Deep Learning for Handwriting Text Recognition: Existing Approaches and Challenges

Nikolay Teslya
SPC RAS
St. Petersburg, Russia
teslya@iias.spb.su

Samah Mohammed
ITMO University
St. Petersburg, Russia
samah369@mail.ru

Abstract—**In recent years, Handwritten Text Recognition (HTR) has attracted widespread attention due to its huge applications. HTR is the process of extracting handwritten text from an image and converting it into a digital form for machine operation. Nevertheless, due to the huge differences in personal writing and the various properties of handwritten characters in multiple languages, HTR is still a challenging open research problem, and robustness and adaptability require additional improvements. The existing approaches to solve the HTR problem are usually systems based on Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), which utilize the Connectionist Temporal Classification (CTC) objective function. However, many approaches based on attention sequence to sequence (Seq2Seq) have been proposed for the HTR task. The Seq2Seq based approaches are more flexible, suitable for the temporal nature of the text and can use different attention mechanisms to focus on the most relevant features of the input. In this paper, we provide extensive comparison of the current Deep Learning approaches for the task of HTR. Also, we outline the current problems that limits the effectiveness of these approaches.**

I. Introduction

Handwriting Text Recognition (HTR) is a subfield of Optical Character Recognition (OCR). Based on the input data, it is further subdivided into offline recognition and online text recognition [1]. Offline OCR is a static system in which the input data is presented as scanned images, while online OCR is applied to data that is being captured in present time or in real time, and is considered more complex and advanced because it solves overlap problems of input data that is not available in the offline system.

Technically speaking, the main phases of OCR are image acquisition, preprocessing, segmentation, feature extraction, classification and/or possible post-processing. Each stage has its own goal, and its efficiency determines the accuracy of the next stage, and ultimately determines the entire recognition process. HTR has been of interest to the Pattern Recognition community for many years. Converting images from handwritten text to machine readable format has a large number of uses such as making HTR an open research problem that is still challenging.

With the advent of neural networks and Deep Learning architectures, HTR, like many other applications, has dramatically improved performance. Actually, the architecture proposed by LeCun et al. [2] for HTR, was one of the first applications of Convolutional Neural Networks (CNNs). This architecture is proposed to recognize handwritten digits from the MNIST dataset.

In the last two decades, several approaches have been proposed for tackling the HTR task such as Hidden Markov Models (HMM) [3], [4], [5]. Recurrent Neural Networks (RNNs) and Connectionist Temporal Classification (CTC) [6], [7], [8], [9] and attention Seq2Seq approaches [10], [11], [12], [13].

Many surveys on HTR have been published [14], [15]. Although these surveys usually cover the basics, they are general and not focused on the HTR architectures [16], [17]. In addition, new research approaches have emerged. Therefore, it will be realistic and necessary to investigate the current state-of-the-art HTR approaches.

In this survey we focus on conducting a comprehensive and extensive comparison between the current Deep learning approaches for the task of HTR, for such comparison, the IAM dataset and the most usual evaluation metrics for HTR systems were adopted: Character Error Rate (CER) [18]. Also, the paper contributes in outlining the current problems that limit the effectiveness of these techniques.

The rest of this paper is structured as follows. In section II we first review the state-of-art models in Deep Learning, that currently used for the HTR problem: Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Transforms, Bidirectional Encoder Representation from Transformers (BERT), attention mechanisms, finally we briefly review the Connectionist Text Classification (CTC). In section III we introduce the most state-of art approaches for HTR: CTC_based approaches and attention Seq2Seq_based approaches, and we conduct a comparison between these approaches. In section IV we present the current challenges in HTR. Finally, the conclusion is presented.

II. Background

A. *Convolutional Neural Networks (CNNs):*

CNN [19] is a type of deep learning model for processing data that has a grid pattern, such as images. CNNs are designed for extracting features from low to high level patterns.

CNN is a mathematical construct that are ideally contains three types of layers or building blocks: convolution, pooling, and fully connected layers. The convolution layer is the main layer and consists of a stack of mathematical operations such as convolution, which is a special type of linear operation. The convolutional and pooling layers perform feature extraction, while the fully connected layer maps the extracted features to the final output, such as classification. The process of optimizing parameters such as the kernel is called training and is performed to minimize the difference between the output and the ground truth label through optimization algorithms such as backpropagation and gradient descent. Studies have shown that CNN performance has improved by increasing the width and depth of the network.

B. Recurrent Neural Networks (RNNs):

RNNs are very powerful in modeling data sequences, for example, Natural Language Processing (NLP) and time series. Their main advantages [20] that RNN can handle input of any length and the input size does not affect the model size.

Although one of the attractions of RNNs is the possibility of connecting previous information to the current task, there have been many cases, where the gap between the relevant information and the place that it's needed is big, RNNs can't learn to use the past information. In such cases, RNNs can be difficult to train because gradients passed back through many layers may vanish or explode. To address this problem, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) are two modules designed as RNN building blocks to deal with vanishing gradient and better learn long-term dependencies.

C. Transformers:

Transformer is the first transduction model that completely relies on self-attention to calculate its input and output representations without using sequence alignment RNN or convolution [21].

The main components of the transformer are a set of encoders and decoders. Encoders encode the input into a representation, which is then decoded by decoders. One encoder block has a self-attention mechanism and a feedforward neural network. On the other hand, a decoder has both encoder components and an encoder - decoder attention. Encoders and decoders are stacks of the same structures, that is, there will be a pair of encoders stacked one on top of the other. Several decoders can be combined in the same way. The word embeddings of the input sequence will be passed to the first encoder. They are then transformed and sent to the next encoder, and so on. The output of the last encoder will be the input of the first decoder, then this decoder applies some transforms and will be sent to the next decoder.

D. Bidirectional Encoder Representation Transformer (BERT):

BERT is supported by Transformer. Its core principle is attention. The attention function can be described as mapping a query and a set of value-key pairs to an output, where the query, keys, values, and outputs are all vectors. The output is calculated as a weighted sum of the values, where the weight assigned to each value is calculated by the query matching function with the corresponding key. It understands the contextual relationship between different words [22]. BERT does not decode the encoded information, but only encodes and generates a language model, so one encoder is sufficient. Compared with directional models such as RNN and LSTM, they conceive each input sequentially (from left to right or from right to left). In fact, Transformer and BERT are non-directional—to be precise, because both models take the entire sentence as input instead of reading it sequentially. This feature allows the model to learn the context of a word relative to all other words in the sentence.

BERT uses two training mechanisms, Mask Language Modeling (MLM) and Next Sentence Prediction (NSP) to overcome dependency challenges. For MLM, 15% of the words in the sequence are masked with a [MASK] token before the input vector is fed into the encoder. In addition, all words are replaced by their vector representation. The goal of MLM is to predict the masked word in relation to all other words in the sentence. For NSP, during training, the model receives a pair of sentences as input. Of the entire input pair, 50% have exactly consecutive sentences as the second term, and the rest have random sentences as the second sentence. The model eventually learns it and predicts if the second sentence in the pair is actually a contiguous sentence.

E. Connectionist Text Classification (CTC):

Connectionist Temporal Classification (CTC) is an excellent solution that can avoid pre-segmentation of training examples and post-processing of converting the output of recurrent neural networks into label sequences [23]. The last layer of the network (SoftMax layer) contains a unit for each label, and outputs the probability of having the corresponding label at a specific time step. When using the CTC method, an additional unit is defined to simulate the probability of having a blank label (that is, no category label).

A representation β is a function that maps the probability sequence π from the network output, that is, the probability sequence of observing a specific label in a given time range, to the prediction sequence l, that is, one of the input observation sequences of the label sequence whose length is less than or equal to. β includes removing duplicate labels, and then blank predictions. The conditional probability of the label sequence l for a given observation sequence x is the sum of the probabilities of the path π, where β(π) = l (Eq. 1):

$$p(1/x) = \sum_{\pi:\beta(\pi)=1} p(\pi/x) \qquad (1)$$

III. HTR APPROACHES

In the literature, researchers have made considerable efforts to employ the most state-of-art computer vision techniques to HTR systems. The Hidden Markov Models (HMM) [3], [4], [5], [6] is one of the most popular approaches. However, HMM failed to make use of the context information, particularly with long text sequences because of the Markovian assumption where each observation depends only on the current state.

A. CTC_based approaches for HTR:

In the past few years, deep learning methods have significantly improved HTR tasks over traditional methods. There are a lot of studies using deep learning methods to solve this problem.

Shi et al. [24] proposed a Convolutional Recurrent Neural Network (CRNN)model, The network architecture (Fig. 1) contains three main layers:

- Convolutional layers.
- Recurrent layers.
- Transcription layer.

Convolutional layers based on the VGG-Very Deep architecture [25] to extract the feature sequence from the input image, particularly 7 CNN layers and batch normalization layers are inserted after the fifth and sixth convolutional layers respectively to accelerate the training process. The Recurrent layers are represented by two deep Bidirectional Long Term Short Memory (BLSTM)layers with 256 units built on the top of the convolutional layers, to predict a label distribution for each frame. Finally, the transcription layer: to translates the predictions made by BLSTM into the final label sequence.

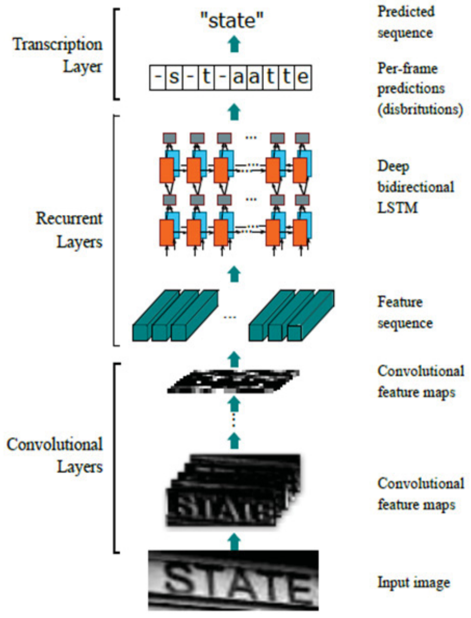

Fig. 1. CRNN architecture [25]

From the network training, we seek to reduce the probability of the negative log-likelihood of the conditional probability of the ground truth Eq. 2):

$$\vartheta = - \sum_{I_i, \mathrm{l}_i \in \mathrm{\gamma}} \log p(\mathrm{l}_i / y_i) \qquad (2)$$

where I_i, l_i are the training image and ground truth label sequence respectively from the training data $\mathrm{\gamma} = \{I_i, \mathrm{l}_i\}$.

Though CRNN is composed of different kinds of network architectures, it can be trained with one loss function and for that the Connectionist Temporal Classification (CTC) function [24] was adopted. However, CRNN was only applied to identify isolated words. Also, the model size is about 6.8 million parameters which makes it difficult to implement in many real-world applications.

A similar architecture (CNN-BLSTM), introduced by Puigcerver [6] as shown in Fig.2, has 5 convolution layers and 5 BLSTM. The number of hidden layers for each is 256. To avoid overfitting a dropout (with probability 0.5) is applied before the linear layer where each column after the BLSTM layers must be mapped to an output label. Puigcerver aimed from this architecture to investigate whether MDLSTM networks [26] are strictly required to achieve state-of-the-art performance for line-level HTR by comparing the proposed (CNN-BLSTM) against 2D-LSTM architectures for two widely used datasets (IAM & Rimes), he observed that the running times of the 1D-LSTM architecture (the proposed architecture) much smaller than the 2D-LSTM (6–7 speedups). Although, (CNN-BLSTM) has a significantly lower WER in both datasets with 9.6 million parameters in total.

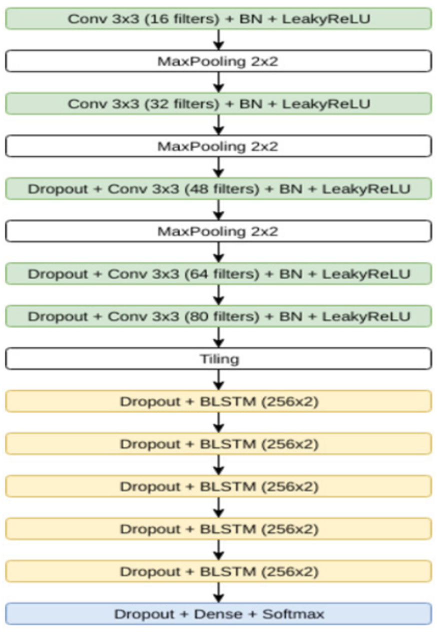
Conv 3x3 (16 filters) + BN + LeakyReLU
MaxPooling 2x2
Conv 3x3 (32 filters) + BN + LeakyReLU
MaxPooling 2x2
Dropout + Conv 3x3 (48 filters) + BN + LeakyReLU
MaxPooling 2x2
Dropout + Conv 3x3 (64 filters) + BN + LeakyReLU
Dropout + Conv 3x3 (80 filters) + BN + LeakyReLU
Tiling
Dropout + BLSTM (256x2)
Dropout + BLSTM (256x2)
Dropout + BLSTM (256x2)
Dropout + BLSTM (256x2)
Dropout + BLSTM (256x2)
Dropout + Dense + Softmax

Fig. 2. CNN-BLSTM architecture [6]

Bluche et al. [7] proposed an elaborated gating system (GCRNN) presented by adding convolution gates to the convolution layers, the gate is implemented as a convolution filter, followed by a sigmoid activation, the system learns in which context a computed feature is relevant. Fig. 3 shows in detail the distribution of parameters and hyperparameters through 8 convolutional layers (3 gated included) and 2 BLSTM.

The output of the gating mechanism is the pointwise multiplication of the input with the output of the gate:

$$y = g(x).x \qquad (3)$$

where

$$\begin{aligned} g(x_{ij}) = \sigma\big(&w_{00}x_{i-1,j-1} + w_{01}x_{i-1,j} + w_{02}x_{i-1,j+1} \\ &+ w_{10}x_{i,j-1} + w_{11}x_{i,j} + w_{12}x_{i,j+1} \\ &+ w_{20}x_{i+1,j-1} + w_{21}x_{i+1,j} \\ &+ w_{220}x_{i+,j+1}\big) \end{aligned} \qquad (4)$$

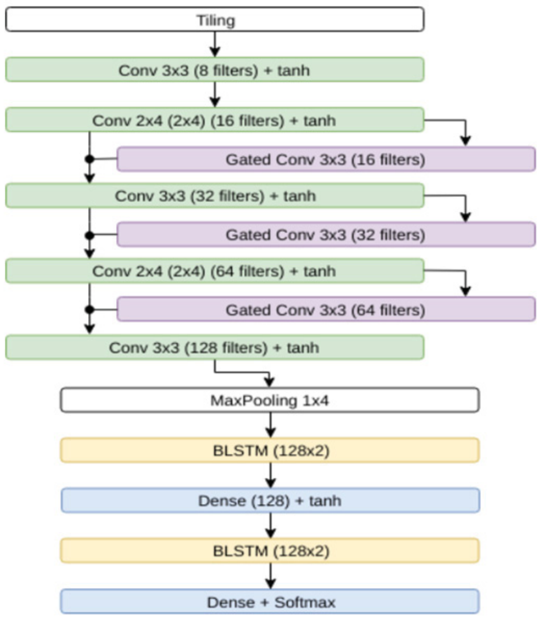
Tiling
Conv 3x3 (8 filters) + tanh
Conv 2x4 (2x4) (16 filters) + tanh
Gated Conv 3x3 (16 filters)
Conv 3x3 (32 filters) + tanh
Gated Conv 3x3 (32 filters)
Conv 2x4 (2x4) (64 filters) + tanh
Gated Conv 3x3 (64 filters)
Conv 3x3 (128 filters) + tanh
MaxPooling 1x4
BLSTM (128x2)
Dense (128) + tanh
BLSTM (128x2)
Dense + Softmax

Fig. 3. GCRNN architecture [7]

To train the previous model a dataset from various sources have been used. Particularly 133k text images from seven languages (English, French, Spanish, German, Portuguese, Italian and Russian), including 30% of private data (mainly Russian). First, the model is trained for all the data to obtain generic Latin-script neural networks then the decoder part (BLSTM)is subsequently adapted to each language. GCRNN architecture contains about 750k parameters compared to Puigcerver (CNN-BLSTM) architecture which contains about 9.6 million parameters and thus GCRNN exchange high performance for simplicity of the model.

Inspired by (CNN-BGRU) and GCRNN, Flor proposed (Gated-CNN-BGRU) architecture [8], aiming at: (*i*) to achieve results compatible with Puigcerver model (CNN-BLSTM); and (*ii*) to keep the small number of parameters (about 0.8 million parameters), such as Bluche model (Gated-CNN-BLSTM). Fig. 4 shows the workflow of Flor architecture through 11 convolutional layers (5 of them are gated) and 2 Bidirectional Gated Recurrent Units (BGRU) instead of BLSTM.

The model was trained in five main public datasets (IAM, Rimes, Bentham [27], Saint Gall [28] and Washington [29]), and then the results compared with the results obtained from previous models (CNN-BLSTM and GCRNN) by analyzing the average error rates obtained in all the datasets, the HTR-Flor model reached an average CER of 3.85% with an average WER of 12.23%. Puigcerver 7.72% with 18.71% and Bluche 7.08% with 19.02%.

Thus, the model is able to achieve satisfied results even in smaller datasets and requires less computational and needs less computational resources, which can be applied for real world applications.

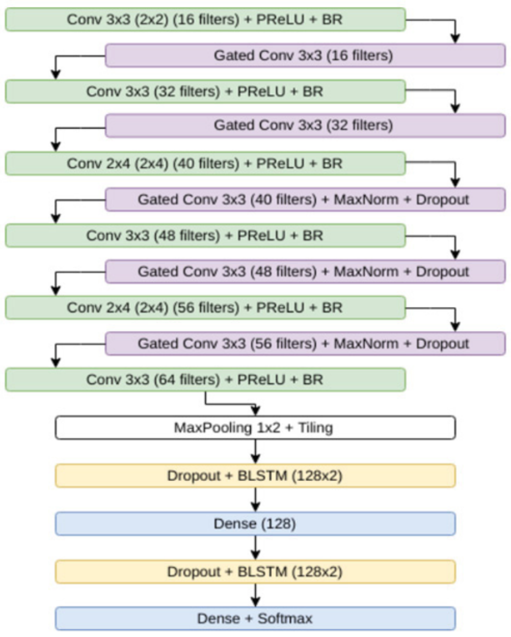
Conv 3x3 (2x2) (16 filters) + PReLU + BR
Gated Conv 3x3 (16 filters)
Conv 3x3 (32 filters) + PReLU + BR
Gated Conv 3x3 (32 filters)
Conv 2x4 (2x4) (40 filters) + PReLU + BR
Gated Conv 3x3 (40 filters) + MaxNorm + Dropout
Conv 3x3 (48 filters) + PReLU + BR
Gated Conv 3x3 (48 filters) + MaxNorm + Dropout
Conv 2x4 (2x4) (56 filters) + PReLU + BR
Gated Conv 3x3 (56 filters) + MaxNorm + Dropout
Conv 3x3 (64 filters) + PReLU + BR
MaxPooling 1x2 + Tiling
Dropout + BLSTM (128x2)
Dense (128)
Dropout + BLSTM (128x2)
Dense + Softmax

Fig. 4. Gated-CNN-BLSTM architecture [8]

Tsochatzidis et al. [9] proposed a new architecture OctCNN-BGRU (Fig. 5) inspired by HTR-Flor architecture and Octave-convolution operation [30], this includes processing the input in two different scales to capture both high- and low-frequency patterns. Thus, the input feature map X is factorized into two portions along the channel axis, so that $X = \{X^H, X^L\}$, resulting in two feature maps that capture fine and low-detailed information. Thus, a new convolution operator is used to operate on this representation resulting in the layer output$Y = \{Y^H, Y^L\}$, as defined in the next equations:

$$Y^H = f(X^H, W^{H\to H}) + unsample\ (f(X^L, W^{L\to H}), 2) \tag{5}$$

$$Y^L = f(X^L, W^{L\to L}) + f\ \left(pool(X^H, 2), W^{H\to L}\right) \tag{6}$$

where $f(X, W)$ represents the convolution of X and the kernel W followed by the activation function, $pool(X, k)$ represents the average pooling of kernel size k, and $unsample(X, k)$ represents upsampling by factor k . The channels of each OctConv layer are divided into high-frequency and low-frequency features, which are configured by hyperparameter a, which affects the number of convolution kernels in each frequency band.

For the experimental evaluation, four Greek historical handwritten documents have been used with the two public datasets (IAM, Rimes). It turned out that the OctCNN-BGRU outperformed the CNN-BLSTM on the Greek documents, on average the difference between the two methods equals to 3.5% and 6.4% for CER and WER, respectively. For the IAM and Rimes datasets the CNN-BLSTM outperformed the OctCNN-BGRU, the difference equals to 11.2% for CER and 3.7% for WER.

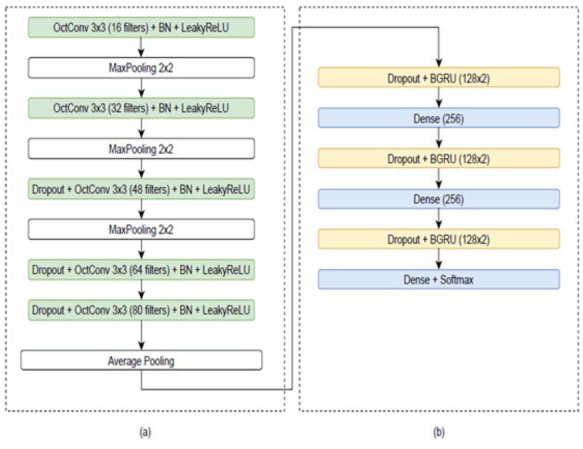
OctConv 3x3 (16 filters) + BN + LeakyReLU
MaxPooling 2x2
OctConv 3x3 (32 filters) + BN + LeakyReLU
MaxPooling 2x2
Dropout + OctConv 3x3 (48 filters) + BN + LeakyReLU
MaxPooling 2x2
Dropout + OctConv 3x3 (64 filters) + BN + LeakyReLU
Dropout + OctConv 3x3 (80 filters) + BN + LeakyReLU
Average Pooling
Dropout + BGRU (128x2)
Dense (256)
Dropout + BGRU (128x2)
Dense (256)
Dropout + BGRU (128x2)
Dense + Softmax
(a)
(b)

Fig. 5. OctCNN-BGRU architecture [9]

B. Attention Seq2Seq_based approaches for HTR:

Attention enhances the ability of the network to extract the most relevant information for each part of the output sequence. In addition, attention networks can not only map inputs to the correct outputs, but also model the language structure within the output sequence.

Attention Seq2Seq_based models have been first used by Valle et al. [31] for the HTR problem, where he used the model proposed by Deng et al. [32] which stacks a multilayer encoder and attention-based decoder on a multilayer convolutional neural network (CNN). CNN contains seven-layer CNN alternating with max-pooling layers. Each convolutional layer uses batch normalization and a ReLU activation function with weights initialized using Xavier initialization. Dropout (p = 0.5) is applied to the inputs after the last convolutional layer.

The CNN extracts image features from the raw input and arranges the features on a grid. Stacked on the CNN is a single-layer bidirectional LSTM encoder (256 hidden units) and a two-layer Gated Recurrent Unit (GRU) decoder (128 hidden units). The encoder reencodes each row of grid, generating a re-encoded feature grid. Regarding to the attention, three attention mechanism are used for the decoder (softmax, sigmoid and no attention).

The closest model to Valle model is the model proposed by Bluche [33], which is a MDLSTM encoder and a softmax attention-enhanced bidirectional LSTM decoder (MDLSTM + Attention). The main difference is that the model does not require a CNN to extract visual features because the encoder is capable of inputting images. Although, the features from the encoder step still needed to pre-trained using CTC loss.

Kang et al [10] proposed an attention-based seq2seq model as shown in Fig. 6, which contains three main parts: i) an encoder: VGG16-BN have been used and initialized with the pre-trained weights from ImageNet then a multi-layered Bi-directional Gated Recurrent Unit (BGRU) which will involve mutual information and extra positional information for each column is added. ii) attention mechanism: location-based attention [24] is applied to calculate the most relevant context vector c_t where:

$$c_t = g(\alpha_t, H) = \sum_{i-0}^{N-1} \alpha_{ti} h_i \tag{7}$$

The attention mask vector at time step t is α_t, h_i is the hidden state of the encoder at the current time step $i \in \{0,1,..,N-1\}$, s_i is the hidden state of the decoder at the current time step $i \in \{0,1,..,T-1\}$, where T is the maximum length of decoding characters. Then,

$$\alpha_i = softmax(e_t) \tag{8}$$

$$e_{t,i} = f(h_i, s_{i-1}) = w^T \tanh(Wh_i + Vs_{t-1} + b) \tag{9}$$

where w, W, V and b are trainable parameters.

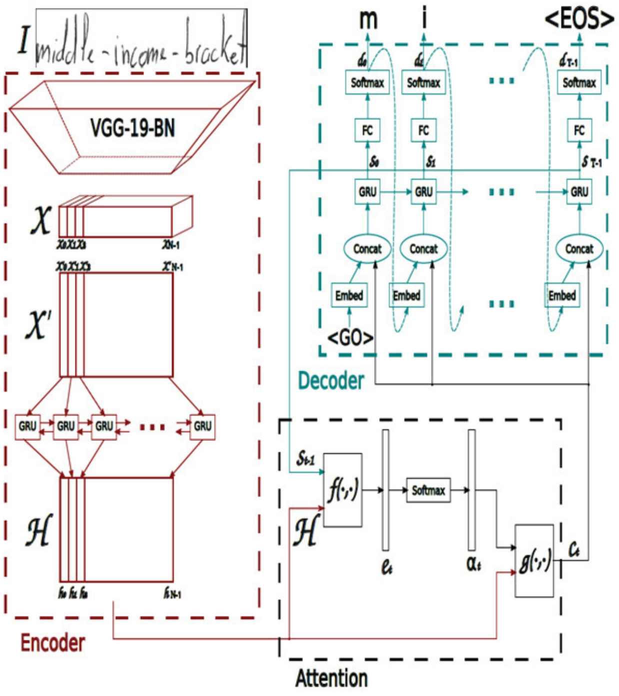

Fig. 6. Kang architecture (attention Seq2Seq_based model) [10]

iii) the decoder: represented by one-directional multi-layered GRUs, its input a concatenation of the embedding vector of the previous time step and the context vector $c_t$and the output s_t ,is the hidden state of decoder at current time step.

Michael et al. [13] proposed a Seq2Seq model. The encoder is a deep CNN layer followed by three BLSTM layers and the decoder is a single unidirectional LSTM with 256 hidden units and a dropout probability of 50% at train time. The encoder extracts low-level features from the written text line and sequentially encodes temporal context between them. The decoder outputs a character sequence one step at a time, using an attention mechanism to focus on the most relevant encoded features at each decoding step. Many experimental comparisons between various attention mechanisms and positional encodings are conducted, particularly, six different attention mechanisms are used (content based, penalized and location-based attention, monotonic and chunk wise attention, as well as hybrid attention).

Diaz et al. [12] studied the general problem of developing a universal architecture that can extract text from any image. They compare 9 different model design options on an internal dataset and on a universal Text-Line Recognition (TLR) task, pairing each of three encoders (Self-Attention, Gated Recurrent Convolutional Layer (GRCL), and BLSTM) to each of three decoders (CTC, CTC with LM and the Transformer decoder).

They found that the model that uses a Self-Attention encoder coupled with the Connectionist Temporal Classification (CTC) decoder, compounded with an explicit language model, outperforms all other models having both maximal text-line recognition accuracy and minimal complexity. Also, they suggest the use of image chunking to ensure that the model works efficiently and effectively on arbitrary long input images without shrinking.

Li et al. [11] proposed an end-to-end text recognition approach with pre-trained image transformer and text transformer model, namely TrOCR (Fig. 7).

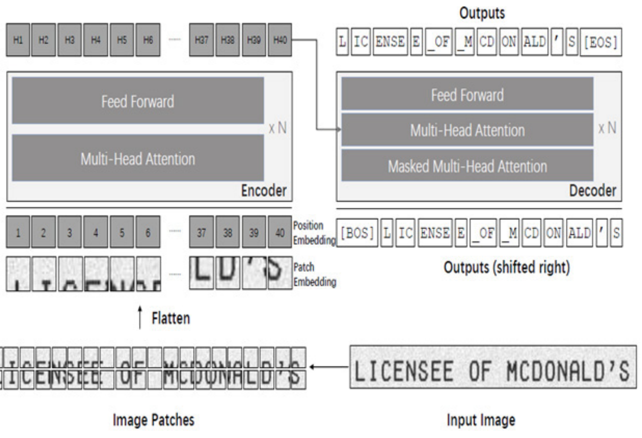

Fig .7. TrOCR architecture [11]

The vanilla transformer encoder-decoder is used in TrOCR, where the encoder is designed to take image patches, first resizing each image to a fixed size *(H, W).* The raw image must be a set of input tokens that the Transformer encoder can process. Here, the encoder decomposes the input image into batches of fixed size $N = HW/P^2$square patches *(P, P).* The width W and height H of the resized image are guaranteed to be divisible by the patch size P. The patch is then flattened into a vector and projected linearly onto the patch-embedded D-dimensional vector. D is the hidden size of Transformers through all its layers. The input sequence goes through a stack of identical encoder layers. Each Transformer layer has a feedforward network that is fully connected to the multi-head self-attention module. Both of these two parts are followed by residual connections and layer normalization.

For the self-attention modules, all of the queries, keys and values come from the same sequence. The matrix of the attention output is computed as:

$$Attention(K, Q, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{10}$$

where K, Q, V are the keys, queries, values matrices, respectively and d_k $is\ the\ dimension\ of\ keys\ matrix.$

Similar to the encoder, the decoder has the same structure of layers, except there is an encoder-decoder attention layer between the multi-head self-attention layer and feed-forward layer, the keys and the values come from the encoder output and the queries come from the decoder input.

Both the encoder and the decoder are initialized by the public models pre-trained on large-scale labeled and unlabeled datasets. To initialize the encoder, the DeiT by Touvron et al. [34] and BEiT by Bao et al. [35] models are used. While the RoBERTa modules [36] are used to initialize the decoder, these modules measure the effect of hyperparameters and training data size, they delete the next sentence's prediction objective and change the masking pattern of the masked language module. Many different combinations of the encoder and the decoder are compared to find the best settings. The metrics used for evaluation are: word level precision, F1 score and recall. In results of combined models, the best performance is obtained using the BEiT encoders and the RoBERTa$_{LARGE}$ decoders (558M parameters in total).

Table I summarizes the main approaches for Handwritten Text Recognition. For the comparison the Character Error rate is used. Also, the main features and disadvantages are presented for each approach.

IV. Main challenges in HTR

This section presents the main challenges in HTR, which mainly came from the inherent variability of handwriting text, the huge different writing styles and the number of different languages and scripts.

- CTC only allows monotonic alignments, which may be a valid assumption for word-level or line-level HTR tasks, but it lacks the possibility for further research on paragraph or even more complex article styles. Also, it is challenging for detectors to separate words for scripts that do not separate words by spaces such as Chinese, Japanese, and Korean and they are more likely to miss punctuation and diacritic marks.
- Considering issues related to long images. There are at least two new aspects that need to be considered, which are efficiency and performance.

  Long images affect the efficiency of models with the Self-Attention encoder due to quadratic scaling with image length. This problem can be solved for CTC models without performance loss by chunking the images.

  Training on images of fixed maximum width affects the performance on longer images of models that make use of the Transformer decoder. This issue can be solved by resizing the images to the train width.
- Although encoder/decoder method achieved great success in the field of HTR, there is still a lot of room to improve with pre-trained CV and NLP models: i) the network parameters in existing methods are trained from scratch with synthetic/human-labeled datasets, leaving large-scale pre-trained models unexplored; ii) as image Transformers

TABLE I. MAIN APPROACHES FOR HANDWRITTEN RECOGNITION SYSTEM

Model	Architecture	IAM CER	Main features	Main drawbacks
CNN+BLSTM/CTC Shi et al. [24]	7 CNN layers + 2BLSTM layers/CTC	3.5 %	-The model handles sequences in arbitrary lengths, involving no character segmentations. -It is not confined to any pre-defined lexicon	-Difficulties in remembering long contexts due to vanishing / exploding gradient problems. -A model with millions of trainable parameters (9.4 million) makes it a challenge to be implemented in many real-world applications.
CNN+BLSTM/CTC Puigcerver [6]	5CNN layers + 5 BLSTM layers/CTC.	7.72%	Using random distortions during training as synthetic data augmentation significantly enhances the accuracy of the model	
GCRNN+BLSTM /CTC Bluche et al. [8]	8 convolutional layers (3 gated included) and 2 BLSTM layers /CTC	7.08 %	-Convolution gate enables hierarchical context-sensitive feature extraction. -Fast computing on GPU	A model with few parameters (0.7Million parameters), exchange high performance for simplicity of the model.
GCRNN +BGRU/CTC Flor et al. [8]	11 convolutional layers (5 of them are gated) + 2 Bidirectional Gated Recurrent Units (BGRU)	3.85%	-Improve recognition results from the CNN-BLSTM approach through the new Gated-CNN-BGRU architecture. - Reduce the number of trainable parameters (thousands)through the Gated-CNN-BGRU architecture, making the model smaller and with lower computational cost instead of the traditional CNN-BLSTM (millions).	Flor approach doesn't stand out in number of trainable parameters (0.8M) and decoding time (55ms/line) when comparing to Bluche [8] which has (0.7M) parameters and decoding time (32ms/line).
OctCNN-BGRU /CTC Tsochatzidis[9]	5 Octave-convolution layers +2 Bidirectional Gated Recurrent Units (BGRU) /CTC	7.30%	Octave convolution includes processing the input in 2 different scales to capture both low and high frequency patterns	Text line detection and segmentation in the document image is not addressed
CNN + LSTM-GRU /Attention (SoftMax) Valle et al. [31]	7CNN layers, stacked on single-layer LSTM encoder and two-layers GRU decoder.	16.58%	-Softmax attention focuses heavily on individual characters when predicting characters. -Attention networks are trained without the aid of a lexicon or explicit language model.	-Doesn't approach the current state -of-the-art CTC_based models due to the complexity in the architecture. -The most comparable model by Michael et al. [13] outperforms it.
MDLSTM + Attention Bluche et al. [33]	a MDLSTM encoder and a softmax attention-enhanced followed by bidirectional LSTM decoder	12.60%	-Does not require a CNN to extract visual features because the encoder is capable of inputting images and that make it easy to implement and maintain	Although MDLSTM extends the capability of the RNNs architecture to multidimensional data, Its complex architecture requires high computational cost.
VGG16-BGRU+GRU/Attention Kang et al. [10]	The encoder is a VGG16-BN followed by BGRU, the location-based attention is applied and the decoder is single-layer GRU	6.88%	The model doesn't need any pre-processing step, predefined lexicon, language model nor CTC loss	Even though no language model is used, the decoder might learn the relations between characters in the training vocabulary.
CNN-BLSTM+LSTM/Attention Michael et al. [13]	The encoder is a deep CNN layer followed by three BLSTM layers and the decoder is a single LSTM with various attention mechanisms	4.87%	-The model can be trained end-to-end and the optional integration of a hybrid loss allows the encoder to retain an interpretable and usable output. -No language Module is needed	Slow computations due to the complexity of the architecture where RNNs are used in the encoder and decoder.
S-Attn+CTC+LM Diaz et al. [12]	Self-attention encoder coupled with the CTC decoder	3.15%	-The model gets the top accuracy with low memory requirements and very good latency	External language model is needed
TrOCR Li et al. [11]	The encoder is pre-trained image transformer and the decoder is a pre-trained text transformer	2.89%	-An end-to-end Transformer-based model with pre-trained CV and NLP models. -Doesn't use CNN as backbone doesn't rely on any pre/post-processing steps. -The model can be extended to multilingual model with minimum efforts	Total number of parameters of the base model 334 million, comparing to 558 million parameters for the large model

become more and more popular [37], [35] especially the recent self-supervised pre-training [37], it is worth to investigate whether pretrained image Transformers can replace CNN backbones, meanwhile exploiting the pre-trained image Transformers to work together with the pre-trained text Transformers in a single framework on the text recognition task. Li et al. [11] were the first to investigate with pre-trained CV and NLP models.

V. CONCLUSION

Since it involves such a broad range of applications, HTR is serviceable in our daily lives. Several researchers proposed many approaches in this field. Although, the researches could enhance the accuracy rate and control the time complex, two important factors that might slow down the widespread adoption of these techniques in the area are the need for large training data and subsequently advanced hardware and more complex architectures to deal with such high amounts of data and achieve the good improvements. The main contribution of this paper is providing a critical state-of-art review of the recent proposed approaches in the HTR field. Although, we identified and discussed the main challenges that still exist.

We hope that our survey will be beneficial for researchers in the HTR field.

References

[1] N. Islam, Z. Islam, and N. Noor, “A Survey on Optical Character Recognition System,” Journal of Information & Communication Technology (JICT), vol. 10, no. 2, p. 4, 2016.

[2] A. L. Bianne-Bernard, F. Menasri, R. Al-Hajj Mohamad, C. Mokbel, C. Kermorvant, and L. Likforman-Sulem, “Dynamic and contextual information in HMM modeling for handwritten word recognition,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 10, pp. 2066–2080, 2011, doi: 10.1109/TPAMI.2011.22.

[3] T. Bluche, H. Ney, and C. Kermorvant, “Tandem HMM with convolutional neural network for handwritten word recognition,” ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, pp. 2390–2394, Oct. 2013, doi: 10.1109/ICASSP.2013.6638083.

[4] E. Giménez Pastor et al., “Handwriting word recognition using windowed Bernoulli HMMs,” Pattern Recognition Letters, vol. 35, no. 1, pp. 149–156, 2014, doi: 10.1016/J.PATREC.2012.09.002.

[5] A. H. Toselli and E. Vidal, “Handwritten text recognition results on the Bentham collection with improved classical N-Gram-HMM methods,” ACM International Conference Proceeding Series, pp. 15–22, Aug. 2015, doi: 10.1145/2809544.2809551.

[6] J. Puigcerver, “Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition?”,.

[7] T. Bluche and R. Messina, “Gated Convolutional Recurrent Neural Networks for Multilingual Handwriting Recognition,” Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, vol. 1, pp. 646–651, Jul. 2017, doi: 10.1109/ICDAR.2017.111.

[8] A. Flor, S. Neto, B. Leite, D. Bezerra, A. H. Toselli, and E. B. Lima, “HTR-Flor: A Deep Learning System for Offline Handwritten Text Recognition”.

[9] L. Tsochatzidis, S. Symeonidis, A. Papazoglou, and I. Pratikakis, “HTR for Greek Historical Handwritten Documents,” Journal of imaging, vol. 7, no. 12, p. 260, Dec. 2021, doi: 10.3390/JIMAGING7120260.

[10] L. Kang, J. I. Toledo, P. Riba, M. Villegas, A. Fornés, and M. Rusiñol, “Convolve, Attend and Spell: An Attention-based Sequence-to-Sequence Model for Handwritten Word Recognition,” undefined, vol. 11269 LNCS, pp. 459–472, 2018, doi: 10.1007/978-3-030-12939-2_32.

[11] M. Li et al., “TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models,” Sep. 2021.

[12] D. Hernandez Diaz Google Research, S. Qin Google Research, R. Ingle Google Research, Y. Fujii Google Research, and A. Bissacco Google Research, “Rethinking Text Line Recognition Models”.

[13] J. Michael, R. Labahn, T. Grüning, and J. Zöllner, “Evaluating Sequence-to-Sequence Models for Handwritten Text Recognition”.

[14] T. Somashekar, “A Survey on Handwritten Character Recognition using Deep Learning Technique”.

[15] A. Baldominos, Y. Saez, and P. Isasi, “A Survey of Handwritten Character Recognition with MNIST and EMNIST”, doi: 10.3390/app9153169.

[16] J. Memon, M. Sami, R. A. Khan, and M. Uddin, “Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR),” IEEE Access, vol. 8, pp. 142642–142668, 2020, doi: 10.1109/ACCESS.2020.3012542.

[17] F. Lombardi and S. Marinai, “Deep Learning for Historical Document Analysis and Recognition—A Survey,” undefined, vol. 6, no. 10, p. 110, Oct. 2020, doi: 10.3390/JIMAGING6100110.

[18] V. Frinken and H. Bunke, “Continuous Handwritten Script Recognition,” Handbook of Document Image Processing and Recognition, pp. 391–425, Jan. 2014, doi: 10.1007/978-0-85729-859-1_12.

[19] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, “Convolutional neural networks: an overview and application in radiology,” Insights into Imaging, vol. 9, no. 4, pp. 611–629, Aug. 2018, doi: 10.1007/S13244-018-0639-9/FIGURES/15.

[20] S. Ashraf Zargar, “Introduction to Sequence Learning Models: RNN, LSTM, GRU”, doi: 10.13140/RG.2.2.36370.99522.

[21] A. Vaswani et al., “Attention Is All You Need”.

[22] J. Devlin, M.-W. Chang, K. Lee, K. T. Google, and A. I. Language, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, Accessed: Jan. 12, 2022. [Online]. Available: https://github.com/tensorflow/tensor2tensor

[23] A. Graves, A. Ch, S. Fernández, F. Gomez, J. Schmidhuber, and J. Ch, “Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks”.

[24] B. Shi, X. Bai, and C. Yao, “An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition”.

[25] K. Simonyan and A. Zisserman, “VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION,” 2015, Accessed: Jan. 12, 2022. [Online]. Available: http://www.robots.ox.ac.uk/

[26] A. Graves, “Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks”.

[27] B. Gatos et al., “Ground-truth production in the transcriptorium project,” Proceedings - 11th IAPR International Workshop on Document Analysis Systems, DAS 2014, pp. 237–241, 2014, doi: 10.1109/DAS.2014.23.

[28] A. Fischer, E. Indermühle, H. Bunke, G. Viehhauser, and M. Stolz, “Ground truth creation for handwriting recognition in historical documents,” undefined, pp. 3–10, 2010, doi: 10.1145/1815330.1815331.

[29] A. Fischer, V. Frinken, A. Fornés, and H. Bunke, “Transcription Alignment of Latin Manuscripts using Hidden Markov Models”,

[30] Y. Chen et al., “Drop an Octave: Reducing Spatial Redundancy in Convolutional Neural Networks with Octave Convolution,” 2019.

[31] R. Valle and J. Poulos, “Character-Based Handwritten Text Transcription with Attention Networks,” Dec. 2017, doi: 10.1007/s00521-021-05813-1.

[32] Y. Deng, A. Kanervisto, J. Ling, and A. M. Rush, “Image-to-Markup Generation with Coarse-to-Fine Attention,” Sep. 2016,

[33] T. Bluche, J. Louradour, and R. Messina, “Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention”.

[34] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jegou, “Training data-efficient image transformers & distillation through attention.” PMLR, pp. 10347–10357, Jul. 01, 2021.

[35] H. Bao, L. Dong, and F. Wei, “BEIT: BERT Pre-Training of Image Transformers,” 2021

[36] J. Devlin, M.-W. Chang, K. Lee, K. T. Google, and A. I. Language, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” pp. 4171–4186.

[37] A. Dosovitskiy et al., “AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE,”2021.