering & Computer Science (SEECS) ,

National University of Sciences and Technology (NUST),

Islamabad, Pakistan.

(August 2022)

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis entitled "Offline Urdu Handwriting Recognition using Conv-Transformer Architecture" written by Nauman Riaz, (Registration No 00000329811), of SEECs has been vetted by the undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/M Phil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the said thesis.

Signature: _____  _____

Name of Advisor: _____ Prof. Dr. Faisal Shafait _____

Date: _____ **24-Aug-2022** _____

HoD/Associate Dean: _____

Date: _____

Signature (Dean/Principal): _____

Date: _____

Approval

It is certified that the contents and form of the thesis entitled "Offline Urdu Handwriting Recognition using Conv-Transformer Architecture" submitted by Nauman Riaz have been found satisfactory for the requirement of the degree

Advisor : Prof. Dr. Faisal Shafait

Signature: 

Date: 24-Aug-2022

Committee Member 1: Dr. Muhammad Moazam
Fraz

Signature: 


Date: 24-Aug-2022

Committee Member 2: Mr. Adnan Ul-Hasan

Signature: 

Date: 25-Aug-2022

Committee Member 3: Dr. Muhammad Imran Malik

Signature: 

Date: 25-Aug-2022

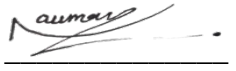
Dedication

This thesis is dedicated to my parents who never stopped believing in me for once, who supported me through thick and thin. Who taught me that it is never too late to pursue your dreams and passion. Without their constant support, I would not have been able to achieve my aim.

Certificate of Originality

I hereby declare that this submission titled "Offline Urdu Handwriting Recognition using Conv-Transformer Architecture" is my own work. To the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics, which has been acknowledged. I also verified the originality of contents through plagiarism software.

Student Name: Nauman Riaz

Student Signature: 

Acknowledgments

Starting by praising Almighty ALLAH(S.W.T), who gave me the strength, courage and willpower to complete my research period. I would like to express my gratitude to my supervisor, Dr. Faisal Shafait for his clear and practical guidance, continuous uplifting, despite his busy schedules. It has been an absolute honor for me to work with the team at TUKL, all faculty members, interns included.

Last but not the least, my family, for continuously supporting me in my times of despair. My thesis milestone could not have been completed without the consistent prayers of my mother and the countless brainstorming sessions with my father.

Nauman Riaz

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Problem Statement	3
1.3	Proposed Solution	3
1.4	Contribution of this Thesis	3
1.5	Overview of Thesis	4
2	Literature Review	5
2.1	Support Vector Machine Model	5
2.1.1	Methodology and Results	6
2.1.2	Drawbacks	7
2.2	Bi- Long-Short-Term Memory Network	7
2.2.1	Methodology and Results	8
2.2.2	Drawbacks	10
2.3	CNN- Recurrent Neural Network Model	10
2.3.1	Methodology and Results	10
2.3.2	Drawback	11
2.4	Convolutional Recursive Model	12
2.4.1	Methodology and Results	12
2.4.2	Drawbacks	13
2.5	Urdu Handwritten Character Recognition using CNNs	14

CONTENTS

2.5.1	Methodology and Results	14
2.5.2	Drawbacks	16
2.6	Sequence-to-Sequence Models for Text Line Recognition with CTC-Prefixes	16
2.6.1	Model Architecture and Methodology	16
2.6.2	Datasets and Results	16
2.6.3	Drawbacks	17
2.7	Attention-Based Techniques	17
2.7.1	Methodology and Results	17
2.7.2	Drawbacks	19
2.8	Transformer Based OCR Model	20
2.8.1	Methodology and Results	20
2.8.2	Drawbacks	21
2.9	Transformer Based Light Encoder Decoder Architecture	22
2.9.1	Methodology and Results	22
2.10	Drawbacks of Architectures Discussed	23
3	Methodology	25
3.1	Unconstrained Off-Line Urdu Handwriting recognition as a Seq2Seq modeling task	25
3.2	Encoder-Decoder Transformer Architecture	26
3.3	Conv-Transformer Architecture	27
3.3.1	Convolutional Neural Networks	28
3.3.2	Transformer	29
3.4	The Attention Mechanism	31
3.5	Beam Search	34
3.5.1	Computation of the Sequence Probability	34
3.5.2	Beam Search Working Mechanism	35

4	Implementation and Results	37
4.1	Dataset	37
4.2	Data-Preprocessing and Augmentation	38
4.3	Implementation Details and Hyperparameters	39
4.4	Experiments Performed	40
4.5	Results	41
5	Discussion	43
5.1	Comparison with Conv-Recursive Architecture	43
5.2	Comparison with Google’s Vision API	44
5.3	Smoke Testing on Random Images	45
5.4	Ablation Studies	46
5.4.1	Ablation Study - Only encoder CTC	46
5.4.2	Ablation Study - Encoder decoder	46
5.4.3	Analysis of Failure Cases	47
6	Conclusion and Future Work	49
6.1	Conclusion	49
6.2	Future Directions	49

List of Figures

1.1	Urdu Handwriting Complexity	2
2.1	Grayscale Images and Gradient Strength	6
2.2	Bi-LSTM Recognition System Architecture	9
2.3	CNN-RNN Network Architecture	11
2.4	CNN Recursive Model Architecture	13
2.5	Character Level Urdu Handwritten text Recognition Model Architecture	15
2.6	Grouping of Urdu Charcaters	15
2.7	Attention-Based Encoder Decoder Architecture	19
2.8	Transformer Based OCR Model	21
2.9	Transformer Based Encoder Decoder Architecture	23
3.1	Encoder-Decoder Transformer Architecture during training	27
3.2	Custom CNN Layers	29
3.3	Encoder-Decoder Transformer Architecture during testing	33
4.1	Data augmentation approaches	39
5.1	Results from the Smoke study	45
5.2	Instances the input with its ground-truth and predicted text	48

List of Tables

3.1	Custom CNN Network Configuration	30
4.1	UHWR dataset statistics	37
4.2	UPTI-2 and Ticker dataset statistics	38
4.3	Comparison of Conv-Recursive with Conv-Transformer	41

Abstract

The task of Unconstrained Off-Line handwriting recognition is challenging in general and particularly difficult for Arabic-like scripts and is an active research area. Recent use of Transformer models for the task of English Handwriting Recognition have shown promising results. The proposed solution includes the fusion of Convolutional Neural Network before a vanilla Transformer architecture and the use of printed Urdu text along with handwriting text during training. Convolutional Blocks decrease the spatial resolutions in order to make up for the Transformer's attention layers' n^2 complexity. Moreover, the use of printed text along with handwritten text aids in learning diverse ligatures and a better language model for the transformer during training. On the publicly accessible NUST-UHWR dataset, the proposed model achieves the state-of-the-art accuracy with a CER of 5.31 percent.

CHAPTER 1

Introduction

Humans are unique among species because of our ability to communicate in writing. This method of communication has continued to be effective till date. Writing things down by hand is still the most practical way to take notes, fill out forms, and write addresses, despite all the advances in speech to text and word processor technology.

The method of converting text seen in images into corresponding editable text is known as automatic text recognition. The process of digitizing documents and putting them in editable format has many practical uses. We can pass onto future generations our cultural legacy and knowledge of our predecessors.

In addition to helping to preserve history and cultural heritage, digitalization is crucial for modernizing many daily tasks. Mail delivery times can be cut down greatly with postal automation. The creation of digital databases and systems for decision support benefits from the information retrieval from documents like medical records and forms. Consequently, text recognition has been a hot topic for research during the past few decades.

1.1 Motivation

Pakistan's national language is Urdu which is one of the two official languages. With approximately 61.9 million native speakers¹), it is the 21st most widely used first language worldwide. It is derived from Arabic and is typically written in Nastaliq script. The majority of Pakistanis are bilingual in speaking and understanding it [48].

¹<https://en.wikipedia.org/wiki/Urdu>

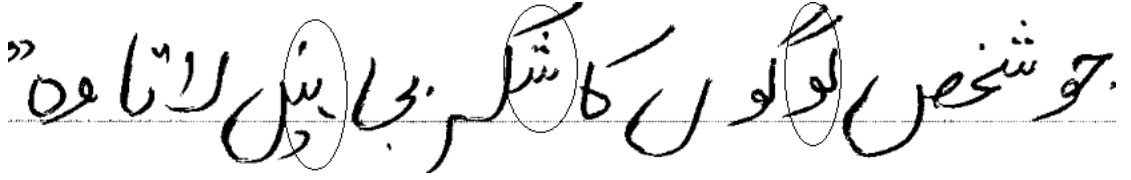


Figure 1.1: This figure shows the complexity of a Urdu handwriting. One can see the characters overlapping each other vertically. Moreover, some characters can very easily be confused with other similar looking characters.

The identification of printed text is seen as a task that has been solved, and there are workable methods to accurately digitize/convert scanned documents into editable text in a variety of languages (Google Vision API supports over 100 languages ²)

Even though handwriting is widespread and intuitive to people, it has proven difficult for computers to detect it. When it comes to handwriting, humans are incredibly inventive, which results in a wide variety of writing styles, character formations, etc. Everybody has a unique writing style, thus teaching a model to understand an unknown handwriting style is a difficult challenge. Therefore, while addressing handwriting recognition, it is important to take into account several features of writing, including writing styles, the type of paper used, the thickness of strokes, human mistake, and a number of other issues.

The characters in Urdu scripts (Nastaliq or Naskh) are linked together to produce ligatures³, just like in Arabic. As a result, context is crucial while recognizing Urdu text. In addition, some ligatures or characters vertically overlap one another due to joining. Some characters are so similar to one another that it's very easy to confuse them with other characters (please refer to Figure 1.1)). There are around 24, 000 different ligatures in Urdu [51], and their joining criteria vary. Due to these difficulties, recognizing Urdu text has become a very challenging task.

There are two main methods for offline handwriting text recognition. The first method is segmentation-based. This method isolates each letter, ligature, or word and identifies it separately [21]. But because Urdu text is so context-sensitive, this method is not particularly effective [36]. The second strategy is non-segmentation based. In this method, text recognition is modeled as a sequence-to-sequence task. Inspiration from

²<https://cloud.google.com/vision/docs/languages>

³<https://www.w3.org/TR/alreq/>

[50] was taken for this methodology in which Neural machine translation is also modeled as Seq2Seq problem.

Text-recognition tools or systems are already available for symbolic and alphabetic languages, but Arabic-based languages, including Urdu, lack such solutions [40]. There have been attempts to manually transcribe material for digitization, but doing so would be challenging, expensive, and time-consuming [51].

1.2 Problem Statement

For document digitization, handwritten text recognition has been a long-standing research challenge in general and particularly for scripts like Arabic. The majority of text recognition techniques currently in use are focused on CNN for image comprehension and RNN for character level text generation. Additionally, as a post-processing step, another language model is typically required to increase the overall accuracy which fails to capture long term complex dependencies of the languages like Urdu.

1.3 Proposed Solution

A Conv-Transformer architecture, an end-to-end text recognition approach based on the Convolutional encoder and the vanilla Transformer model, which uses the Convolution + transformer encoder for image understanding and the Transformer decoder for character-level text generation is proposed. Use of Transformer decoder eliminates the need for a separate language model and Convolutional blocks cater for the n^2 complexity of the attention layers of Transformer by reducing the spatial resolution of the input text image. Moreover, printed Urdu text along with the handwritten text is used during training to aid the Transformer in learning a diverse language model.

1.4 Contribution of this Thesis

In contrast to handwriting recognition, which is open to novel approaches, most research on Urdu text recognition concentrates on printed text [38, 21, 17]. The vast majority of literature in Urdu handwriting recognition is on stroke-based online handwriting recognition [41], which uses touch input on touch-sensitive devices such mobile devices

as a guide to recognize text written by hand. The scope of this thesis on the other hand is offline Urdu handwriting recognition, which makes use of images of handwritten Urdu text.

This thesis majorly contributes to the use of a CNN + Transformer architecture that has been used for the implementation of the task of Urdu Handwriting Recognition. This architecture involves the following major components to perform the task of Urdu Handwriting Recognition:

- A Convolutional Neural Network - used for the extraction of the visual information from the images.
- A Full-Transformer - The visual information extracted from the Convolutional Neural Network is then fed to a full-transformer [50]. This transformer consists of three layers of encoders and decoders that are stacked on the top of each other.
- A Transformer Decoder - The transformer decoder then accepts the encoded sequence and produces a digitized handwritten text as its output. Due to the use of a transformer in our model, the model works as an auto-regressive model.
- Testing - A technique inspired from the beam search was used for the testing purposes so that the most probable outcome can be selected amongst the top most $-k$ probable output sequences.

1.5 Overview of Thesis

The following chapters serve as the primary divisions of the thesis. The relevant work in the area of offline Urdu handwriting recognition is summarized in **Chapter 2** of this thesis. The proposed approach addressing the task at hand is covered in detail in **Chapter 3**. The experimental setup is described in **Chapter 4**, along with the data augmentations that were employed and implementation specifics. The results and their analysis are presented in **Chapter 5**. **Chapter 6** brings the study to a completion and offers suggestions for further research.

CHAPTER 2

Literature Review

Most of the Urdu Handwriting Recognition techniques that have been implemented so far can broadly be categorized into the following two of the major categories:

- Holistic Methodology - For the Roman scripts, this methodology typically refers to word-level handwriting identification and in terms of the Arabic and Urdu languages, they speak of ligatures or fragmentary words.
- Analytical Methodology - This methodology refers to the character level recognition of the text. It includes both, the printed and the handwritten writings. Though much improvements have been made in the area of printed Urdu text recognition, but handwritten text recognition still lacks research and can open new areas of research in this regard.

The analysis that follows displays the work that has been done in this area of the study and a detailed comparison in the up coming section explains how the proposed methodology of this thesis can be implemented in order to improve the task of Urdu Handwriting text recognition.

2.1 Support Vector Machine Model

Support Vector Machine model for Urdu text Recognition was used by Sagheer et al. in [48]. This paper is based on the holistic handwritten urdu word recognition methodology.

2.1.1 Methodology and Results

The following steps were implemented in order to perform the task at hand:

- Pre-processing: As the first pre-processing step, the input images were first converted into grayscale images as well as the binarized images. The salt and pepper noise was eliminated from the images using the median filter.



(a) 128 x 128 Grayscale Image (b) Image's Gradient Strength (c) Gradient Direction

Figure 2.1: Figure 2.1a represents the 128 x 128 grayscale Image that was obtained as a result of the pre-processing step. Nine vertical blocks and nine horizontal blocks made up the 81 blocks that made up the gradient image. Figure 2.1b represents the gradient strength of the image. The gradient strength was tallied in 32 directions for each block. Figure 2.1c represents the Gradient Direction. By using a weight vector of $[1 \ 4 \ 6 \ 4 \ 1]$ to downsample, the number of directions is reduced from 32 to 16, yielding a feature vector of 400 size [48].

- Extraction of gradient features: The structural and the gradient features were extracted in order to get the feature maps. From each normalised (64×64 size) binary image, only the upper profile characteristics and well-known horizontal and vertical projection profiles were extracted. The structural characteristics of an image are its external components. The structural features include the upper and lower profiles [9], features [7], the projection profile, and others. The upper and lower profile elements capture the shape contour of a printed or handwritten text [28]. The outline contour of the top portion of the word is captured using the upper profile features. A two-dimensional array was created from each photograph. Each column was looked from right to left. The distance between the top of the image and the first ink pixel was computed for each column. The distance value

would be returned in the range of 0 to 64 after this step. The upper profile effectively conveys the contour of a word’s outline. An example of the gradient images of an Urdu word is shown in the Figure 2.1: Figure 2.1a represents the Grayscale image of size 128 x 128. The Figure 2.1b below represents the gradient strength of the image. Likewise, the Figure 2.1c represents the gradient direction.

- **Classification:** The Support Vector Machine model was used along with the Radial Basis Function(RBF) kernel for the implementation of the classification task. Cross Validation was used in order to choose the best hyper-parameters.
- **Dataset:** The dataset ‘CENPARMI Urdu word dataset’ contained the following distribution for training and the test sets: The Training set contained 14,407 samples. The Test set included 3770 samples. Other than this distribution, this dataset contains 57 words. These words include words for measurements, currencies and weights.
- **Performance of the model:** According to the paper, the model achieved 97 percent recognition performance.

2.1.2 Drawbacks

The support vector machine is a simple machine learning model which ignores the dependencies within the Urdu language. Such models fail for complex datasets like UHWR. It is a must to capture diversities of the language itself in order to perform good on handwritten text images in which some of the written characters are ambiguous. When some words in the written text are hard to recognize we can take help of the the trained language model. Humans also recognize text effectively due to their ability to better understand the language.

2.2 Bi- Long-Short-Term Memory Network

For the effective recognition of the recursive text, convolutional recursive architectures can be implemented. The techniques that have been used in [51], [21], [39], [40], and [33] show how these techniques can be a good idea for the effective recursive text recognition task.

For the character segmentation task, Hassan et al. in [39] proposed an approach that is based on the analytical methods for the character level text recognition tasks. Cursive handwritten urdu text recognition was used as a case study for the paper.

2.2.1 Methodology and Results

This approach consists of the following major tasks:

- **Pre-processing:** The input image is first binarized. The result is a series of stroke sequences that are then mapped in accordance with the transcription.
- **Feature Extraction:** In this technique, the character segmentation is performed implicitly using the Convolutional Neural Network that act as the feature extractors for the underlying model.
- **Classification:** The classification task is then performed using the Bi-Long Short Term Memory(LSTM) Network. The Feature maps that have been extracted using the Convolutional layers are transformed into the feature sequences. The LSTM layer is then given these feature sequences.
- **Network Architecture:** A total of 7 convolutional layers are used in the network architecture that are then followed by pooling layers, batch normalization layers and drop out layers in between them. The architecture also consists of 2 Bi-LSTM layers.
- **Classification Results:** The study claims that an average character identification rate of more than 83 percent was attained in studies conducted on a sample of 6000 different text lines.
- **Dataset:** The dataset that was used for this study is UNHD dataset [35].

Furthermore, the authors of this paper also proposed that their work can be extended so that the main ligatures can be recognised separately. This might result in fewer character classes overall.

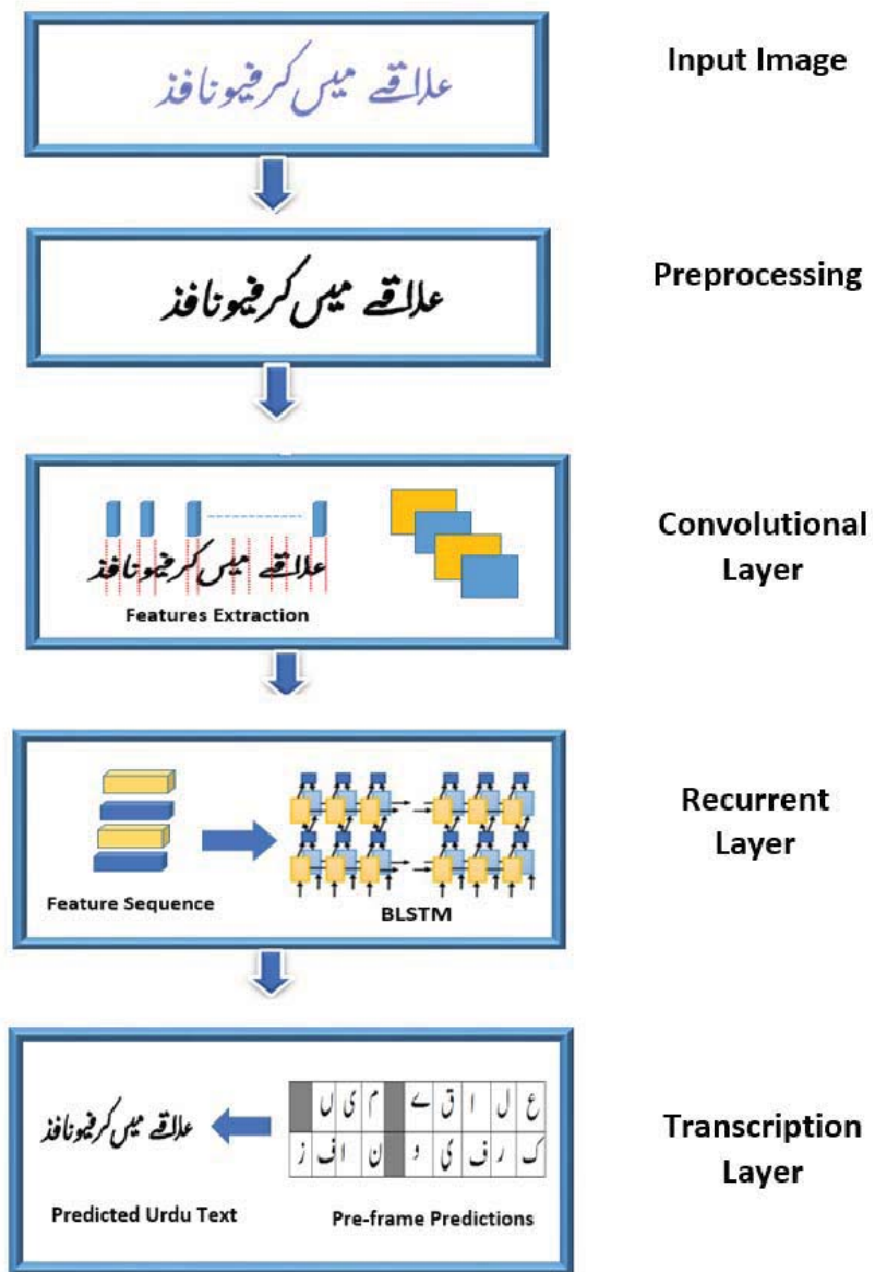


Figure 2.2: The figure represents the steps that are applied to the input image in order to perform the recognition task. The input image is first preprocessed and then CNN is applied that acts as a feature extractor. The Bi-LSTM is then applied to perform the classification task [39].

The Figure 2.2 shows the architecture of the model that was presented in this study.

2.2.2 Drawbacks

This architecture uses the Connectionist Temporal Classification loss (CTC loss) for the task of handwriting recognition. The use of CTC loss again ignores the need for a separate language model which would fail for images in which words are ambiguous to understand.

2.3 CNN- Recurrent Neural Network Model

A handwriting recognition model was presented in a similar study by Zia et al. in [51]. This model is based on CNN-RNN architecture with the use of separate n-gram language model.

2.3.1 Methodology and Results

This architecture works by first increasing the size of input image from 100 pixels height to 128 pixels height. The input is passed through a convolution block and relevant features from the image are extracted. For the pre-processing steps, before feeding the features to the LSTM layer, the features are concatenated. This technique is used instead of the max pooling layer so that the extra dimension can be reduced. CTC loss is used at the end in order to recognize the text. In order to distort the images randomly for image augmentation, the random distortion layer is added before the input layer.

Separate n-gram language model is used to capture the diversity of the language. To prevent zero frequency n-gram difficulties [11], which might occur as a result of unseen words, smoothing and interpolated n-gram models were used, which blend the strengths of higher- and lower-order n-grams. The lower-order and higher-order models were merged using the contextual relationship between the n-gram orders using modified Kneser-Ney smoothing [3], a cutting-edge method for language modelling, and an enhanced form of absolute discounting [1].

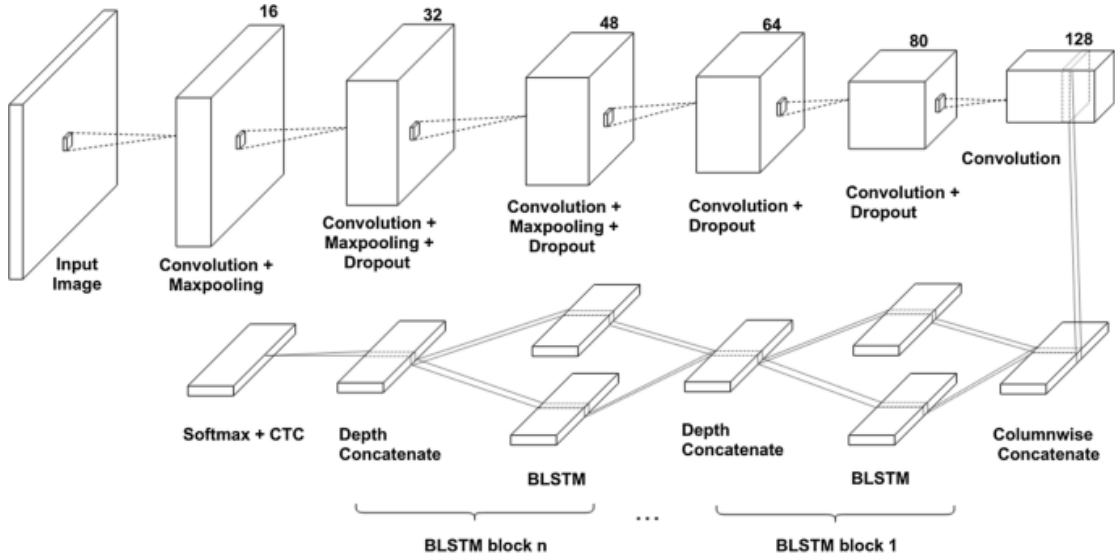


Figure 2.3: The architecture of the network. Convolutional blocks are the first part of the architecture, followed by a bidirectional LSTM block and a transcription layer. The features are first concatenated before feeding them to the LSTM layer [51].

The architecture for the proposed model has been presented in the Figure 2.3.

The model proposed in this study has a minimum Character Error Rate(CER) of 5.28 percent that was achieved on a newly created dataset called ‘NUST-UHWR’. The architecture proposed in this study makes the use of CTC loss at the end. The use of CTC loss makes this problem as a sequence labelling problem. Since it is treated as a sequence labelling problem, so in order to capture the probability of the next character when the previous n-characters are given, requires the use of n-gram modelling.

2.3.2 Drawback

This study uses the n-gram modelling technique for the implementation of the Handwriting Recognition Task, but it is quite clear and evident from the models like LSTM, gpt-3 [37], etc. that the task of the language modelling can be performed way more efficiently if the deep learning models are implemented rather than the statistical models like n-grams. This information inspires us to implement the handwriting recognition task as sequence to sequence task like the Machine Translation using Neural Networks.

2.4 Convolutional Recursive Model

In [21], Naz et al. used the state-of-the-art technique for printed text recognition that is the Convolutional Recursive Technique. A new convolutional-recursive deep learning model that combines CNN and MDLSTM is presented in this study. The proposed model is used to address the character recognition issue on Urdu text written in the Nastaliq script and is primarily based on the one described by Raina et al. [8]. For autonomously extracting lower level characteristics from a large MNIST dataset, the suggested approach uses CNN. The MDLSTM model is utilised as the classifier after the learnt kernels are convolved with text line images to extract features.

2.4.1 Methodology and Results

A 5-layered CNN model was used in order to extract the generic and the abstract features. The dataset from which the features were extracted was the MNIST dataset. Then, for the contextual features and classification, the extracted features are fed into a Multi-Dimensional LSTM.

An LSTM with many dimensions is used to train the system. An adaptation of recurrent neural networks (RNNs) is the LSTM [4]. Artificial neural networks having cyclic routes or loops are known as recurrent neural networks. The network can handle any sequence of inputs through internal memory due to the loops, which also allow for dynamic temporal activity. Long-term dependencies, however, cannot be learned by these networks. The issue was solved by the development of LSTM-RNN [2], which can retain and correlate information over extended delays. A memory block containing memory cells and three gates is the fundamental building component of the LSTM architecture (input, forget and output). By utilising n self connections with n forget gates, the conventional one-dimensional LSTM network can also be expanded to several dimensions.

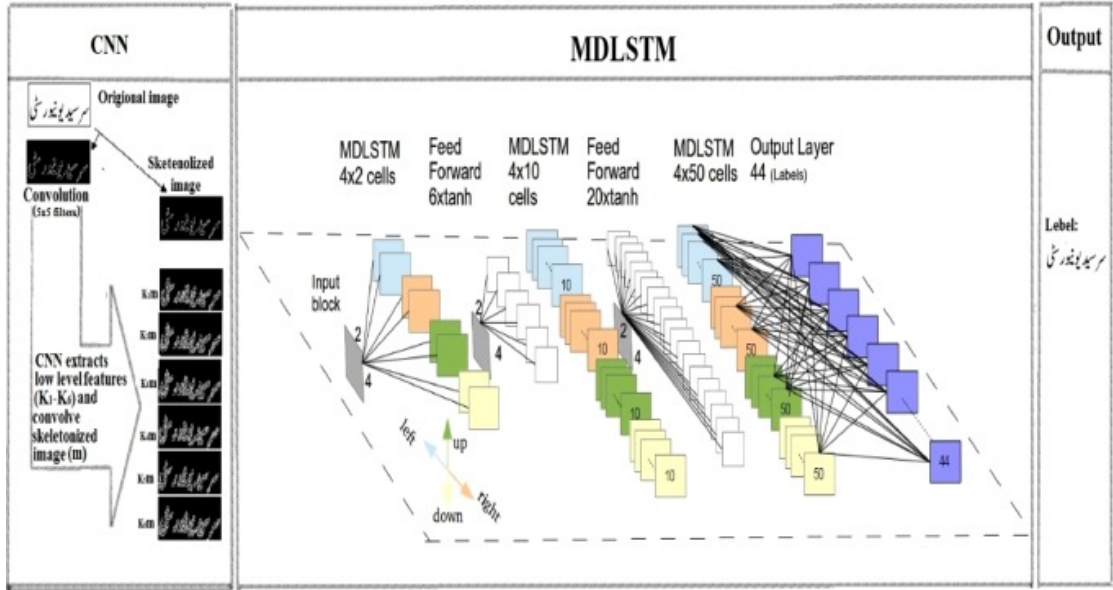


Figure 2.4: Using a single CNN layer, a convolutional-recursive deep learning network extracts low level characteristics from an Urdu textline. Six filters, K1 through K6, are picked from CNN’s top layer and are applied to the contoured image. A MDLSTM with random weights is given the convolutionalized images and the contour representation of the textline as input. The features are then iteratively mapped onto a lower level space by each neuron. The final feature vector for a Connectionist Temporal Classification (CTC) output layer is created by concatenating every resultant vector [21].

The overview of the proposed architecture has been shown in the Figure 2.4.

The model proposed in this study achieved 98.12 percent accuracy when implemented on the UPTI dataset. The authors of this paper also proposed that their work can be extended for other languages like Persian and Arabic as well.

2.4.2 Drawbacks

The suggested architecture again uses CTC-loss for decoding which ignores language modeling task. No separate language model was used as well. The thesis study shows that the architectures which ignore the language modeling aspect have failed to show good results on the the task of Urdu handwriting recognition.

2.5 Urdu Handwritten Character Recognition using CNNs

In order to recognize the Urdu handwritten characters, Husnain et al. in [40] used the Convolutional Neural Networks.

2.5.1 Methodology and Results

The proposed technique consisted of the following major tasks for handwritten text recognition:

- **Preprocessing:** The images from the proposed dataset underwent certain preprocessing operations in this phase to get them ready for more processing. First, noise was removed from the images using the techniques described in the [13]. After that, the images in the dataset were transformed to grayscale and shrunk to 28x28 pixels while preserving the aspect ratio.
- **Feature Extraction:** In order to extract the structural and the geometrical information of the characters, each Urdu handwritten character was analysed.
- **Classification:** In order to produce the high quality results, the characteristics extracted were incorporated with the data based on the image's pixels. Once the features were extracted, they were then passed to Convolutional Networks consisting of 4 layers. At the end, a fully connected layer was incorporated for the purpose of classification.

The proposed architecture has a dataset of 31K images with 10 labels (0–9) and 32x32 pixel images of handwritten numbers in Urdu that will be put into the CNN model. The first layer in the model was a 2D convolution layer with a 55 kernel size. Each and every input image pixel will be interlaced using the aid of this layer. The result of this layer had a 28x28-pixel feature map encoded in it. The structural features were then integrated to create a feature vector. The activation function ReLU (rectified linear unit) allowed each output in the convolution to be activated [12]. When compared to the "sigmoid," the ReLU function can handle the gradient vanishing problem better [15].

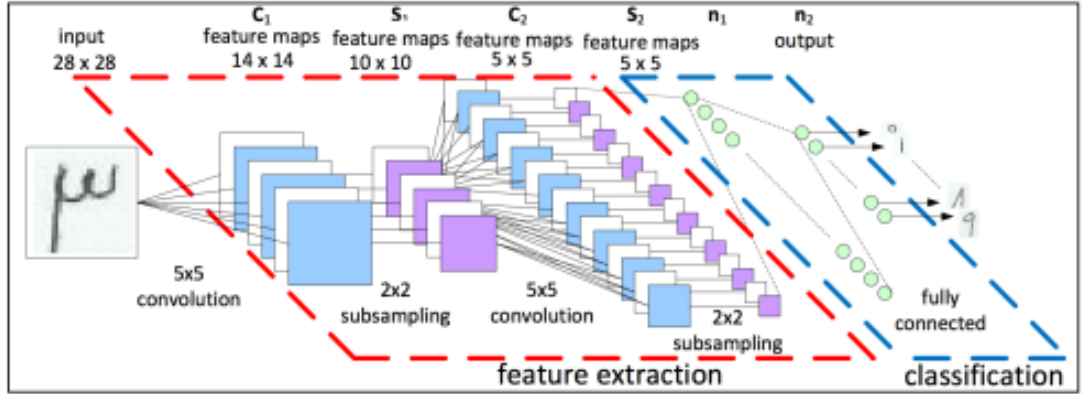


Figure 2.5: The figure represents the steps for Character Level Urdu Handwritten text recognition. The feature extraction technique is applied in order to extract the structural and the geometrical information of the characters. After the extraction of the features, they are then passed to CNN. Finally, a fully connected layer was applied for classification [40].

The architecture of the proposed model has been shown in the Figure 2.5.

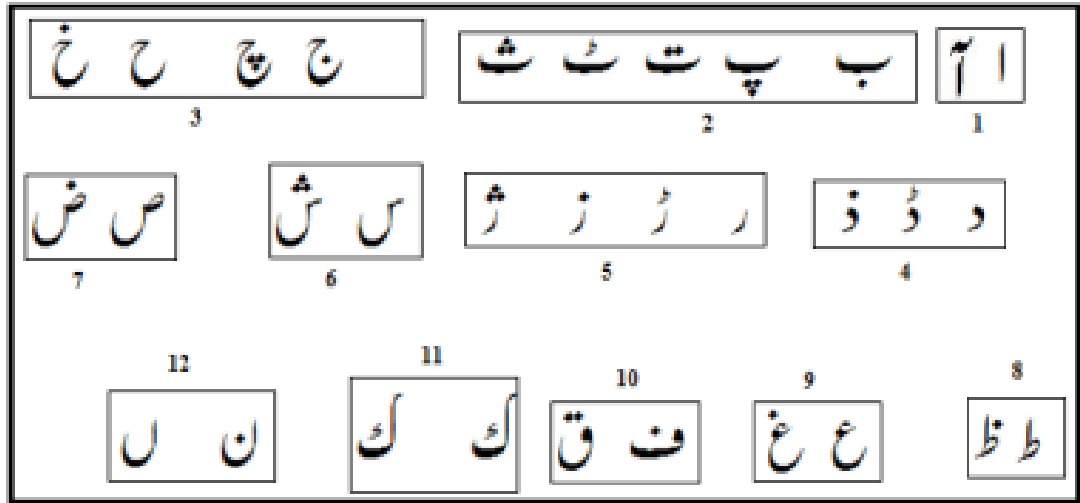


Figure 2.6: The Urdu characters are grouped according to the similarity in their shapes. Each one of the groups is numbered from the left to the right [40].

- Grouping of Characters: In the proposed technique the Urdu characters were grouped based on the shape similarity. The Figure 2.6 shows the group of charac-

ters:

- **Dataset:** The dataset for this study contained a total of 800 images of 10 numerals and 80 Urdu characters.
- **Model Accuracy:** An accuracy of 96.05 percent was achieved for the character-level Urdu Handwritten text recognition. reported according to the paper.

2.5.2 Drawbacks

The major drawback of this approach is that it only recognizes Urdu characters. Urdu character recognition is relatively much easier task and cannot be used for recognizing and digitizing complete Urdu sentences in handwritten images.

2.6 Sequence-to-Sequence Models for Text Line Recognition with CTC-Prefixes

The sequence to sequence models often do not work well and raise errors for the repeated words or the skipped words. This issue was then addressed by the authors in in [34] using the Connectionist Temporal Classification(CTC) prefixes.

2.6.1 Model Architecture and Methodology

In this methodology, the paths that are found to be invalid are first penalized. This process is implemented during the beam search. The proposed architecture consists of a CNN block that is followed by a Long-Short Term Memory based encoder. A Transformer based decoder is then used to decode the feature space. The next best characters for the purpose of decoding are then obtained by weighing and summing up the character costs of the CTC Prefix score and LM.

2.6.2 Datasets and Results

The model proposed by this study was evaluated on the datasets of IAMe, StAZH and Rimes. Unconstrained handwritten text can be found in the IAM Handwriting Database [5]. The state archive of Bozen's collection serves as the foundation for the ICFHR2016

READ data set (Bozen) [19]. It originated from the Horizon 2020 READ project of the European Union and is freely accessible. The data set comprises of a subset of Ratsprotokolle collection papers that are copies of the minutes of council meetings that took place between 1470 and 1805. A HTR competition at the ICFHR2016 used this data set as its foundation [18]. The state archive of the canton of Zurich is the source of the StAZH data set. On the IAM dataset, the model achieved a CER of 2.95%.

2.6.3 Drawbacks

The use of CTC-Prefix score greatly increase the decoding time of the proposed model. Moreover, pretrained Transformer decoder on English language modeling task is used. It gives poor results when fine-tuned on Urdu handwriting recognition task for obvious reasons. Pre-training the transformer decoder from scratch on Urdu language modeling task requires a lot of resources and is computationally quite expensive which becomes out of the scope of our thesis.

2.7 Attention-Based Techniques

In order to achieve a higher accuracy and improve the results of the previously mentioned techniques, [44] and [42] made use of an attention-based mechanism. By implementing the technique of attention based mechanism, for the prediction of each word, global reference can be achieved. An inspiration was gained by the model that was proposed in [30] for the task of neural machine translation and Michael et al.in [44] then redesigned the above mentioned attention based sequence to sequence model for the underlying task.

2.7.1 Methodology and Results

A contrast normalisation without any picture binarization, a skew correction, and a slant normalizer are all included in the preprocessing of text line images. Additionally, each image maintains its aspect ratio while scaling to a set height of 64 pixels. This prevents any length limits on the feature sequence and guarantees that all vectors comply to the same dimensionality. The existing preprocessed images are augmented by making slight adjustments to them in order to fictitiously increase the amount of training data. Dilation, erosion, and grid-like distortions are used to imitate naturally occurring

variations in handwritten text line images [23]. These techniques are randomly applied with an independent probability of 50% to the initial line image.

The proposed model consists of a combination of a CNN that acts as a feature extractor and RNN. The RNN consists of three Bi-LSTM layers. It is the architecture's encoder part, This is used to encode the visual information in the input images. Not only the visual information, but also the temporal context of the input images is also encoded.

In order to decode the actual character sequence, a separate LSTM is used which consists of 256 hidden units. The extracted features and the decoder's hidden state are then linked by applying the attention between them. Since the relative positions of the tokens in any sequence can be provided by the positional embeddings so the positional embeddings are injected in the input sequence.

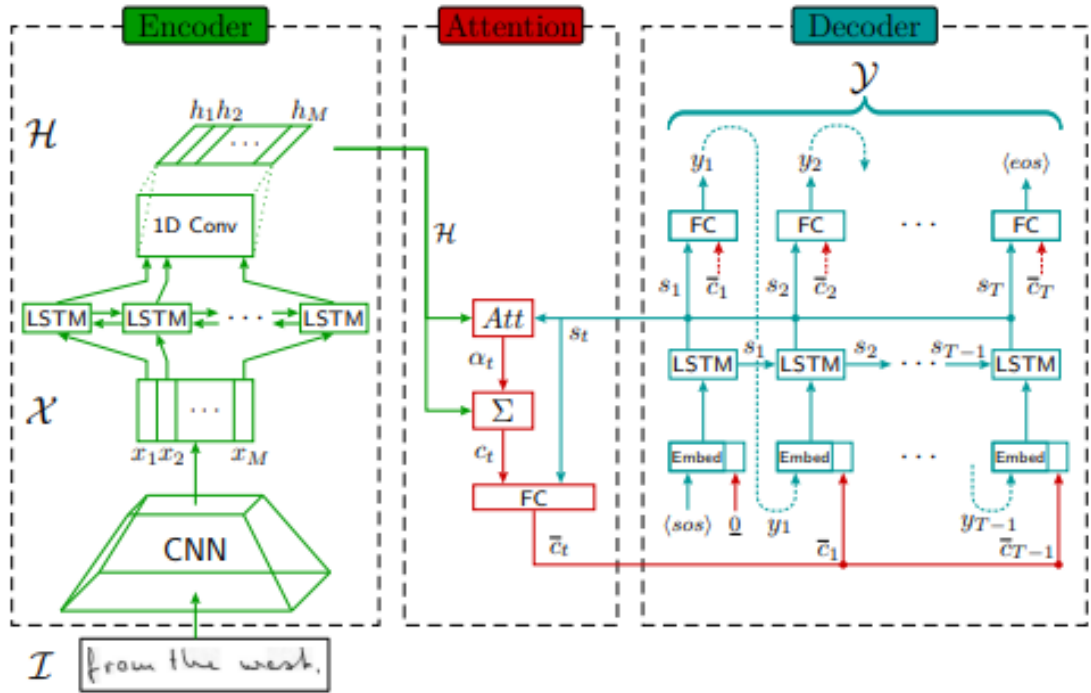


Figure 2.7: The Seq2Seq model for attention has a general architecture. An input image is transformed into a series of fixed feature vectors by the encoder. One character at a time is output by the decoder as the output sequence. It uses an attention mechanism at each time step to create a context vector from the time-dependent decoder hidden state and the encoded feature vectors. The decoder's output for the current time step is generated using this context vector. The context vector and embedded output are concatenated to form the decoder's subsequent input [44].

The Figure 2.7 gives an overview of the proposed architecture.

On the IAM dataset, the model gave the minimum CER of 4.87 and on the BOZEN dataset it gave the minimum CER of 4.66. For future work, the authors of the study aim to use the pre-trained model for the improvement of the encoder part of the model architecture.

2.7.2 Drawbacks

The suggested model treats handwriting recognition as a Seq2Seq problem but uses LSTM layers in the decoder. Even though LSTMs were introduced to cater for vanishing

gradients problem, they still fail for larger sequences of tokens. Transformer decoders on the other hand completely eradicate the problem of vanishing gradients even for sequences up to thousand tokens.

2.8 Transformer Based OCR Model

By utilising the Transformer ([50]) structures, recent advancements in text recognition ([27]) have shown notable gains. The self-attention is constructed on top of CNN backbones as encoders to interpret the text image, although current approaches are still relying on CNNs as the backbone. To increase overall accuracy for decoders, Connectionist Temporal Classification (CTC) ([6]) is frequently combined with an external language model on the character level. Despite the hybrid encoder/decoder method's considerable success, there is still much opportunity for improvement using pre-trained CV and NLP models: 1) In present methods, the network parameters are trained from scratch using artificial or human-labeled datasets, leaving large-scale pre-trained models unexplored. 2) It is simple to investigate whether pre-trained image Transformers can replace CNN backbones as image Transformers gain in popularity ([24]), especially the recent self-supervised image pre-training ([25]). In the meantime, pre-trained image Transformers can be used to collaborate with pre-trained text Transformers in a single framework on the text recognition.

2.8.1 Methodology and Results

For the task of effective handwriting recognition, the authors in [42] proposed an end-end model that is based on Transformer OCR.

First of all, the input images are divided into patches. After being divided into patches, they are then concatenated. Once concatenated, they are then fed flattened. By doing this, an embedded matrix is obtained. This embedded matrix is then fed into Transformer Based Encoder.

A pre-trained Image Transformer is used as an encoder and a pre-trained text transformer is used as a decoder. The handwriting task is treated as a sequence to sequence problem by this TrOCR model. the weights are pre-trained on the the Image net and the encoder is then initialized by those pre-trained weights. For the decoder part, the

weights are pre-trained on the wiki-text and the decoder is initialized by those pre-trained weights.

A minimum CER of 2.89 is obtained by the proposed model on the Syntetic and IAM Datasets. The authors of this study proposed that their model can be extended and tested to the task of multi-lingual text recognition.

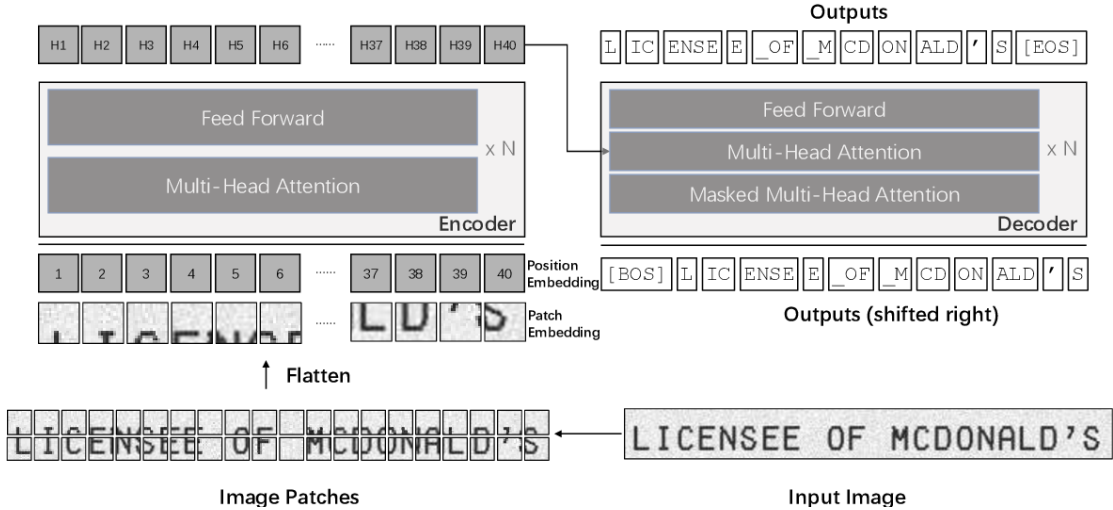


Figure 2.8: An encoder-decoder model for TrOCR is created using a previously learned image. Transformer serving as the encoder and a text Transformer serving as the decoder [42].

An overview of the architecture is shown in the Figure 2.8.

2.8.2 Drawbacks

Since this architecture involves the use of a pre-trained transformer encoder like DIET [49] and a pre-trained transformer decoder like Roberta [43], so this makes the architecture computationally complex.

The architecture of the proposed model has been designed in such a way that it is heavily reliant on the pre-training and gives good results on the IAM data after the process of fine-tuning.

As mentioned before that the decoder is initialized by the weights that are pre-trained on the wiki-text and since the wiki-text is the English language, so this model becomes ineffective and gives poor results when performing the task of Urdu handwriting recog-

dition.

2.9 Transformer Based Light Encoder Decoder Architecture

Drawback of Transformer Models

The use of the transformer models and architectures have been producing outstanding results in the field of Natural Language Processing. However, one of the main difficulties of employing the Transformer Models is that substantial amounts of training resources are needed for them to be effective.

A Transformer based light encoder-decoder architecture was presented by the authors in [31].

2.9.1 Methodology and Results

This Transformer based light encoder-decoder architecture has the advantage that it can be trained effectively on the small datasets without having the need for the additional data resources.

In this proposed architecture, the number of training parameters were reduced from 100M to 6.9M.

In this architecture, the encoder draws inspiration from both Transformer layers and conventional CRNN [50]. Five convolutional blocks make up the encoder's first section, which is used to extract visual features from images. Each convolutional block, with the exception of the final one, is made up of a 2D convolutional layer with a kernel of size 3x3, a stride of 1, and no padding. To more closely mimic the shape of a character, the last convolutional block has a 42 kernel size [[20], [22]]. The decoder is based on the Transformer model and it acts as a Language model for this architecture.

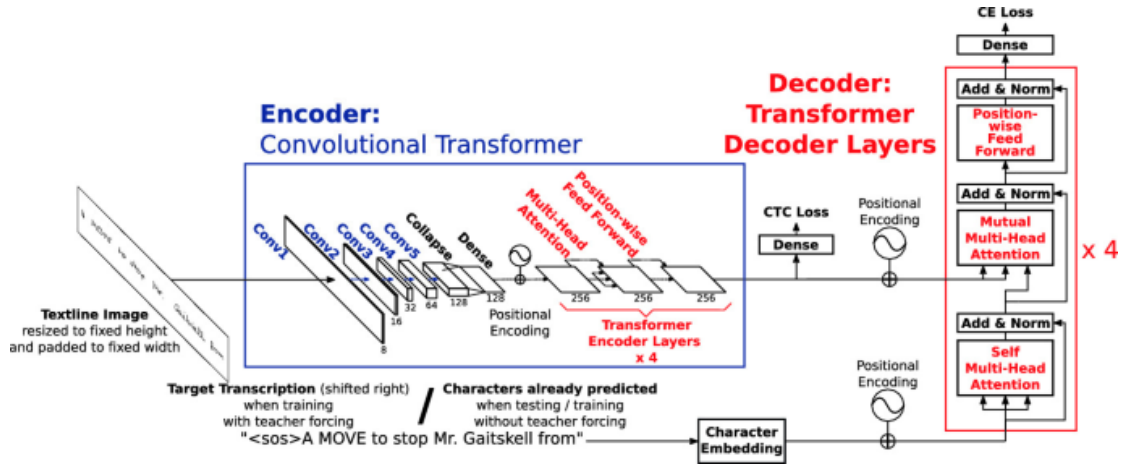


Figure 2.9: The decoder-encoder architecture based on transformers. The architecture consists of a Transformer-based decoder and an encoder that combines convolutional layers with Transformer layers [31].

An overview of the proposed model's architecture has been shown in Figure 2.9.

In this study, the models were trained with a hybrid loss which combined the following od the two losses:

- Connectionist Temporal Classification (CTC) loss [6]
- Cross-Entropy (CE) loss

IAM dataset was used for the testing of the proposed model. The model was tested with the additional data as well as without the additional data so that the efficiency of the model can be compared. On the IAM dataset, the model achieves a 5.70% CER. According to the authors, their proposed methodology and the model can be applied to the historical documents.

In addition to the training dataset, synthetic data is also employed to better analyse the model. Synthetic data is used by recent Transformer topologies in the field, and they show significant advances as a result [[32], [29]].

2.10 Drawbacks of Architectures Discussed

Some of the techniques mentioned prior use CTC loss which ignores the importance of a language model in the task of handwriting recognition. This leads to a use of

a separate language model like n-grams. Moreover, some techniques also model the task of handwriting recognition as a Seq2Seq task. This leads to the use of encoder decoder architectures. The authors either use models like LSTM or GRUs which fail to capture long term dependencies or they use transformers without caring for the n^2 computational complexity overhead. the methodology proposed in this thesis includes the use of a transformer with a convolutional block to reduce the spatial resolutions of the text image and hence catering for the n^2 complexity overhead. Moreover printed text along with handwritten text was also used for the training of the architecture which aids the model in learning a diverse language model for the task specific language.

Methodology

3.1 Unconstrained Off-Line Urdu Handwriting recognition as a Seq2Seq modeling task

In the proposed methodology of the thesis, the Unconstrained Off-Line Urdu Handwriting recognition is treated as a Seq2Seq modeling task [50, 30]. The architecture is inspired by the task of Neural Machine Translation as the input of varying length in one language is encoded by an encoder and then the decoder acts upon to generate a decoded output of varying length in some other language. The same is true about the task of handwriting recognition as the aim is to encode the input which is an image and then decode it to produce the digitized output which is the text. Recent developments in Seq2Seq models show that the Transformer architectures give state of the art results. This is due to the presence of attention layers in the Transformer module which are inspired by the working of a human brain. For instance, say that a human wants to translate a book written in a certain language to some other language. It would be more convenient for the interpreter to do so sentence by sentence rather than reading the whole book first and then translating it to a new book. This process can be thought of as attending to the relevant parts of the book while generating a translation for it. The same way the attention layers in the transformers attend to the relevant parts of the encoded sequence while decoding it and generating the corresponding output.

3.2 Encoder-Decoder Transformer Architecture

Inherently a Transformer is an encoder decoder architecture. The Transformer encoder and decoder along with the attention mechanism allows not only to capture dependencies between the input handwritten text image and the output text but also allows to learn a language model for the particular language. Language modeling is the modeling of probability of next word given previous sequence of words. This allows the model to learn the language itself. So the use of Transformer architecture caters for the dependencies between the input image and the output text along with the inherent learning of a language model without the need of a separate language model like N-grams.

Such encoder decoder architectures work in an autoregressive fashion. During the training of the autoregressive models the teacher forcing technique is used where the shift-right ground truth is fed into the decoder part. This aids in faster convergence during training.

The testing or inference phase of such models is different from training as now the ground truth is not present and the output is generated word by word or token by token. The probability of the next word is calculated given the sequence of previous words and the feature vector from the encoder.

Techniques like greedy decoding and beam search during the inference phase. Beam search gives better results than greedy decoding because it searches for the most probable sequence given a certain hyperparameter of beam width whereas greedy decoding looks for the next most probable word given the sequence of previous words and the feature vector from the encoder. The sequence produced by greedy decoding might not be the most probable sequence and hence it usually results in lower performance than beam search. Although beam search is more accurate, it is computationally expensive for a large hyperparameter of beam width and is equivalent to greedy decoding if beam width is one.

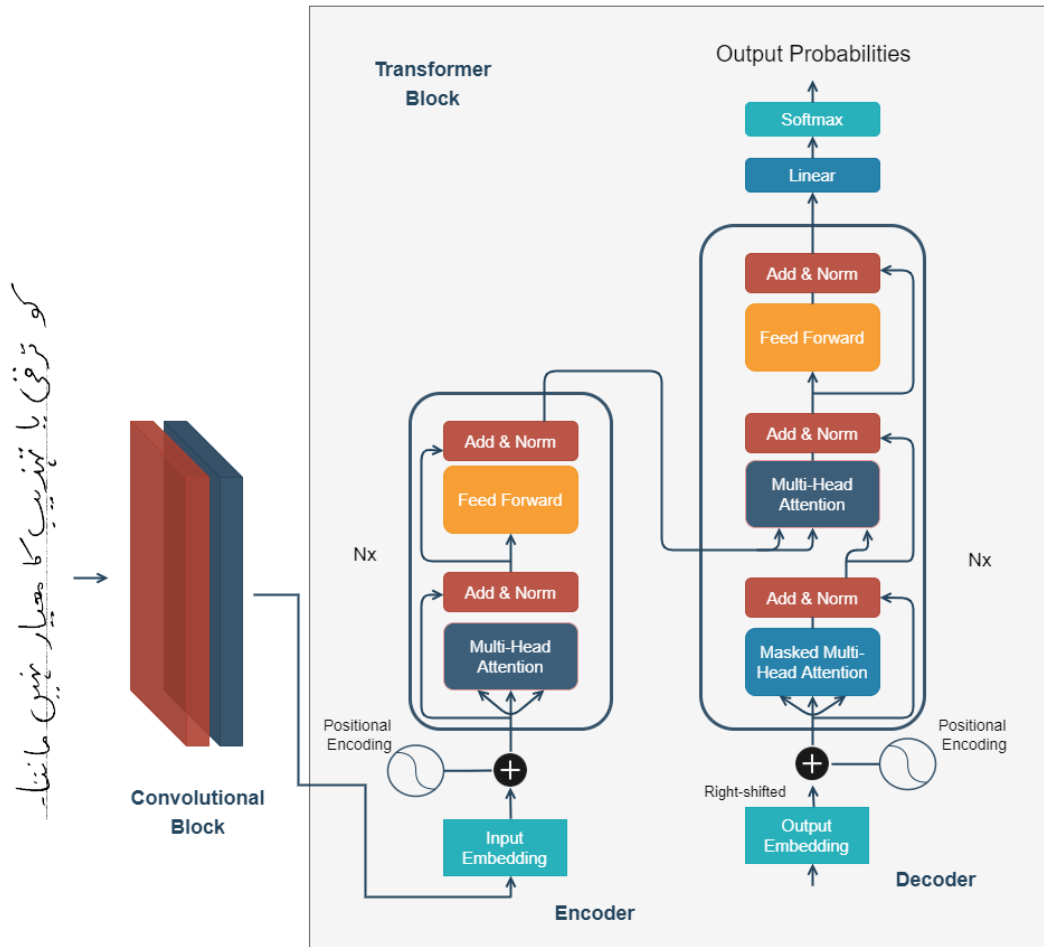


Figure 3.1: The Encoder-Decoder Transformer Architecture during training phases consisting of the convolutional blocks followed by a full transformer.

3.3 Conv-Transformer Architecture

Despite being a huge success, Transformer architectures are painfully slow for very long sequences. If we have a sequence length n then the multi-head attention layers in the Transformer have n^2 computational complexity[47]. To address this issue the Conv-Transformer architecture is proposed (refer to Figure 3.1). Convolutional layers before the complete vanilla transformer act to reduce the spatial resolution of the input Urdu text image and generate feature maps which are then treated as an input to the encoder of the transformer. These features are further encoded by the Transformer encoder and then the decoder acts to generate the relevant sequence of digitized or editable Urdu

text. The whole architecture being Seq2Seq also works in an autoregressive fashion and beam search is used for decoding the output sequence. Also the model works on character level rather than word level or byte pair encodings. This is due to the small dataset that we have for Urdu handwriting recognition.

The following subsections explain the individual segments of the proposed Conv-Transformer architecture. Subsection 3.3.1 discusses the working of the Convolutional Block along with the layers used. Subsection 3.3.2 discusses the significance and details of the Transformer module. Subsection 3.5 explains the implementation of the Beam Search algorithm.

3.3.1 Convolutional Neural Networks

Convolutional layers are stacked at the start in order to extract features from the input handwritten text image[14]. These features have low spatial resolution compared to the input image. Given a gray scale input image of spatial dimensions $W \times H$ (where W is the width and H is the height), the convolutional blocks act to reduce the dimension to $S \times 1 \times d$ (where S is the width of the feature map and d is the depth). The output feature map is then reshaped to $S \times d_{model}$ and fed into the Transformer module where S is treated as the sequence length and d as d_{model} which is the embedding dimensions.

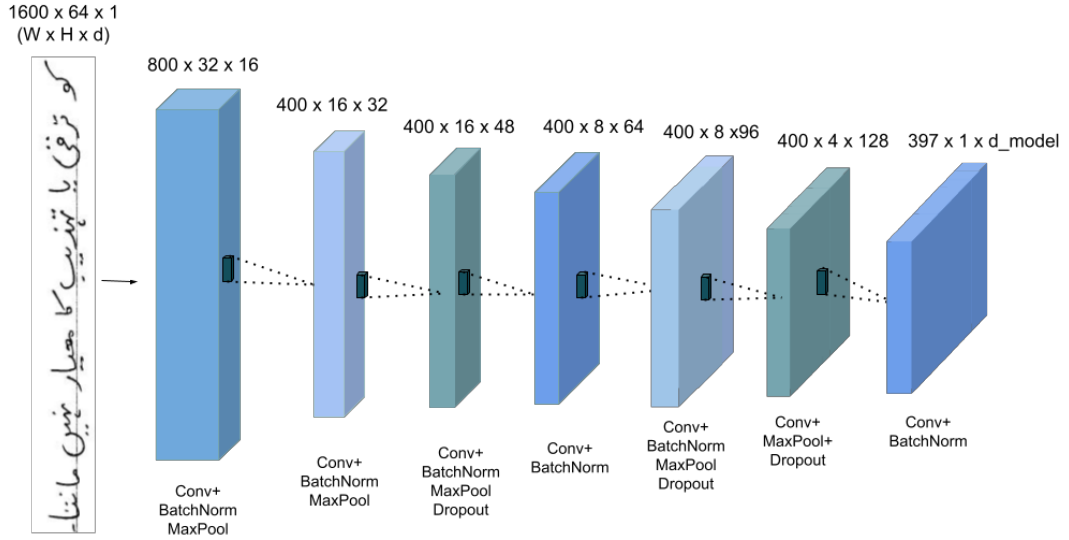


Figure 3.2: The custom CNN layers that are used as feature extractors in the proposed architecture.

The configuration¹ of each layer of the proposed custom CNN block is given in the table 3.1 and Figure 3.2.

3.3.2 Transformer

The Transformer architecture introduced in [50] relies completely on the attention layers to capture long and short term dependencies. It completely replaces the recursive architecture of models like RNNs, LSTMs and GRUs which struggle with long sequences due to the vanishing gradient problem[30]. For RNNs the back propagation occurs through time due to which the gradients might vanish if the sequence is too long. LSTMs were introduced to cater for the vanishing gradients but still they suffer from vanishing gradients for very large sequences. In Transformers the gradients hop in one step due to the simple trick of matrix multiplication for the attention layers. The encoder side of the transformer applies a self attention mechanism in which every position of the input embeddings attends to every other position whereas on the decoder side causal attention is used. Causal attention restricts the tokens to attend to previous positions or tokens only. Attention mechanism is also applied between the encoder and decoder

¹For details of each Conv. block's dimension, please refer to Figure 3.2

Table 3.1: The table displays the configuration of each layer of the proposed custom CNN block

Layer	Configuration
Conv	1→16, 3×3
BatchNormalization	-
LeakyReLU	-
MaxPooling	2×2
Conv	16→32, 3×3
BatchNormalization	-
LeakyReLU	-
MaxPooling	2×2
Conv	32→48, 3×3
BatchNormalization	-
LeakyReLU	-
Conv	48→64, 3×3
BatchNormalization	-
LeakyReLU	-
MaxPooling	(1,2) (2,1)
Dropout	0.2
Conv	64→96, 3×3
BatchNormalization	-
LeakyReLU	-
Conv	96→128, 3×3
BatchNormalization	-
LeakyReLU	-
MaxPooling	(1,2) (2,1)
Dropout	0.2
Conv	128→ d_model , 3×3
BatchNormalization	-
LeakyReLU	-

so that the model attends to the relevant pixels of the input image while generating

the output text. The attention between an encoder and decoder is called multi-head encoder-decoder attention. The vanilla transformer architecture is used in the suggested model.

3.4 The Attention Mechanism

The attention mechanism consists of 3 input matrices namely Queries (Q), Keys (K) and Values (V). In the self attention part of the encoder, these matrices are the three different representations of the input embeddings after passing the embeddings through three separate linear layers. The causal attention takes place on the decoder side and these matrices are the representations of output embeddings of the text being generated. For the encoder-decoder attention, the output of the encoder undergoes two transformations via linear layers to produce K and V . Q comes from the decoder and the attention between these matrices takes place. The Attention scores are calculated by the dot product of these matrices as shown in the Equation 3.4.1.

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (3.4.1)$$

The dot product of Queries and keys is scaled by a factor of square root d which is the depth of the embeddings. Softmax converts the result into probabilities or attention weights. Dot product of these weights with V matrix yields the final output which is the new representation after attending to relevant positions of the sequence.

The output feature map from the convolutional block of dimension $W \times H$ is fed as the input embedding to the transformer encoder. The positional embeddings are added with the input embeddings and then passed to the multi-head self attention layers of the encoder. The encoder yields K and V after linear transformations for the decoder at the end. Decoder works differently for training and inference. During training, teacher forcing is used and the right shift ground truth embeddings are passed to the decoder. Same as encoders, the positional embeddings are added but this time the embeddings are fed into the Masked Multi Head attention layer which performs causal attention. After that Q from the decoder and K and V from encoder undergo multi-head encoder-decoder attention.

The encoder and decoder of Transformer has multiple blocks. After the decoder blocks

a linear layer and finally the softmax layer is present to predict the ground truth tokens while training and autoregressive prediction is used during inference. The softmax projects the decoder embedding dimensions to the vocabulary size dimensions² in order to predict the tokens or characters as in this case character level Urdu handwriting recognition is being performed.

²To avoid the contradiction with the value (V) of the transformer equation, the vocab size is being represented as small v .

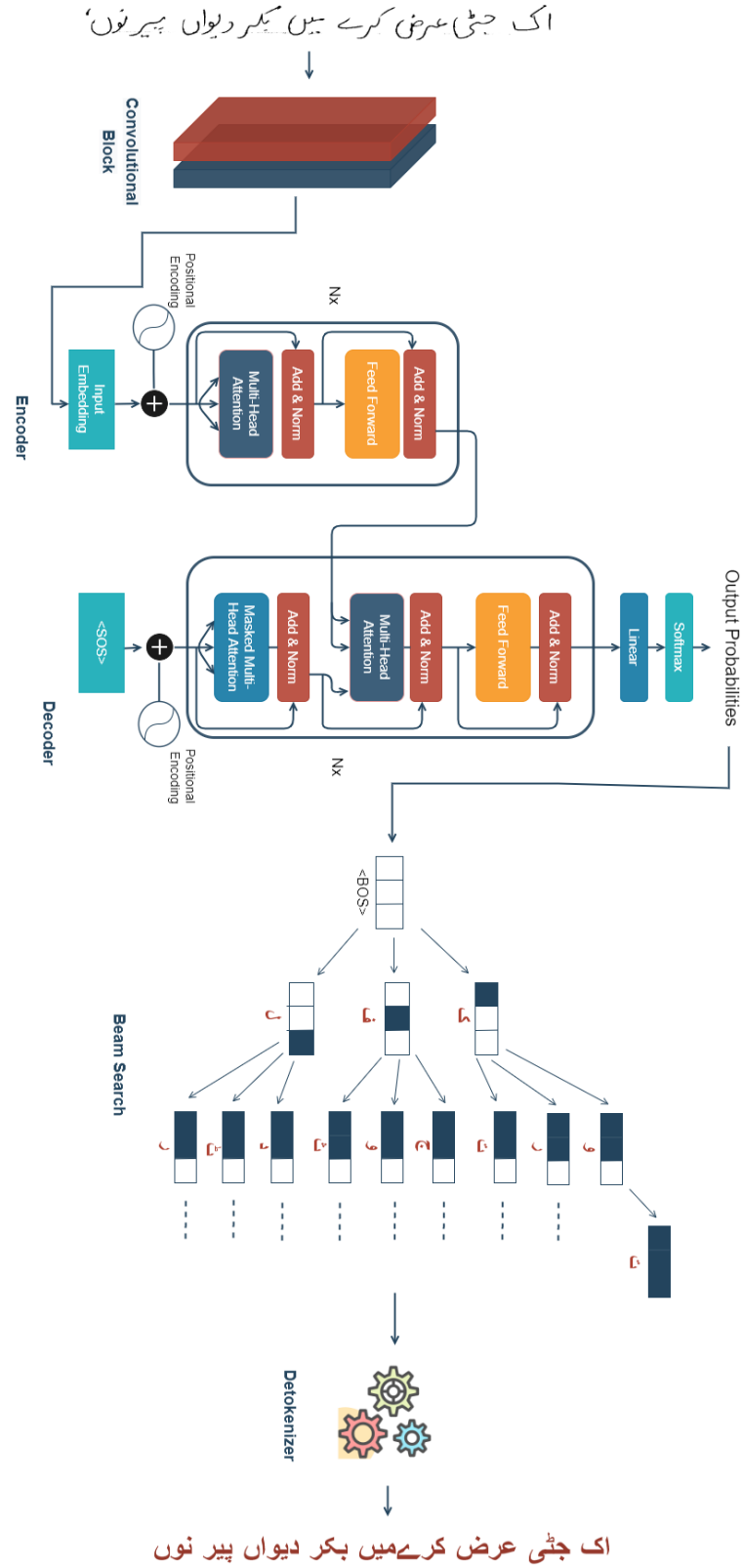


Figure 3.3: An overview of the model architecture during the testing phase consisting of a transformer, a beam search method and convolutional blocks.

3.5 Beam Search

During the inference stage, the proposed architecture works in an autoregressive fashion (refer to Figure 3.3). Given input of previous characters and the encoded features from the Convolution block and the transformer encoder, the decoder is expected to predict the next character. The softmax layer outputs the probability of all the possible next characters. An initial solution to the decoding problem at hand would be to take the highest probability character on which the model is most confident and then append it with the already predicted characters and then feed the sequence again to the decoder for next prediction and so on until an [EOS] token is received. This approach is called greedy decoding and it might not be the best solution as taking the character with the highest probability at every step might not reveal the most probable sequence. Finding the most likely sequence rather than the token with the highest probability at each step is the ultimate objective.

3.5.1 Computation of the Sequence Probability

The probability of the sequence can be calculated by multiplying the probability of all the tokens or characters received at each step to construct the sequence. With the greedy decoding approach the multiplication of probabilities might not yield the best result as at each step taking the most probable token is not the best option since choosing that character might cause the next most probable characters to have low probabilities. So we need a solution that would search all the possible sequences and then return the most probable one. This solution, although very attractive, is computationally quite infeasible. It would be an NP-complete algorithm and impractical to implement.

Not only the computational time of this algorithm increases exponentially but also the vocabulary might range from tens of thousands or millions tokens in some cases which would lead to the occupation of a lot of space by the algorithm. Given a vocabulary size V and an index location n , there would be v^n partially complete sequences upto that index location³. In order to predict the next token at location $n + 1$ we would have to feed all these sequences into the model and get the probability for the next token for each sequence and then select the sequence with the highest probability and repeat the process until an EOS token is received for the sequence with the highest probability.. The

³<https://www.width.ai/post/what-is-beam-search>

computational resources and time required to run this algorithm increase exponentially for each iteration and hence it is infeasible. This algorithm is however the only one which guarantees that the model will output the sequence with the highest probability.

3.5.2 Beam Search Working Mechanism

Beam search comes to the rescue by introducing heuristics to make the same algorithm computationally more tractable. Rather than keeping all the v^n sequences for the index location n for the prediction of token at $n + 1$ we can always keep top ' k ' most probable sequences at each step. The hyperparameter ' k ' is introduced, which is also called the beam width. In practice this algorithm works very well for a reasonable value of ' k '. It is to be noted that for the value of ' k ' equal to one, beam search becomes a greedy decoding algorithm. For ' k ' sequences at a particular step the model outputs ' k ' tokens for each of the ' k ' sequences based on the ranking by highest probabilities among the V number of tokens. This means that at every step we have k^2 possible sequences after the prediction by the model. Out of the k^2 sequences, we take top ' k ' sequences again, ranked on the basis of sequence probabilities. The following equation 3.5.1 determines the probability of a sequence..

$$\frac{1}{n^\alpha} \sum_{i=1}^n (P(y_n)) = \text{Sum of } \hat{y} \text{ of length } n \quad (3.5.1)$$

Here ' n ' represents the length of the sequence, the ground truth label is ' \hat{y} ' and ' α ' is the penalty criteria for longer sequences which is set to 0.7.

Here log probabilities are applied to cater for numerical underflow that might arise due to multiplication of probabilities and hence the log probabilities are calculated and summed at the end giving the same effect without the danger of numerical underflow. This might arise a new problem with sequences with more tokens being more probable compared to sequences with less number of tokens. Any sequence, out of the top k , receiving an '*EOS*' token would halt and not be expanded further but other sequences might get priority over the shorter one due to the log probabilities being summed and naturally longer sequences might get higher probabilities. To cater this issue the parameter ' α ' is introduced which would penalize longer sequences and hence the beam decoding algorithm would give a reasonable output.

The mathematical formulation for the beam search used in the proposed methodology is given in Eq. 3.5.1. The technique used here just computes $k \cdot k$ probabilities, as opposed to traditional implementations that typically compute $k \cdot v$ probabilities at each index and look for the best ‘ k ’⁴.

In addition, the beam search is carried out on a character level rather than a word level because of the architecture of the model described in the thesis and limitations of the dataset. A language’s unique characters and a few ligatures rather than its whole lexicon make up the vocabulary size for character-level recognition, which is typically substantially lower. As a result, beam search performs much better and gets closer to the NP-complete algorithm. This is because even moderate total beam values are much closer to the actual vocabulary size and are therefore much more efficient than when the total number of beams is relatively small.

⁴<https://www.baeldung.com/cs/beam-search>

Implementation and Results

4.1 Dataset

Table 4.1: The UHWR dataset is partitioned into training, validation, and testing splits. The dataset only has about 10,000 text lines

Total no. of samples in UHWR dataset	10,606
No. of samples used for training	8,484
No. of samples used for testing	1,061
No. of samples used for validation	1,061

The NUST-UHWR dataset [51] was used for the experimental setup in this study. The dataset includes images with a single sentence or lines of Urdu Handwritten text and associated text labels. The images are distinctive and include diverse text written in different ways. As shown in Table 4.1, the UHWR dataset was partitioned into training, validation, and testing splits. The dataset only has about 10,000 text lines, which isn't enough to train an adequate handwriting text recognition model. The traditional way of adding training samples is to use data augmentation; however, it is claimed that data augmentation methods are ineffective for training a text recognition system.

In [45], it is demonstrated how ligature coverage can increase a text recognition system's accuracy. In Arabic-like scripts, where there are many ligatures, this is particularly true. It is also proposed that printed text corpora can be useful for developing a handwriting

recognition system since the basic writing strokes of Urdu remain the same whether they are printed or handwritten. The efficacy of the proposed hypothesis is demonstrated by comparing the outcomes of the same model using both ways - 'conventional data augmentation and "printed text along with handwritten dataset".

Table 4.2: This table provides the statistics for the two printed text datasets (used only for Training): UPTI 2.0 dataset [51] and Urdu Ticker Text dataset [46].

Total no. of samples in Ticker dataset	19,437
Total no. of samples in UPTI-2 dataset	1million

During the training phase, two printed text datasets were appended with the handwriting dataset to improve the results of the proposed handwriting recognition architecture. The first is the Urdu Ticker Text dataset [46], which was recently proposed, and the second is the UPTI 2.0 dataset [51]. Table 4.2 provides the statistics for these two printed text datasets.

The proposed architecture was also trained and evaluated on the ADAB dataset [26], which is an Arabic handwriting dataset, to further assess the effectiveness of the suggested model on related languages. The data includes text that is scripted in Arabic and names of 937 Tunisian towns and villages were utilized to produce this data¹. For training, testing, and validation, the splits utilized for this dataset are in the proportion 80:10:10. These splits were decided upon at random.

A few preprocessing techniques were carried out before inputting the images into the model. Data augmentation approaches were also utilized to boost the dataset's diversity and sample size.

4.2 Data-Preprocessing and Augmentation

The dataset needs to be formalized as a first step. In a nutshell, the data is converted into an acceptable format. It lessens the CPU bottleneck during model training. All of the images were converted to gray scale and resized to a uniform height of 64 *px*. The original aspect ratio was not disrupted. For batching, zeros were padded to the image

¹<https://iee-dataport.org/open-access/adab-database>

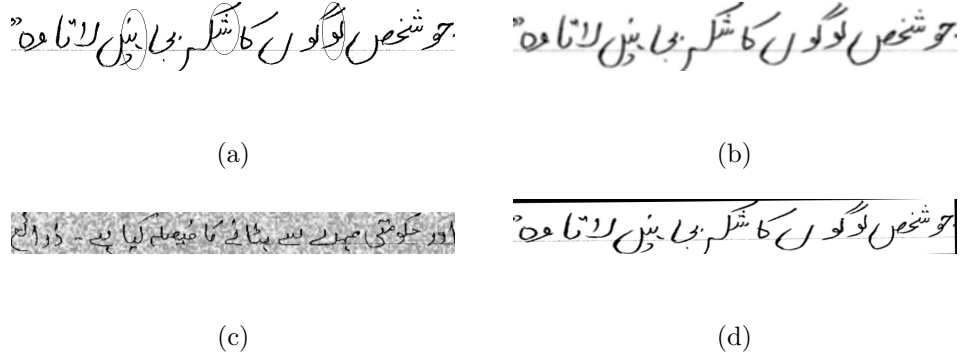


Figure 4.1: A variety of data augmentation approaches used. Original image (a), artifacts compressed, and blur effect (b). (c) degradation, soft-noise, and colour inversion. textbf(d) The effect of squeezing, rotation, and rescaling.

width up to the maximum length hyper-parameter (set to 1600).

Data augmentation was used during model training to add variation to the training datasets. Any machine learning or deep learning model can benefit from data augmentation in terms of performance and output. There were 10 augmentation functions in total[46]. Four of them are color-based enhancements, while the remaining five are shape-based. Shape-based augmentations are used to distort the image into several dimensions. Color inversion, padding, color correction, soft noise, mild blurring, squeezing, degradation effect, axis rotation, artifact compression, and rescaling the individual image chunks are some of the augmentation functions. In Figure 4.1 a few examples of data augmentations are displayed.

4.3 Implementation Details and Hyperparameters

PyTorch was used to implement the proposed architecture. As indicated in Table 4.3, the CNN was constructed with a leaky-ReLU activation function and batch normalization for faster convergence as it is one of the standard practices². Only max pooling layers are responsible for reducing the spatial resolution of feature maps, while padding was used to maintain the spatial resolution in convolutional layers. 3 encoder and 3 decoder blocks for the transformer were utilized since this configuration produced the best results in the least computational time. Different settings were tested with various encoder

²<https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/>

and decoder layer counts. There was no performance increase when the encoder and decoder layers of the transformer were set to a number higher than 3. The output was transformed to the shape $(B \times Sq \times v)$ using a linear layer after the transformer, where ‘ B ’ stands for the batch size, ‘ Sq ’ for the output sequence length, and ‘ v ’ for the vocabulary size. The vocabulary size is the total number of characters encountered in the training split of the dataset plus the tokens like *BOS* (Beginning Of Sentence), *EOS* (End Of Sentence), *PAD* (padding token) and *UNK* (Unknown character) as character level handwriting recognition is being performed. For training and validation, *Softmax* was utilized before cross-entropy loss. A single Nvidia RTX 3080 GPU was used to train the proposed model. While using a 16-batch size, the length of the right-shifted output sequence was padded with pad tokens until the batch’s longest sequence. For the architecture’s training, the Adam optimizer was utilized. With the hyper-parameters betas (0.9, 0.98) and epsilon $1e - 9$ of the Adam optimizer, a learning rate of 0.0003 was employed. Other learning rate settings reduced the effectiveness of the training by diverging the loss for higher learning rates or by slowing convergence for lower learning rates.

4.4 Experiments Performed

The experimentation consists of combining multiple datasets with augmentation strategies to examine the effect of UHWR testing and validation splits on CER (Character Error Rate). To increase the variety of the Urdu language, handwritten along with printed Urdu text databases were combined. This was carried out to assist the devised architecture in learning a more diverse language model. Results shown in Section 4.5 demonstrate that printed text helps the model capture more linguistic variation in Urdu. First, only the UHWR train split was used to train the suggested model. Next, the entire Ticker printed text along with full UPTI-2 training set was included to improve the analysis

The three training configurations are as follows:

1. UHWR training split
2. UHWR training split + Ticker
3. UHWR training split + Ticker + UPTI-2

Table 4.3: This table compares Conv-Recursive [51] the present state-of-the-art model for Urdu handwriting recognition, and the proposed model using CER of UHWR valid and test splits. Only the UPTI-2 dataset was used to train the n-gram Language Model (LM) for the conv-recursive model.

Dataset (used for training)	Conv-Recursive Model		Conv-Transformer Model (Proposed)	
	Valid CER	Test CER	Valid CER	Test CER
UHWR Train Split	7.25% (no LM)	7.35% (no LM)	6.0%	6.4%
UHWR Train Split + Ticker	8.15% (no LM)	8.3% (no LM)	5.35%	5.5%
UHWR Train Split + UPTI-2	5.28% (with LM)	5.49% (with LM)	5.12%	5.34%
UHWR Train Split + Ticker + UPTI-2	5.27% (with LM)	5.5% (with LM)	5.14%	5.31%
ADAB (Arabic) Dataset	5.2% (no LM)	5.0% (no LM)	2.3%	2.0%

The transformer does certainly acquire a language model in addition to digitizing the input image, as evidenced by the fact that the Character Error Rate on UHWR testing and validation splits decreased with the inclusion of more data from varied distributions of printed text.

Experiment with the ADAB dataset was also performed to see how well the suggested method worked for offline handwriting recognition in other inflectional languages, such as Arabic.

4.5 Results

After training the model on UHWR training split, Character Error Rate (CER) of 6.4% and 6.0% were obtained on UHWR testing and validation splits, respectively. The combination of printed Urdu datasets like Ticker and UPTI led to the results that were even better with the best CER of 5.14% on validation and 5.31% on the testing set (see Table 4.3 for details). These results show that the new state-of-the-art on the task of Urdu Handwriting Recognition is achieved. Combining these text datasets diversifies the representation of the Urdu language. The proposed architecture captures this variability, and the transformer also learns a diverse language model.

Also the suggested methodology performed better in the Arabic offline handwriting recognition experiment than the model proposed by Zia et al. in [51]. On the validation and test splits of the ADAB dataset, the Character Error Rates (CER) for the proposed

Conv-Transformer model of 2.3% and 2.0% were obtained, respectively (see Table 4.3 for comparisons). It might be inferred from these results that the proposed architecture is capable of recognizing the intricate Nastaleeq scripts used in both Arabic and Urdu languages.

TrOCR [42], the current state of the art on the task of English handwriting recognition, was also employed on UHWR and ADAB dataset which led to very poor results. TrOCR is pretrained on datasets like WikiText which is in the English language. Fine tuning the model on Urdu or Arabic data gives very poor results which are not comparable to the state-of-the-art. TrOCR is computationally very expensive model to be pretrained from scratch on Urdu datasets like UPTI-2 or Ticker and then fine tuned for handwriting task. Due to lack of sufficient resources it was out of scope for this thesis. On the other hand Conv-Transformer model gives state-of-the-art results being computationally efficient at the same time.

For further analysis, IAM [5] English Dataset was used for the comparison between the three models namely TrOCR [42], Conv-Recursive [51] and Conv-Transformer (proposed). Even though the task of English handwriting recognition is out of scope for this thesis, the models were compared in order to test the generalization capabilities. TrOCR decoder being pretrained on WikiText [16] and encoder trained on ImageNet [10] dataset gives the state-of-the-art results on IAM dataset with CER of 2.89%. Conv-Recursive model gives the CER of 5.68% and Conv-Transformer being slightly better than Conv-Recursive with CER of 4.2%.

Discussion

5.1 Comparison with Conv-Recursive Architecture

The convolutional recursive design put forth by Zia et al. in [51], which is the state-of-the-art in Urdu handwriting recognition, is comprehensively compared to the proposed architecture. The results shown in Table 4.3 demonstrate that the state-of-the-art was outperformed significantly.

The architecture suggested in Zia et al [51] combines a convolutional recursive deep learning model which works at the character level with a discrete word level n -gram language model. However, proper justification or methodology of how these two models work in conjunction to produce the results is lacking. It is possible that the word level n -gram language model negates or overrides the predictions of the character level deep learning model at the end, leading to a reduction in CER from 7.42% without LM to 5.49% with LM on the test set. Since the text uses real Urdu language vocabulary, replacing the character level results with an n -gram word level language model would undoubtedly produce better results as all the words would belong to the vocabulary and n -gram would work as a spell corrector.

The paradigm suggested by [51] with a separate n -gram would not work given a handwritten text with random letters not formulating a word that could possibly be present in an Urdu dictionary. The presence of a Transformer in the proposed Conv-Transformer architecture inherently models the task of Handwriting recognition as the probability of next character given the sequence of previous characters and feature map excerpted from the input image. The mathematical formulation would be $P(n_c|p_c, c)$. Here, ' n_c '

stands for the next character, ‘ p_c ’ for the previous sequence of characters, and ‘ c ’ is the feature map extracted from the input image through convolutional layers. This leads to the learning of two tasks concurrently, i.e. language modeling at character level and digitization of the input image. Table 4’s comparison of CER for the two models demonstrates that the proposed model exceeds the state-of-the-art without the requirement of an additional language model. Additionally, the model performs noticeably better than the state of the art as additional data is introduced to the UHWR dataset, even though it comes from a different distribution of printed text. This is because more data allows the transformer to acquire a stronger language model, which improves performance.

5.2 Comparison with Google’s Vision API

In order to recognize Urdu handwriting¹, Google Vision recently released an experimental API. This API was validated and tested using splits from UHWR. The outcomes were subpar compared to state-of-the-art. On UHWR testing and validation splits, the Google Vision API provided CER values of 27.8% and 26.5%, respectively. Due to the possibility that the Google vision API model was not developed using the UHWR dataset, the vision API was also evaluated using a set of random images and compared with the outcomes to those obtained by Zia et al [51] and the proposed architecture.

¹<https://cloud.google.com/vision>



Figure 5.1: Results from the Smoke study are shown in the figure. Results from the Google Vision API are shown in Figs. (a) and (b), with CER values of 7.8% and 5.7 percent, respectively. Results from Zia et al [51]’s model with CER of 5.2 percent and 5.7 percent, respectively, are shown in Figs. (c) and (d). Results from the suggested model, which has a CER of 2.6% and 0%, are shown in Figs. (e) and (f)

5.3 Smoke Testing on Random Images

To test the generalization potential of the Zia et al [51], Google vision API, and the proposed model, random images of Urdu handwriting were used. The images were gathered at random by having a few people write a text line or sentence in Urdu on a blank white paper and scanning the paper later. For each model’s prediction, two test images were chosen as input. The qualitative smoke testing results are shown in Figure 5.1, along with a comparison of the Google vision API, the model developed by Zia et al., and the proposed Conv-Transformer. These findings clearly show that the CER provided by Conv-Transformer on these images was the best. We may infer that, despite complex writing styles or overlapping letters in the sample image, the Conv-Transformer architecture generalizes well on handwritten text since the images for smoke analysis were chosen at random.

5.4 Ablation Studies

The Conv-Transformer was also subjected to an ablation study in order to assess the role of the transformer decoder in developing a language model that decreases CER on the UHWR test set. Additionally, an ablation study was performed to check the effectiveness of the architecture’s performance with regard to the role that the Convolutional layers have.

5.4.1 Ablation Study - Only encoder CTC

The decoder layers of the Conv-Transformer were entirely discarded, and the UHWR dataset was utilized to train the Convolutional block and the transformer encoder alone. The same CTC loss was applied as in Zia et al. [51]. This model is comparable to Zia et al. [51] with the exception that GRU-style recurrent neural networks have been replaced by transformer encoders. From validation and testing outcomes, it was clear that the model was behaving in this scenario similarly to Zia et al. [51] without n-gram language modeling. A CER of 7.28% was obtained on the UHWR validation split and 7.4% on the test split from the convolutional block and transformer encoder. With the modification in the loss function, i.e. CTC loss, the same hyper parameter values were employed as in the case of the complete conv-transformer architecture.

5.4.2 Ablation Study - Encoder decoder

In order to evaluate the influence of the Convolution Network as a whole prior to a full transformer, Conv-Transformer architecture was also evaluated by omitting the Convolutional Block from it. After positional embeddings, the image was directly supplied to the transformer encoder. Better outcomes were achieved in this scenario than Zia et al. [51], although it was more difficult for the model to converge during training. On the UHWR validation and test splits, the CERs of 6.97% and 7.1% were obtained respectively. A vanilla transformer without convolution layers can be utilized for training and testing if there is enough data. Convolution layers are crucial in producing the best results possible with the minimal data that is present. The training of this ablation architecture uses the same hyperparameter values as before.

5.4.3 Analysis of Failure Cases

Figure 5.2 displays some instances of the model failing. The characters that were predicted based on the input image demonstrate that prediction errors occurred when either the input image was distorted (Figure 5.2(b)) or a character's writing was nearly identical to another character (Figure 5.2). (a). These errors could be decreased by pre-training the transformer decoder on an Urdu language modeling task, which would enable the architecture to still anticipate characters in input images that are ambiguous based on the probability of the next character given the preceding sequence of characters.

[H]



Figure 5.2: The image displays instances of some of the input together with its ground-truth and predicted text presented below. Examples of label noise in the dataset are shown in (a). Given a picture, the given ground truth has extra, unneeded words or characters. present in the supplied image, raising this example's CER. The model accurately foretells the characters in the input image. (b) includes an an illustration of a distorted input image The model cannot foresee the actual label since the input images only contain a few difficult-to-read letters or literals because of the author's writing style. The input image for (c) has a calligraphic writing style that is intricate and challenging for the model to correctly predict. There is a sample of label noise in the dataset in (d). The provided ground truth for a given image is unreliable, but the predicted result is accurate given the input image. This demonstrates that the model is effective enough to yield reliable outcomes. The mismatched ground truth is the cause of the high CER. The situations where the model accurately detected the handwritten text and produced an extremely low CER are contained in the table (e), (f)

Conclusion and Future Work

6.1 Conclusion

Urdu handwriting recognition is treated as a Seq2Seq modeling problem, as inspired by [50] and presented a Conv-Transformer architecture that avoided the necessity of a separate language model. Additionally, the convolution layers at the beginning of a full transformer work to lower the spatial resolution of the images of handwritten Urdu text and extract crucial features. Reduced training and inference running times are achieved by compensating for the n^2 computational complexity of the Multi head Attention layers of the transformers with feature maps that have lower spatial resolution than the input image.

To the best of my knowledge, this thesis is the first to suggest a deep learning architecture that trains concurrently on datasets of printed and handwritten Urdu text, producing state-of-the-art results for unconstrained Urdu handwriting recognition.

6.2 Future Directions

Future steps include pre-training the architecture on a sizable dataset before fine-tuning it to a downstream task. An ImageNet classification task can be used to pre-train the Conv-Transformer encoder, and a language-specific language modeling task can be used to pre-train the transformer decoder. Due to the convolution and transformer architectures' strong generalization properties, this pre-training would significantly increase the accuracy on task-specific datasets following fine-tuning on them.

Bibliography

- [1] Hermann Ney, Ute Essen, and Reinhard Kneser. “On structuring probabilistic dependences in stochastic language modelling”. In: *Computer Speech & Language* 8.1 (1994), pp. 1–38.
- [2] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: [10.1162/neco.1997.9.8.1735](https://doi.org/10.1162/neco.1997.9.8.1735). eprint: <https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf>. URL: <https://doi.org/10.1162/neco.1997.9.8.1735>.
- [3] Stanley F Chen and Joshua Goodman. “An empirical study of smoothing techniques for language modeling”. In: *Computer Speech & Language* 13.4 (1999), pp. 359–394.
- [4] Herbert Jaeger. “Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the " echo state network" approach”. In: (2002).
- [5] U-V Marti and Horst Bunke. “The IAM-database: an English sentence database for offline handwriting recognition”. In: *International Journal on Document Analysis and Recognition* 5.1 (2002), pp. 39–46.
- [6] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. “Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks”. In: *Proceedings of the 23rd international conference on Machine learning*. 2006, pp. 369–376.
- [7] Mohamed Cheriet, Nawwaf Kharma, Ching Suen, and Cheng-Lin Liu. *Character recognition systems: a guide for students and practitioners*. John Wiley & Sons, 2007.

- [8] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. “Self-Taught Learning: Transfer Learning from Unlabeled Data”. In: *Proceedings of the 24th International Conference on Machine Learning*. ICML '07. Corvallis, Oregon, USA: Association for Computing Machinery, 2007, pp. 759–766. ISBN: 9781595937933. DOI: [10.1145/1273496.1273592](https://doi.org/10.1145/1273496.1273592). URL: <https://doi.org/10.1145/1273496.1273592>.
- [9] Zaher Al Aghbari and Salama Brook. “HAH manuscripts: A holistic paradigm for classifying and retrieving historical Arabic handwritten documents”. In: *Expert Systems with Applications* 36.8 (2009), pp. 10942–10951.
- [10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. “ImageNet: A large-scale hierarchical image database”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: [10.1109/CVPR.2009.5206848](https://doi.org/10.1109/CVPR.2009.5206848).
- [11] James H Martin. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Pearson/Prentice Hall, 2009.
- [12] Vinod Nair and Geoffrey E Hinton. “Rectified linear units improve restricted boltzmann machines”. In: *Icml*. 2010.
- [13] Remya Soman and Jency Thomas. “A Novel Approach for Mixed Noise Removal using ROR Statistics Combined WITH ACWMF and DPVM”. In: *International Journal of Computer Applications* 86.17 (2014).
- [14] K. O’Shea and R. Nash. “An introduction to convolutional neural networks”. In: *arXiv preprint arXiv:1511.08458* (2015).
- [15] Xiang Zhang, Junbo Zhao, and Yann LeCun. “Character-level convolutional networks for text classification”. In: *Advances in neural information processing systems* 28 (2015).
- [16] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. “Pointer sentinel mixture models”. In: *arXiv preprint arXiv:1609.07843* (2016).
- [17] S. Naz, A. I. Umar, R. Ahmed, M. I. Razzak, S. F. Rashid, and F. Shafait. “Urdu Nasta’liq text recognition using implicit segmentation based on multi-dimensional long short term memory neural networks”. In: *SpringerPlus* 5 (2016).

- [18] Joan Andreu Sanchez, Veronica Romero, Alejandro H Toselli, and Enrique Vidal. “ICFHR2016 competition on handwritten text recognition on the READ dataset”. In: *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE. 2016, pp. 630–635.
- [19] Joan Andreu Sánchez, Verónica Romero, Alejandro H Toselli, and Enrique Vidal. “Read dataset bozen”. In: *Zenodo* (2016).
- [20] Théodore Bluche and Ronaldo Messina. “Gated convolutional recurrent neural networks for multilingual handwriting recognition”. In: *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*. Vol. 1. IEEE. 2017, pp. 646–651.
- [21] S. Naz, A. I. Umar, R. Ahmed, I. Siddiqi, S. Ahmed, M. I. Razzak, and F. Shafait. “Urdu Nastaliq recognition using convolutional-recursive deep learning”. In: *Neurocomputing* 243 (2017), pp. 80–87.
- [22] Joan Puigcerver. “Are multidimensional recurrent layers really necessary for handwritten text recognition?” In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 1. IEEE. 2017, pp. 67–72.
- [23] Curtis Wigington, Seth Stewart, Brian Davis, Bill Barrett, Brian Price, and Scott Cohen. “Data augmentation for recognition of handwritten words and lines using a CNN-LSTM network”. In: *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*. Vol. 1. IEEE. 2017, pp. 639–645.
- [24] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xi-aohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. “An image is worth 16x16 words: Transformers for image recognition at scale”. In: *arXiv preprint arXiv:2010.11929* (2020).
- [25] Hangbo Bao, Li Dong, and Furu Wei. “Beit: Bert pre-training of image transformers”. In: *arXiv preprint arXiv:2106.08254* (2021).
- [26] Houcine Boubaker, Abdelkarim Elbaati, Najiba Tagougui, Haikal El Abed, Monji Kherallah, Volker Märgner, and Adel M. Alimi. *ADAB database*. 2021. DOI: [10.21227/wpf8-dk19](https://doi.org/10.21227/wpf8-dk19). URL: <https://dx.doi.org/10.21227/wpf8-dk19>.
- [27] Daniel Hernandez Diaz, Siyang Qin, Reeve Ingle, Yasuhisa Fujii, and Alessandro Bissacco. “Rethinking text line recognition models”. In: *arXiv preprint arXiv:2104.07787* (2021).

- [28] Tayyab Nasir, Muhammad Malik, and Khurram Shahzad. “MMU-OCR-21: Towards End-to-End Urdu Text Recognition Using Deep Learning”. In: *IEEE Access* PP (Sept. 2021), pp. 1–1. DOI: [10.1109/ACCESS.2021.3110787](https://doi.org/10.1109/ACCESS.2021.3110787).
- [29] Sumeet S Singh and Sergey Karayev. “Full page handwriting recognition via image to sequence extraction”. In: *International Conference on Document Analysis and Recognition*. Springer. 2021, pp. 55–69.
- [30] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jegou. “Training data-efficient image transformers and distillation through attention”. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, pp. 10347–10357. URL: <https://proceedings.mlr.press/v139/touvron21a.html>.
- [31] Killian Barrere, Yann Soullard, Aurélie Lemaitre, and Bertrand Coüasnon. “A Light Transformer-Based Architecture for Handwritten Text Recognition”. In: *International Workshop on Document Analysis Systems*. Springer. 2022, pp. 275–290.
- [32] Lei Kang, Pau Riba, Marçal Rusiñol, Alicia Fornés, and Mauricio Villegas. “Pay attention to what you read: non-recurrent handwritten text-line recognition”. In: *Pattern Recognition* 129 (2022), p. 108766.
- [33] George Retsinas, Giorgos Sfikas, Basilis Gatos, and Christophoros Nikou. “Best Practices for a Handwritten Text Recognition System”. In: *International Workshop on Document Analysis Systems*. Springer. 2022, pp. 247–259.
- [34] Christoph Wick, Jochen Zöllner, and Tobias Grüning. “Rescoring sequence-to-sequence models for text line recognition with CTC-prefixes”. In: *International Workshop on Document Analysis Systems*. Springer. 2022, pp. 260–274.
- [35] S. Ahmed, S. Naz, Salahuddin, M. Razzak, and A. Umar. “UCOM Offline Dataset – a Urdu Handwritten Dataset Generation”. In: *International Arab Journal of Information Technology*, 14(2) · March 2016 (Mar. (2016)).
- [36] K. Asad, M. Z. Asghar, S. Anam, I. A. Hameed, S. H. Asif, and A. Shakeel. “A survey on sentiment analysis in Urdu: A resource-poor language”. In: *Egyptian Informatics Journal* 22.1 ((2021)), pp. 53–74. ISSN: 1110-8665. DOI: [https://](https://doi.org/10.1016/j.eij.2021.01.001)

- doi.org/10.1016/j.eij.2020.04.003. URL: <https://www.sciencedirect.com/science/article/pii/S1110866520301171>.
- [37] Tom B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. *Language Models are Few-Shot Learners*. (2020). DOI: [10.48550/ARXIV.2005.14165](https://doi.org/10.48550/ARXIV.2005.14165). URL: <https://arxiv.org/abs/2005.14165>.
- [38] A. Ul-Hasan, S. Ahmed, F. Rashid, F. Shafait, and Thomas M. Breuel. “Offline Printed Urdu Nastaleeq Script Recognition with Bidirectional LSTM Networks”. In: *2013 12th International Conference on Document Analysis and Recognition*. (2013), pp. 1061–1065. DOI: [10.1109/ICDAR.2013.212](https://doi.org/10.1109/ICDAR.2013.212).
- [39] S. Hassan, A. Irfan, A. Mirza, and I. Siddiqi. “Cursive Handwritten Text Recognition using Bi-Directional LSTMs: A Case Study on Urdu Handwriting”. In: *2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML)*. (2019), pp. 67–72. DOI: [10.1109/Deep-ML.2019.00021](https://doi.org/10.1109/Deep-ML.2019.00021).
- [40] M. Husnain, Malik M. Saad Missen, S. Mumtaz, Muhammad Z. Jhanidr, M. Coustaty, M. Muzzamil Luqman, J. Ogier, and G. Sang Choi. “Recognition of Urdu handwritten characters using convolutional neural network”. In: *Applied Sciences* 9.13 ((2019)), p. 2758.
- [41] K. Khan and I. Haider. “Online recognition of multi-stroke handwritten Urdu characters”. In: *2010 International Conference on Image Analysis and Signal Processing*. (2010), pp. 284–290. DOI: [10.1109/IASP.2010.5476113](https://doi.org/10.1109/IASP.2010.5476113).
- [42] M. Li, T. Lv, L. Cui, Y. Lu, D. Florencio, C. Zhang, Z. Li, and F. Wei. “Trocr: Transformer-based optical character recognition with pre-trained models”. In: *arXiv preprint arXiv:2109.10282* ((2021)).
- [43] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. (2019). DOI: [10.48550/ARXIV.1907.11692](https://doi.org/10.48550/ARXIV.1907.11692). URL: <https://arxiv.org/abs/1907.11692>.

- [44] J. Michael, R. Labahn, T. Gruning, and J. Zollner. “Evaluating sequence-to-sequence models for handwritten text recognition”. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE. (2019), pp. 1286–1293.
- [45] M. F. Naeem, N. Zia, A. Awan, A. Ul-Hasan, and F. Shafait. “Impact of Ligature Coverage on Training Practical Urdu OCR Systems”. In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 01. (2017), pp. 131–136. DOI: [10.1109/ICDAR.2017.30](https://doi.org/10.1109/ICDAR.2017.30).
- [46] A. Rehman, A. Ul-Hasan, and F. Shafait. “High Performance Urdu and Arabic Video Text Recognition Using Convolutional Recurrent Neural Networks”. In: *International Conference on Document Analysis and Recognition*. Springer. (2021), pp. 336–352.
- [47] N. Riaz, S. Latif, and R. Latif. “From Transformers to Reformers”. In: *2021 International Conference on Digital Futures and Transformative Technologies (ICoDT2)*. (2021), pp. 1–6. DOI: [10.1109/ICoDT252288.2021.9441516](https://doi.org/10.1109/ICoDT252288.2021.9441516).
- [48] Malik W. Sagheer, Chun L. He, N. Nobile, and C. Y. Suen. “Holistic Urdu Handwritten Word Recognition Using Support Vector Machine”. In: *2010 20th International Conference on Pattern Recognition*. (2010), pp. 1900–1903. DOI: [10.1109/ICPR.2010.468](https://doi.org/10.1109/ICPR.2010.468).
- [49] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. *Training data-efficient image transformers distillation through attention*. (2020). DOI: [10.48550/ARXIV.2012.12877](https://doi.org/10.48550/ARXIV.2012.12877). URL: <https://arxiv.org/abs/2012.12877>.
- [50] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin. “Attention is all you need”. In: *Advances in neural information processing systems* 30 ((2017)).
- [51] N. Zia, Muhammad F. Naeem, Syed K. Raza, Muhammad M. Khan, A. Ul-Hasan, and F. Shafait. “A convolutional recursive deep architecture for unconstrained Urdu handwriting recognition”. In: *Neural Computing and Applications* ((2021)), pp. 1–14.