OCFormer: A Transformer-Based Model For Arabic Handwritten Text Recognition

In this paper, we present an OCR approach that utilizes state-of-the art Deep learning techniques for the Arabic language.
We built a custom dataset of obfuscated and noisy images to imitate the noise in historical Arabic documents, with a collection of 30 million images paired with their ground truth.

The model utilizes both page segmentation and line segmentation techniques to enhance the resultant transcription.
The model is complex enough for transcribing handwritten manuscripts. In addition, the model can detect and transcribe documents that contain Arabic diacritics.

The model attained a **CER** of **0.0727**, a **WER** of **0.0829**, and a **SER** of **0.10**.

For a successful supervised training process, the training examples and ground truth (GT), such as line images and their corresponding transcriptions, must be prepared manually.

Recently, important milestones were the introduction of Recurrent Neural Networks (**RNNs**) with Long Short-Term Memory (**LSTM**) trained using a Connectionist Temporal Classification (**CTC**) decoder tailored for **OCRs**, Attention mechanisms; namely, Self-Attention, Transformers, and Linformer (that is, Self-Attention with Linear Complexity).
These methods have undeniably improved the recognition accuracy of characters and words/sentences.

In this paper, we investigate how **Transformers** can be employed to build an OCR for historical Arabic manuscripts that is capable of

1. appropriate line and character segmentation
2. proper transcription of handwritten manuscripts
3. the recognition of Arabic diacritics

Also, we will present a custom dataset created to combat the lack of ground truth for many historical Arabic manuscripts.

Related works

Because of the diversity of handwritten character shapes and types, algorithms designed to recognise handwritten characters have had less success than those designed to recognise printed characters.

Ahmad et al.
experimented on the **APTI** dataset using an adaptive window for feature extraction of both characters and words , achieving an **Error Rate of 0.57%** for the **characters** level, and **2.12%** for **words**.

Furthermore, they experimented using the **KHATT** database , attaining the **highest CER** of **1.04%** for the **character** recognition.

They also employed **LSTM** and its variants **Bi-LSTM** and **MDLSTM** on the **KPTI1** data set , evaluating using both normalized and non-normalized data (the **CER** using MDLSTM was **9.22%**).

Mahmoud et al
Using the **KHATT** dataset, developed a **Hidden Markov Model (HMM)** recognition system.
They used text-line image pixel density, horizontal vertical edge derivatives, statistical features, and gradient features.
They achieved a **51.2 %** for **character recognition** using gradient features.

Elleuch et al
presented a model that employs **CNNs** along with **SVMs** for Arabic handwriting recognition.
To measure the model's performance, they used both the **HACDB** and the **IFN/ENIT** datasets.
The authors investigated the **CNN-based SVM model** s performance (for Arabic characters recognition) with and without dropout.
The **Error rate** was **5.83%** and **7.05%** using the **HACDB** and **ENIT** datasets respectively.

Aziz et al
Using the **APTID-MF** dataset.
implemented a segmentation-based, omni-font For **printed Arabic text**, **open-vocabulary OCR** was used, with an **average accuracy of 95%**. The **recognition stage has an overall accuracy of 99.97%** without using font-type or other post-processing techniques.

Ahmed et al.
proposed a machine learning model that selects the best features for Arabic handwritten character recognition by **combining neighbourhood** rough sets with a **binary whale optimization algorithm**.

Their **Linear Discriminant Analysis (LDA)** model achieved a **recognition accuracy of 96%**. using the **CENPARMI** dataset

Dataset
The constructed dataset consists of images containing Arabic text collected from the web along with their ground truth.

A portion of the text includes

- Arabic diacritics.
- Multiple Arabic fonts that closely resemble the old fonts used in historical manuscripts (dating back to the 18th century) are used.

There are four categories of images:

1. Full sequences (i.e., images with more than five words)
2. Short sequences (i.e., images that have five or fewer words), Full sequences
3. with diacritics (the images with more than five words with diacritics)
4. Short sequences with diacritics (images with five or fewer words, with diacritics).

The handwritten manuscripts from the **KHATT** database, is also included (KHATT contains unconstrained handwritten Arabic Texts written by 1000 different writers).

The rationale for this is to train the model using all types of sequences and sentence positions that may appear in historical manuscripts.

For example, marginal notes are Short sequences while artistic borders are Full sequences, thus ensuring that the model will be trained using all possible types of texts, and that will help in the segmentation process of handwritten manuscripts.

Figure illustrates the various categories of the dataset.

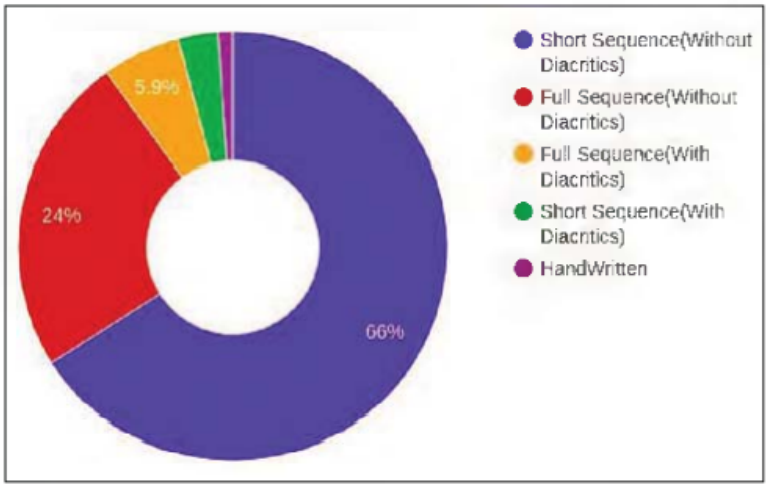

Fig. 2. The Percentages Of The Dataset

Methodology

A. Image Preprocessing

- Image augmentation
  a well-known type of data augmentation that increases the images available for the training model, without collecting any new data.
  The augmentation can be done with cropping, padding, horizontal flipping, zooming, and rotating the image
  Shearing and **Changing** the Luminosity are two commonly applied augmentation approaches when training large neural networks

- Image Segmentation
  two types of line segmentation are used. procedures are applied.
  Layout segmentation is applied, where the model segments areas containing valid text from a page.
  line segmentation, where the model segments the output from the layout segmentation (i.e., text areas) into segments each corresponding to sentence (i.e., line) detected within the image.

Fig. 3. Example Segmented images of historical printings

B. OCR Model

In the context of Deep Learning (DL) techniques for image analysis, **CNNs** are of main importance.
Therefore, we trained a **CNN** model (**Resnet101**) with **101 layers** as a feature extractor, resulting in the feature vectors that are crucial for the subsequent Transformer Encoder phase.

Transformers are a form of neural network architecture that gained popularity after the Self-Attention mechanism was introduced (i.e., focusing on a subset of the information they get).
An RNN, for example, can monitor the performance of another RNN.
It focuses on different positions of that other RNN at each time stage.
A **Transformer** is a model that uses Self-Attention to increase the speed of training in Deep Learning models.
Transformers also solve the parallelization problem.

The proposed OCR Model utilizes CNNs together with a Transformer.

Transformers consist of two modules, an encoder, and a decoder, which have customizable parameters that can affect a model's performance based on its task.
Our proposed model is initialized with four encoders, four decoders, four attention heads, and 256 hidden dimensions.

The **encoder module** receives the features extracted by the **CNN** as an input, then performs embedding and positional encoding, creating a sequence input that can be passed to the **Multi-Head Attention**.

The **Multi-Head Attention layer** performs Self-Attention, then adds a layer-normalization before passing the feature vector (of 2048 elements) to a feed-forward network that applies layer-normalization on residuals.

These operations are performed four times (because the model is initialized with four encoders).

Regarding the **decoder module**, its input is the text to be extracted from the images, which will undergo similar operations to that of the encoder. The embedding layer performs embedding and positional encoding, creating the sequence that will be passed to the **masked Multi-Head Attention** layer.

The masked Multi-Head Attention layer performs Self-Attention on the text sequence, then adds layer-normalization before performing cross attention between the encoder's output and the text sequence received from the Multi-Head Attention of the decoder. Then it applies layer- normalization and passes the vector to a feed-forward network and a linear layer.
Finally, a **Softmax** function produces the probabilities for the predicted words.

Proposed Dataset

The proposed dataset is a large, multi-font, and multi-style, text recognition dataset in Arabic. Various considerations must be met while constructing the dataset to ensure the diversity of the writing styles. That includes different fonts, different styles, and different noise patterns on the characters used while generating the images. The database is constructed using 13 Arabic fonts and multiple font styles. The dataset contains 30.5 million single-line images, including more than 270 million words and 1.6 billion characters. The ground truth, style, and font used to generate each image are available.The generated text images vary according to the following:

1. Twelve Different Fonts: Alsamt Diwani, Barada Reqa, Diwani Letter, HSN Naskh Farisi, M Unicode Abeer, Mj Ghalam, Nawel, Old Antic Outline, Phalls Khodkar, Sahel, Tarwat Emara Ruqaa Hollow, and Tarwat Emara Ruqaa Light.
2. Thirteen Different Sizes; one for each font.
3. Multiple Styles: Bold, Plain, Italic, Italic and Bold, etc.
4. Various Forms of Overlap among characters due to the different fonts and large combinations of words and characters.
5. Enormous Vocabulary, which allows the models to be tested on novel unseen data.
6. Multiple downsampling and antialiasing filter artefacts caused by the random addition of white-pixel columns at the beginning and end of lines in the images.
7. Each image's height and width are variable.The last point in a word is crucial to the series of characters that appear in it. There is no prior knowledge of the text's location in the picture, so the baseline must be calculated by the recognition model

Proposed Model

In this section, we will emphasize some details of the model.

According to Vaswani et al, and Devlin et al
the model's performance to generalize to unseen data increases proportionally with the number of parameters, which are the

1. hidden dimensions
2. number of encoder layers
3. number of decoder layers
4. number of heads.

The base transformers' model of Vaswani et al. achieved a BLEU of 38.1 (in the English-to-French), while the big transformers' model achieved a BLEU of 41.8, and the same follows with the BERT model [3].

We employed the Adam optimizer with weight decay, and label smoothing.
The learning rate was initialized to 0.0001, then an adaptive learning rate technique was applied (as follows: if the validation loss increases for 3 consecutive iterations, the learning rate changes according to the LR scheduler).
The model was trained for 700 update steps.

In Sequence-to-Sequence (seq2seq) models, The most widely used decoding algorithms are greedy search and beam search [29]. At each time level, the greedy search seeks out the token with the highest conditional probability. Beam search, on the other hand, is a better version of greedy search that, unlike greedy search, selects the k highest conditional probabilities at each time level. With each passing time phase, Based on the k tokens from the previous time stage, it chooses the k highest conditional probabilities.

FUTURE WORKS

In this research, we proposed and investigated a novel approach for harnessing the power of Transformers to transcribe historical Arabic manuscripts. It is expected that this approach will help speed up, and increase the accuracy of, the OCR process.

In our future work, we aim to improve the model's accuracy by training the model with the entire constructed dataset (allowing the model to train on all the data will significantly increase the accuracy of the OCR). As well, we aim to increase the number of encoders, decoders, and other parameters that can highly affect the model's performance and accuracy. Finally, we aim to optimize the model's time and space complexity from quadratic complexity to linear complexity.