Transformer-based Models for Arabic Online Handwriting Recognition

Fakhraddin Alwajih[1,2] *, Eman Badr[1,3], and Sherif Abdou[1]
Department of Information Technology, Cairo University, Giza, Egypt[1]
Department of Computer Science and Information Technology, Ibb University, Ibb, Yemen[2]
University of Science and Technology, Zewail City of Science, Technology and Innovation, Giza, Egypt[3]

***Abstract*—Transformer neural networks have increasingly become the neural network design of choice, having recently been shown to outperform state-of-the-art end-to-end (E2E) recurrent neural networks (RNNs). Transformers utilize a self-attention mechanism to relate input frames and extract more expressive sequence representations. Transformers also provide parallelism computation and the ability to capture long dependencies in contexts over RNNs. This work introduces a transformer-based model for the online handwriting recognition (OnHWR) task. As the transformer follows encoder-decoder architecture, we investigated the self-attention encoder (SAE) with two different decoders: a self-attention decoder (SAD) and a connectionist temporal classification (CTC) decoder. The proposed models can recognize complete sentences without the need to integrate with external language modules. We tested our proposed models against two Arabic online handwriting datasets: Online-KHATT and CHAW. On evaluation, SAE-SAD architecture performed better than SAE-CTC architecture. The SAE-SAD model achieved a 5% character error rate (CER) and an 18%word error rate (WER) against the CHAW dataset, and a 22% CER and a 56% WER against the Online-KHATT dataset. The SAE-SAD model showed significant improvements over existing models of the Arabic OnHWR.**

***Keywords*—Selft attention; Transformer; deep Learning; connectionist temporal classification; convolutional neural networks; Arabic online handwriting recognition**

I. Introduction

OnHWR is essentially a task of converting digital input handwriting into digital text. Handwriting recognition can be classified into two main categories based upon input data: online and offline handwriting recognition. In online handwriting, data is represented as a series of points with the precision of other information, such as timestamps, dependent upon the capabilities of the input device. In offline handwriting recognition, data is represented as images scanned from documents.

In recent years, OnHWR has attained increased importance concomitant with rapid developments in related hardware and software. Most current communication software supports notetaking and writing on boards using online handwriting as both a communication media and a vehicle of computer-aided education. In the rising markets, greater access to computing devices has allowed ever-increasing populations to connect across the internet, with many depending solely on mobile devices with touchscreens. Handheld devices with styluses are becoming more widely available and used in many domains. Concomitantly, there have been tremendous advances in prime technologies of deep learning and natural language processing (NLP) algorithms. Such advances have led, in turn, to considerable progress in the field of OnHWR. The Arabic language is spoken by around half a billion people around the world. A number of other languages, including, Urdu, Persian, Kurdish, and Pashto adopted and use Arabic script. Arabic is a 'right to left' language in its written form. It consists of 28 letters, 10 digits as well as a number of punctuation marks. Each Arabic letter has four contextual forms, depending upon its position in a word: isolated, beginning, middle, and end position forms, as shown in Fig. 1. Arabic OnHWR is a challenging problem for multiple reasons. One reason is the existence of a wide range of variations in handwriting styles, in part due to the existence of multiple calligraphies in Arabic. There are eight basic calligraphies in Arabic script [1]. The tendency is to use a combination of these calligraphies when writing in Arabic. This further compounds the variations in styles of writing, thus adding to the challenges that would face the developer of an Arabic script recognition system. Compared to Latin and Chinese and other scripts, published work in the Arabic OnHWR field has to date been fairly limited.

OnHWR is a sequence-to-sequence (S2S) classification task. Input frames are ingested into the S2S model which in turn generates text. Recent advances in S2S models have shown their reliability solve complex NLP tasks such as translation [2] and automatic speech recognition (ASR) [3]. Additionally, the performance of OnHWR systems has improved with the advent of deep learning models including convolutional neural network (CNN) [4] and long short-term memory (LSTM) [5], [6].

Recently, E2E OnHWR systems have achieved remarkable performance, with input handwriting features being mapped directly to an output sequence of letters or tokens. In E2E systems, all components are trained and optimized jointly, thus reducing the complexity of the system and minimizing error propagation between components compared to conventional hybrid systems. Using CTC, E2E modeling has been utilized for handwriting recognition tasks as well as attention-based encoder-decoder systems designed for mathematical expression recognition tasks [7], [8]. Moreover, E2E has been incorporated with external language models (LM), effectively boosting performance [5]. In general, the competitive performance obtained by E2E models and their simplicity facilitate the building of state-of-the-art OnHWR systems. In this work, we explore building an E2E OnHWR system based on self-

*Corresponding authors.

Contextual forms				
Isolated	End	Middle	Beginning	Name
ا	ـا	-	-	*Alif*
ب	ـب	ـبـ	بـ	*Beh*
ت	ـت	ـتـ	تـ	*Teh*
ث	ـث	ـثـ	ثـ	*Theh*
ج	ـج	ـجـ	جـ	*Jeem*
ح	ـح	ـحـ	حـ	*Hah*
خ	ـخ	ـخـ	خـ	*Khah*
د	ـد	-	-	*Dal*
ذ	ـذ	-	-	*Thal*
ر	ـر	-	-	*Reh*
ز	ـز	-	-	*Zain*
س	ـس	ـسـ	سـ	*Seen*
ش	ـش	ـشـ	شـ	*Sheen*
ص	ـص	ـصـ	صـ	*Sad*
ض	ـض	ـضـ	ضـ	*Dad*
ط	ـط	ـطـ	طـ	*Tah*
ظ	ـظ	ـظـ	ظـ	*Zah*
ع	ـع	ـعـ	عـ	*Ain*
غ	ـغ	ـغـ	غـ	*Ghain*
ف	ـف	ـفـ	فـ	*Feh*
ق	ـق	ـقـ	قـ	*Qaf*
ك	ـك	ـكـ	كـ	*Kaf*
ل	ـل	ـلـ	لـ	*Lam*
م	ـم	ـمـ	مـ	*Meem*
ن	ـن	ـنـ	نـ	*Noon*
ه	ـه	ـهـ	هـ	*Heh*
و	ـو	-	-	*Waw*
ي	ـي	ـيـ	يـ	*Yeh*

Fig. 1. Arabic Letters and their Contextual Forms.

attention models.

RNNs have been adopted for sequence modeling and have provided remarkable accuracy in multiple NLP tasks [9], [2], [10]. RNNs have been extensively utilized in OnHWR, including in the build-up of LSTM and gated recurrent units (GRU). In RNN, each hidden state depends on the previous one which makes parallelizing computations of RNNs difficult. Additionally, the hidden states are condensed into a fixed-length vector which introduces a 'bottleneck' making capturing long dependencies difficult as well [10].

As alternatives to RNNs, transformer-based models [11] have recently yielded outstanding results, achieving state-of-the-art performance in a variety of NLP tasks, including text and image-related tasks, and ASR [12], [11], [13]. Transformers rely on a self-attention mechanism, which extracts a more representative sequence by relating all position pairs of an input sequence. The self-attention mechanism offers two attractive features compared with RNNs: (1) computations can be parallelized and carried out efficiently through batched tensor operations, and (2) self-attention allows direct connection for long-range and short-range dependencies without propagating contextual information between intermediate hidden states (as in case of RNNs) [11]. In addition to self-attention, the transformer model utilizes multi-head attention (MHA) in order to learn different representations in one instant. As with RNNs attention-based models, transformers are architecturally designed as encoder-decoder models, with both the encoder and decoder containing stacked self-attention networks (SANs) on top of each other. The cross-attention mechanism is used to bridge between the encoder and the decoder. The successes of transformer models have inspired this work in which self-attention was applied to an OnHWR task.

In this paper, we introduce transformer-based models OnHWR for Arabic script. The proposed models can transform a full-sentence handwriting input sequence into the corresponding letter sequence. Basically, we applied CNN layers to subsample input sequence features (via convolution strides) and process local relationships between handwriting frames of the input sequence. The output is added to positional embedding output to maintain input orders and then fed into the self-attention encoder (SAE). For the decoder, we employed two decoders: the self-attention decoder (SAD) and the CTC decoder. The proposed models were trained and evaluated against two datasets: Online-KHATT dataset [14] and CHWA dataset [15]. To the best of our knowledge, there has been no prior work on OnHWR that has proposed or applied self-attention models. As far as we are aware, this is the first attempt to apply the transformer model to an OnHWR task. Our results show that our proposed SAE-SED model can outperform existing RNNs models.

The main contributions can be summarized as follow:

- We introduce a new self-attention-based non-recurrent neural network models for OnHWR task.
- Two architectures have been developed in the decoding stage for the transformer: a SAD decoder and a CTC decoder.
- The proposed models have been evaluated against a full sentence (OnKHATT) [14] and a word-based (CHAW) Arabic dataset [15]. Results were compared with existing models, with our model evidently outperforming these models.

The rest of this paper is structured as follows: Section II details related work previously conducted on OnHWR. In Section III we layout the architecture of the transformer we designed for the OnHWR task. Then, experimental results are presented in Section IV. Lastly, Section V details our conclusions and recommendations for possible future work.

II. Related Work

OnHWR data have a temporal structure and can be represented as a sequence of geometrical features vectors over time. OnHWR relies on sequence modeling, including statistical modeling. Previously, hidden Markov models (HMMs) have been reportedly utilized to model online handwriting in multiple published works. In [16], the HMM was designed

to model stroke segments as handwriting model units. In their model, letter models are subsequently formed by concatenating model units as defined in a pronunciation dictionary. Letter models are integrated as word sequence probabilities to form a stochastic language model. In [17], The researchers integrated Gaussian mixture models (GMMs) with the HMMs as continuous HMMs, using the GMMs to estimate observation probability distributions emitted by HMM states. Hybrid HMMs with feed-forward neural networks (NNs) were also part of their design [18]. Authors in [19] integrated a time-delay neural network (TDNN) with an HMMs into a single architecture, combining the recognition and segmentation phases into a single hybrid architecture. In their model, this hybrid architecture was intended to utilize the power of TDNN in the recognition and power of HMMs in the segmentation.

Traditional approaches involve multiple components that are separately trained and optimized, introducing suboptimality. On the other hand, deep learning models work on feature representation by learning discriminative representation from the raw data, thus providing an E2E solution for concomitant training of OnHWR system components jointly. One of the first works used implicit segmentation that was to be trained jointly with the recognition phase in [20]. In [20], the connectionist temporal classification (CTC) loss was introduced as an objective to map input frames into letters and optimize recognition jointly with LSTM.

Deep CNN was also utilized by [21]. In this study, the authors integrated CNN with domain-specific technologies to form an integrated network to improve performance. The efficacy of a combination of a CNN, RNN, and CTC was also investigated by [22]. They placed CNN layers at the front in order to support features representation. Next, they added LSTM to model the sequence of OnHWR along with a CTC to optimize the integrated network in an E2E manner. Authors investigated handcraft features and raw data ingested to CNN and reported that the proposed model had performed better with handcraft features. Furthermore, in [1], CNN-BiLSTM-CTC architecture was used to design an Arabic OnHWR model.

Recent work by Google investigated a model consisting of bidirectional LSTM (BiLSTM) with CTC [5]. In this work, the authors utilized a BiLSTM encoder trained using CTC loss. In decoding, they used different scorning LMs to incorporate prior knowledge about the underlying language and decode the output of the RNN encoder. GRU was employed in S2S with attention architecture and used in recognition tasks of online handwriting data of mathematical expression [8], [23], which was originally used in neural machine translation (NMT) by [24]. In [25], authors utilized attention encoder-decoder to recognize unconstrained Vietnamese Handwriting. The encoder was fronted with a CNN to extract invariant features and a BiLSTM to encode the output of CNN. The decoder was composed of BiLSTM layers with attention incorporated with encoders in order to generate text output. In [26], an edge graph attention network (EGAT) was proposed as a model that would perform stroke recognition. Stroke classification was formulated as node classification in a graph neural network (GNN).

In Arabic OnHWR, the line of work simply flows the Latin OnHWR workflow [27], [28]. As with traditional OnHWR systems, HMMs was utilized in many works for Arabic OnHWR [29], [30], [31], [32], [33]. Hybrid NN/HMMs were investigated in [34] and a DNN/HMMs model was tested by [15]. CTC based models were employed in several works for Arabic OnHWR [35], [36], [37], [1]. Most of the aforementioned studies targeting Arabic OnHWR tested their models against word-based datasets, with the exception of our previous work [35], [1] in which we tested our models against both sentence-based and word-based datasets. In our previous work [35], we proposed an E2E BiLSTM-CTC model and incorporated LM with outputs of RNNs to boost the performance of the system. More recently, we developed a writer adaptation method that utilized an E2E CNN-BiLSTM-CTC model [1]. In the current work, we did not integrate with any external module, and we evaluated our work against the CHWA and Online-KHATT datasets.

Variations can be reduced by normalization preprocessing steps. Normalization acts by reducing geometric variants in order to facilitate extracting features that are relevant to recognition. In the literature on OnHWR, multiple normalization methods, including slant correction, smoothing using a Gaussian filter, and resampling, have been proposed and tested against OnHWR data. The most comprehensive preprocessing steps were detailed by [38].

Features extraction refers to the process of extracting a meaningful set of features from raw data to be ingested and eased in the recognition phase. In OnHWR, traditional features can be classified into local features per point and global features per stroke or character [38]. Recently, as deep learning helped perfect features representation, the need for handcraft features with learnable features representation was eliminated in such areas as NLP [39], ASR [3], and computer vision [40]. Two recent works in which the authors used deep learning for features representation are [5], [21]. Despite its advantages, deep learning needs large-scale datasets to learn features representation and OnHWR datasets are rare and limited in size.

To summarize, state-of-the-art OnHWR models based on deep recurrent networks have begun to achieve remarkable recognition results, although training is computationally expensive and takes a long time to converge. Furthermore, the problem with pure RNN methods is that information may be forgotten during the encoding process, thus degrading overall model performance. In this work, we propose the usage of transformer-based models for the OnHWR task for the first time with no-recurrent design. A single, unified E2E architecture capable of recognizing full sentences from input online handwriting without the need for predetermined lexicons or language models.

III. Transformer for OnHWR

Typically, the OnHWR is an S2S task in which the lengths of input and output can differ. In our framework, the architecture of the transformer is based on an encoder-decoder structure. Given the handwriting input sequence $X = (x_1, x_2, ..., x_{T^{in}}), x_i \in R^{\mathrm{d_{in}}}$ where T^{in} is the length of input sequence and d_{in} is the number of features. Before feeding the input into the encoder, we prepended the encoder with CNN layers to extract better representative handwriting features and

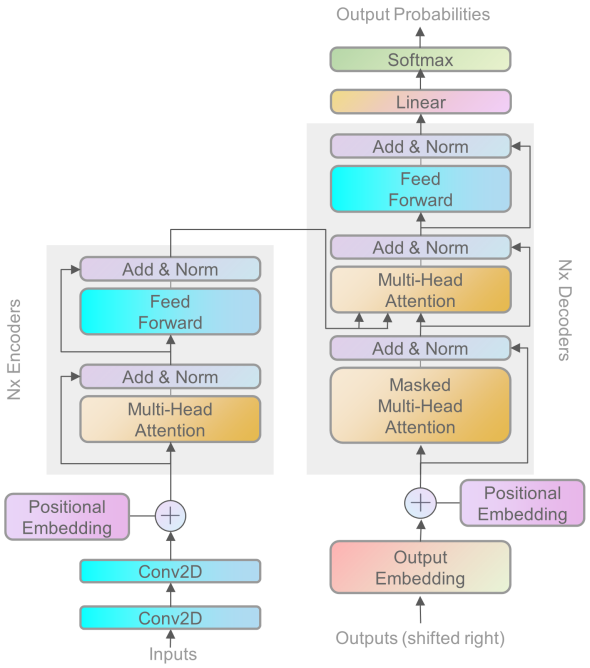

Fig. 2. The Architecture of OnHWR Transformer.

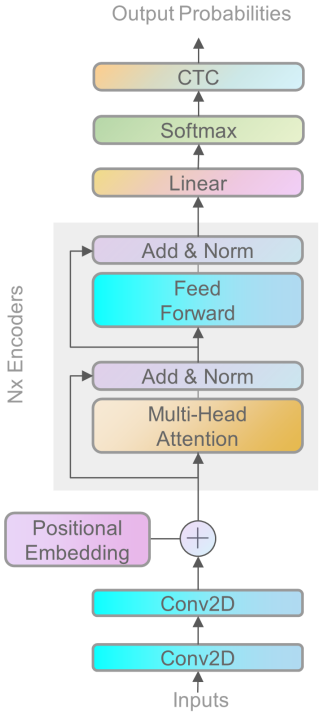

Fig. 3. Self Attention Encoder (SAE) with CTC.

perform subsampling on the input handwriting frames. Then, we applied positional embedding to the output of CNN layers $X = (x_1, x_2, ..., x_T), x_i \in R^{\mathrm{d_{model}}}$, where $T = |X|$ after subsampling. Positional embedding affords the input sequence a perception of order. The encoder was designed to encode the input sequence $X = (x_1, .., x_T), x_i \in R^{\mathrm{d_{model}}}$ and generate intermediate updated representation using a self-attention mechanism $h = (h_1, h_2, ..., h_T), h_i \in R^{\mathrm{d_{model}}}$. The decoder transduces the output sequence using autoregressive approach. Given h representation and previously emitted characters of the decoder outputs to that point $y_{i-1} = (y_1, ..., y_{i-1})$, the decoder computes the next character y_i. This procedure is repeated until the end of the sentence token is emitted as shown in Figure 2.

We also examined using a CTC decoder instead of the transformer decoder in which h representation would be ingested directly into the linear output layer. The output $y = (y_1, ..., y_L)$ with length L is emitted by CTC decoder at once as shown in Fig. 3.

A. Self-Attention

The transformer-based models are built on a new concept of self-attention as an extension of attention introduced in S2S [24], [2]. Self-attention is a mechanism to compute updated representations for each sequence element in parallel. The attention mechanism would allow each representation to deferentially consider the representations in every other position in a sense, and the communication paths would have the same length for all pairs of elements. The attention mechanism consists of a query matrix Q, a key matrix K, and a value V matrix. The basic idea is that a query vector would be compared to a set of key vectors to determine their rapport. Each key vector comes paired with a value vector. The greater the rapport of a given key with the query the greater influence the corresponding value would have on the output of the attention mechanism. Transformer employs **scaled dot product attention** to map a query with a series of key-value pairs to output using the following equation:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

where $Q \in R^{\mathrm{M \times d_k}}$and $K, V \in R^{\mathrm{N \times d_k}}$ denote queries, keys and values in the matrix form, M and N are the number of queries and key-value pairs, and d_k is the dimension of representation. Scaling by factor $\sqrt{d_k}$ is done to prevent extremely small gradients.

B. Multi Head Attention (MHA)

Using a single attention head, the linear combination of value vectors leads to an averaging outcome that restricts the resolution of the learned representations. Therefore, the authors propose using multiple attention heads that can simultaneously learn different representations to alleviate this. MHA is computed as follows:

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \\ \text{where head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \\ W_i^Q &\in R^{\mathrm{d_{model} \times d_k}}, \\ W_i^K &\in R^{\mathrm{d_{model} \times d_k}}, \\ W_i^V &\in R^{\mathrm{d_{model} \times d_v}}, \\ W_i^O &\in R^{\mathrm{hd_v \times d_{model}}} \end{aligned} \tag{2}$$

First, the inputs: query matrix Q, key matrix K and value matrix V are linearly projected using W^Q, W^K and W^V. The projected query QW^Q, key KW^K and value VW^V are split into h heads. Scaled dot-product attention is computed for each head i. The independent attention head computed are then concatenated and linearly projected using W^O.

C. Self-Attention Encoder (SAE)

Instead of positional encoding proposed in the original paper, we adopted learnable positional embedding [41]. The positional embedding has the same dimensionality as the input embedding, and we summed them together before feeding them to the encoder. MHA is the first of two sub-layers of an encoder layer. After each sublayer, both residual connection and layer normalization were applied. The residual connection adds a copy of the input to the output, which means the input representations before an MHA block are added to the output representations. Then layer normalization takes the input vectors and essentially normalizes each one individually to have zero mean and variance. This is done to assure training stability. The second sub-layer is a **position-wise feed-forward network**, composed of a simple network of two fully connected layers with value activation between them to each input representation as follows:

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2 \tag{3}$$

After the second sub-layer, we again applied a residual connection and layer normalization, thus completing one encoder layer. The aforementioned layers can then be stacked up N times to form the full encoder.

D. Self Attention Decoder (SAD)

The design of the self-attention decoder mimicked that of the aforementioned encoder architecture, except that it is composed of two MHA layers. The first MHA layer applies attention to outputs generated by the decoder up to a point. The first layer is masked to avoid attending to future positions, while the second MHA layer applies attention to the encoder outputs.

E. The CTC Decoder

CTC objective loss was described by [7], [20]. CTC directly estimates prediction labels in E2E models without the need for explicit segmentation or alignment between input frames and output labels. As with the RNN encoder [35], the encoder (SAE) outputs sequences with the same length of input sequence frame length of the input. CTC manages this condition by introducing an additional blank label b symbol to the target labels and allowing repetition of labels or by adding banks across frames to match the lengths of input frames. Given input handwriting frames $\mathbf{x} = (x_1, .., x_T)$,where $T = |\mathbf{x}|$ and $x_t \in R^{\mathrm{d}_{\mathrm{model}}}$ and output sequence labels $\mathbf{y} = (y_1, .., y_L)$,where $L = |\mathbf{y}|$ and $y_l \in Z$ and Z denote the (finite) label alphabet, the encoder (SAE) generates posteriors $P(\mathbf{y}|\mathbf{x})$ as follows:

$$P(\mathbf{y}|\mathbf{x}) = \sum_{\hat{\mathbf{y}} \in H_{CTC}(x,y)} \prod_{t=1}^{T} P(\hat{y_t}|x_1, .., x_T)$$

where $\hat{\mathbf{y}} = (\hat{y_1}, .., \hat{y_T}) \in H_{CTC}(\mathbf{x}, \mathbf{y}) \subset \{Z \cup b\}^T$ harmonizes to any possible paths under the condition that $\hat{\mathbf{y}}$ yields $\mathbf{y}$ after dropping the blank symbols b and repeated successive symbols of $\hat{\mathbf{y}}$. The CTC loss assumes that each label in the output sequence is conditionally independent given the input handwriting sequence.

IV. Experiments and Results

A. Datasets

We tested our models against two open vocabulary datasets, the Online-KHATT and CHAW. The Online-KHATT dataset is an open vocabulary dataset collected by KFUPM [14]. It is comprised of 10,040 sentences of Arabic text written by 623 writers using Windows and Android run devices. Writers that contributed to the Online-KHATT dataset represent different ages, education levels, nationalities, genders, and handedness. This dataset consists of natural and unrestricted handwriting styles. The Online Arabic handwriting Cairo University dataset (CHAW) [15] is a word-level collection of Arabic writing. CHAW was collected using Android Samsung tablets. It consists of 18k of distinct words within a total of 192k samples. These samples are split into a training set, consisting of 17k unique words within a total of 180k samples, and a testing set, consisting of 500 unique words within 12k samples. A total of 1250 writers had contributed to this dataset. These writers are of varied ages, genders, and handedness.

B. Model Description

For input features, all preprocessing steps and features described in [17], [20] are used except delayed strokes representation features. The input sequence consists of a vector of 20 features per point. We normalized the input samples using z-score normalization before samples are fed into the models. For output, we adopted 160-character units, including 28 Arabic characters and their variations at different positions within a word, numbers, blank, punctuations, a start of sentence label (SOS) and end of sentence label (EOS).

We placed 2 CNN layers for the purposes of handwriting feature embedding. To stabilize training, we applied batch normalization (BN) [10] after each CNN layer, followed by ReLU activation. In our models, CNN layers perform subsampling by time reduction of the input frame handwriting sequence and retaining more representative features.

For the SAE-SAD model shown in Fig. 2, we stacked 6 SAE encoders and only one SAD decoder layer. In addition, we used $h_{heads} = 4$ for MHA. A total of 256 units comprised the feed-forward sub-layers. For the SAE-CTC model shown in Fig. 3, we mimicked the structure of the SAE-SAD model, replacing the SAD layer with a CTC as described in Section III-E.

In the training phase, we used an Adam optimizer with a learning rate scheduling [11]. In cross-entropy loss, which is used to optimize the SAE-SAD model, we applied label smoothing with a plenty factor of 0.1 [42]. SAE-CTC model optimized, the entire model using CTC loss. To avoid overfitting during training, dropout, at a rate of 0.3, is used [43]. In the end, we averaged the parameters of models of the last five epochs [44].

TABLE I. THE CERS COMPARISONS OF DIFFERENT HYPERPARAMETERS COMBINATIONS FOR SAE-SAD MODEL.

# of encoders	# of decoder	h_{heads}	d_{ff}	CER [%]
4	4	2	128	34.77 %
4	4	4	256	28.64 %
6	4	4	256	23.16 %
6	2	4	256	21.35 %
6	1	4	256	**20.03** %
6	1	4	512	25.88 %
8	1	4	256	23.47 %

TABLE II. COMPARING OUR MODELS TO OTHER HYBRID AND E2E SYSTEMS REPORTING ON ONLINE-KHATT AND CHAW.

Models	CHAW dataset		Online-KHATT dataset	
	CER [%]	WER [%]	CER [%]	WER [%]
DNN/HMMs [15]	-	25%	-	-
BiLSTM-CTC-LM [35]	4.08%	14.65%	12.24%	28. 35%
CNN-BiLSTM-CTC [1]	9.43%	34.48%	**18.49%**	59.94%
SAE-CTC	10.83%	40.68%	23.89%	78.17%
SAE-SAD	**5.70%**	**18.45%**	22.88%	**56.48%**

C. Results

We used the standard matrices word error rate (WER) and character error rate (CER) to evaluate our experiment results. WER is calculated by summing up insertions, substitutions, and deletions present in recognized words divided by the length of words in the target sentence. CER is calculated in a similar fashion, this time focusing on characters instead of words.

To select the best hyperparameters for our proposed models, we ran multiple experiments of different hyperparameter combinations, varying the number of blocks in encoders, feed-forward units in the sub-layers of each block, number of heads h_{heads} for the encoder and number of blocks of the decoder in SAE-SAD. For the subsampling CNN module, we followed the architecture and hyperparameters in [1]. Table I shows different configurations we had tried for SAE and SAD with the CER on the validation set.

We trained the SAE-SAD model for 228 epochs and SAE-CTC model for 60 epochs. Training stopped when models started overfitting. We then selected the best model with the lowest CER on the validation set. Fig. 4, shows a comparison of validation loss and training loss for both SAE-SAD and SAE-CTC models, respectively. We also compared CER and WER on the validation dataset for both the SAE-SAD and SAE-CTC models. We trained all models using a GeForce GTX 1080 Ti, and we conducted all experiments using a Tensorflow [45]. At this scale, 228 epochs of SAE-SAD model run over 112 hours, whilst SAD-CTC model took over 30 hours. In Fig. 4, we see that the SAE-CTC model converges faster than the SAE-SAD model. However, the SAE-SAE model took more epochs to converge, and its WER was superior to that of the SAE-CTC model. We also found that CER was closer to WER in the case of the SAE-SAD model than in the case of the SAE-CTC model, indicating the SAE-SAE model to be capable of capturing words more accurately at a higher rate than the SAE-CTC model.

The online-KHATT dataset is challenging and contains sentence-based samples and a subset of segmented characters. All previous works, [46], [47], [48], [35], [1] conducted their experiments against the character set in Online- KHATT except

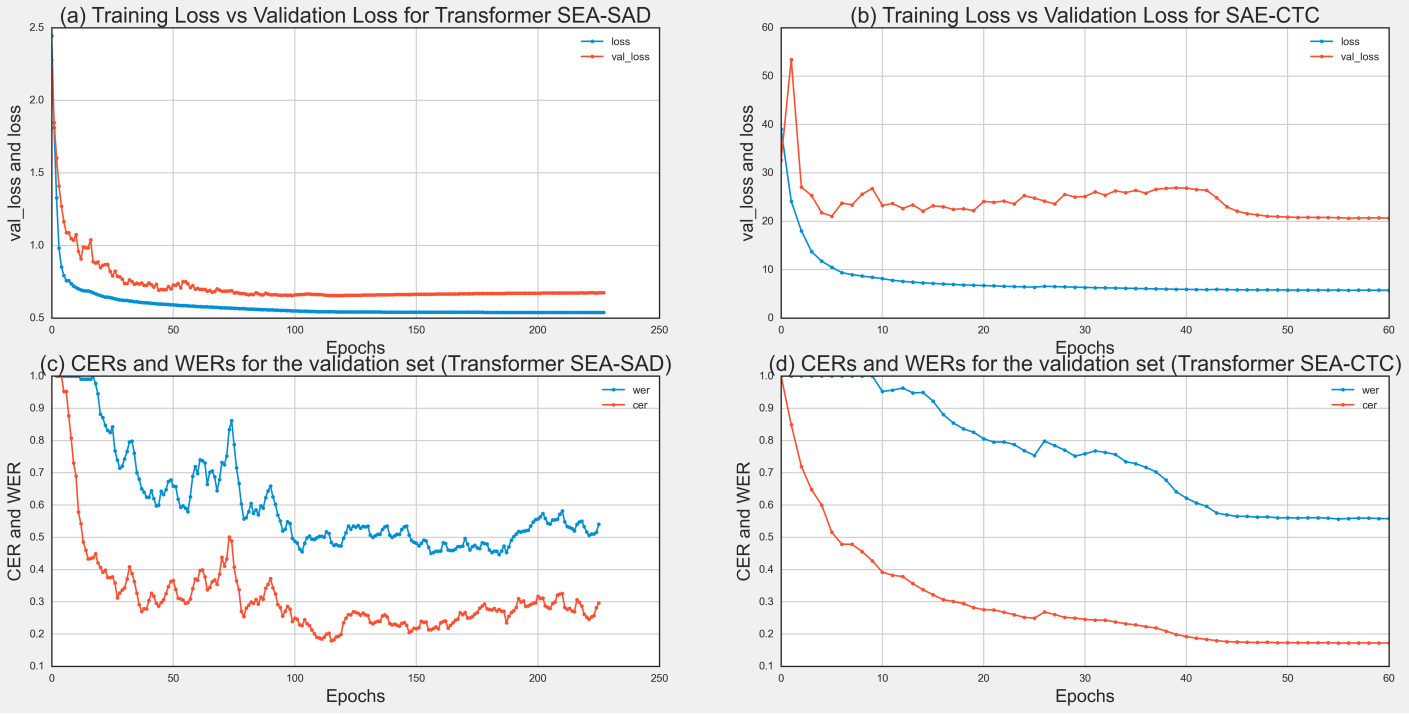

Fig. 4. (a) Training and Validation Losses for (a) SEA-SAD Model (b) SEA-CTC Model and CERs and WERs on Validation Dataset (c) SEA-SAD Model (d) SEA-CTC Model

[1], [35]. In our work, we compared our proposed models to existing systems that had tested their models against the full sentence-based set in the Online-KHATT dataset.

Table II shows the evaluation results on Online-KHATT and CHAW datasets. For the hybrid DNN/HMM-based approach in [15], the authors evaluated their work on the CHAW dataset, which is a word-based dataset. Furthermore, they integrated a dictionary with the model output to improve the results of the proposed approach. For the E2E system in our previous work [35], we incorporated n-gram LM to boost the result of the proposed approach, and we evaluated the proposed method on both Online-KHATT and CHAW datasets. Naturally, LMs boost the result of DNN models, and in this work, we have not incorporated any external LM or dictionary. Thus, this approach is not comparable to this work. The bottom row in Table II compares our previous CNN-BiLSTM-CTC [1] model with the proposed models since it does not integrate any external module. We compared our results with [1] results of the writer independent model as this result was achieved on the whole test dataset. The results show that SAE-SAD model outperforms our prior CNN-BiLSTM-CTC model [1]. Also, SAE-SAD outperforms SAE-CTC models. In addition, our proposed SAE-SAD performs better than the hybrid DNN/HMM model [15].

D. Discussion

Deep learning models learn to model discriminative features representation. As shown in Table I, deeper encoders perform better as we increase encoder layers. This is because each layer learns at a different level of abstraction for a given set of features. Multiple encoder layers are capable of generalization because each layer learns a different intermediate representation of raw data which helps at the classification level.

E2E CTC-based models are typically trained jointly using the loss CTC function. However, CTC-based models assume that relationships among produced labels from the CTC-based model are conditionally independent. Thus, such models cannot implicitly learn the LM from the training data. On the other hand, transformer-based models with a SAD decoder generate with each time step a label that is conditionally dependent on the previously generated ones. Consequently, they are capable of capturing the LM directly from training data. This would explain why the SAE-SAD model outperformed the SAE-CTC model, as shown in Table II. Also, we believe that SAE-SAD models could outperform traditional models that are integrated with external LMs in the presence of sufficient data. However, one advantage of the SAE-CTC model compared with the SAE-SAD model is its ability to generate the output labels in parallel at inference time.

CNN networks are widely used in transformer-based ASR models for down-sampling as well as providing positional encoding [13]. However, in the case of our OnHWR models, CNNs did not provide sufficient order information to the models other than that contributed through subsampling. Thus, we utilized positional embedding to add order sense to CNN outputs before feeding them into the encoder. The inability of CNN to provide sufficient order information may be due to the nature of handwriting data which contains delayed strokes, and the limited nature of Arabic handwriting datasets. We found that adding learnable positional embedding made the model converge faster.

V. Conclusion

In this work, we have introduced self-attention based Arabic OnHWR models. We trained and evaluated the proposed models against sentence-based and word-based datasets. We utilized different strategies and structures to improve the performance of models. Our transformer-based modes are actual E2E models following the S2S architecture with a self-attention encoder (SAE) and two decoders, a self-attention decoder (SAD), and a CTC decoder. Despite we did not incorporate any external modules such as an LM nor a dictionary into our architecture design, the proposed models are capable of recognizing complete sentences and words. Compared to state-of-the-art models, our transformer models have outperformed RNN models, which do not use LMs. Our best SAE-SAD model achieved a 5% CER and 18% WER against the CHAW dataset and 22% CER and 56% WER against the Online-KHATT dataset. Planned future work will involve investigating other features and expanding datasets by synthesizing new samples. We also plan to incorporate LM with transformer-based models to boost the performance.

References

[1] F. Alwajih, E. Badr, and S. Abdou, "Writer adaptation for e2e arabic online handwriting recognition via adversarial multi task learning," *Egyptian Informatics Journal*, 2022.

[2] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[3] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[4] X.-Y. Zhang, Y. Bengio, and C.-L. Liu, "Online and offline handwritten chinese character recognition: A comprehensive study and new benchmark," *Pattern Recognition*, vol. 61, pp. 348–360, 2017.

[5] V. Carbune, P. Gonnet, T. Deselaers, H. A. Rowley, A. Daryin, M. Calvo, L.-L. Wang, D. Keysers, S. Feuz, and P. Gervais, "Fast multi-language lstm-based online handwriting recognition," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 23, no. 2, pp. 89–102, 2020.

[6] S. Tabassum, N. Abedin, M. M. Rahman, M. M. Rahman, M. T. Ahmed, R. Islam, and A. Ahmed, "An online cursive handwritten medical words recognition system for busy doctors in developing countries for ensuring efficient healthcare service delivery," *Scientific reports*, vol. 12, no. 1, pp. 1–13, 2022.

[7] A. Graves, S. Fernández, F. Gomez, and J. 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.

[8] J. Zhang, J. Du, and L. Dai, "A gru-based encoder-decoder approach with attention for online handwritten mathematical expression recognition," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 902–907.

[9] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates *et al.*, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[10] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*. PMLR, 2016, pp. 173–182.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, . Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[12] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[13] A. Mohamed, D. Okhonko, and L. Zettlemoyer, "Transformers with convolutional context for asr," *arXiv preprint arXiv:1904.11660*, 2019.

[14] S. A. Mahmoud, H. Luqman, B. M. Al-Helali, G. BinMakhashen, and M. T. Parvez, "Online-khatt: an open-vocabulary database for arabic online-text processing," *The Open Cybernetics & Systemics Journal*, vol. 12, no. 1, 2018.

[15] O. Khaled, A. Fahmy, and S. Abdou, "Large vocabulary hybrid dnn/hmm arabic online handwriting recognition system," in *2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*. IEEE, 2017, pp. 876–881.

[16] J. Hu, M. K. Brown, and W. Turin, "Hmm based online handwriting recognition," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 18, no. 10, pp. 1039–1045, 1996.

[17] M. Liwicki and H. Bunke, "Hmm-based on-line recognition of handwritten whiteboard notes," in *Tenth international workshop on frontiers in handwriting recognition*. Suvisoft, 2006.

[18] Y. Bengio, Y. LeCun, C. Nohl, and C. Burges, "Lerec: A nn/hmm hybrid for on-line handwriting recognition," *Neural computation*, vol. 7, no. 6, pp. 1289–1303, 1995.

[19] M. Schenkel, I. Guyon, and D. Henderson, "On-line cursive script recognition using time-delay neural networks and hidden markov models," *Machine Vision and Applications*, vol. 8, no. 4, pp. 215–223, 1995.

[20] M. Liwicki, A. Graves, S. Fernàndez, H. Bunke, and J. Schmidhuber, "A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks," in *Proceedings of the 9th International Conference on Document Analysis and Recognition, ICDAR 2007*, 2007.

[21] W. Yang, L. Jin, Z. Xie, and Z. Feng, "Improved deep convolutional neural network for online handwritten chinese character recognition using domain-specific knowledge," in *2015 13th international conference on document analysis and recognition (ICDAR)*. IEEE, 2015, pp. 551–555.

[22] P. S. Mukherjee, B. Chakraborty, U. Bhattacharya, and S. K. Parui, "A hybrid model for end to end online handwriting recognition," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 658–663.

[23] J. Wang, J. Du, J. Zhang, B. Wang, and B. Ren, "Stroke constrained attention network for online handwritten mathematical expression recognition," *Pattern Recognition*, vol. 119, p. 108047, 2021.

[24] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[25] A. D. Le, H. T. Nguyen, and M. Nakagawa, "Recognizing unconstrained vietnamese handwriting by attention based encoder decoder model," in *2018 International Conference on Advanced Computing and Applications (ACOMP)*. IEEE, 2018, pp. 83–87.

[26] J.-Y. Ye, Y.-M. Zhang, Q. Yang, and C.-L. Liu, "Contextual stroke classification in online handwritten documents with graph attention networks," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019, pp. 993–998.

[27] A. Al-Salman and H. Alyahya, "Arabic online handwriting recognition: a survey," in *proceedings of the 1st international conference on internet of things and machine learning*, 2017, pp. 1–4.

[28] B. M. Al-Helali and S. A. Mahmoud, "Arabic online handwriting recognition (aohr) a survey," *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, pp. 1–35, 2017.

[29] F. Biadsy, J. El-Sana, and N. Y. Habash, "Online arabic handwriting recognition using hidden markov models," 2006.

[30] H. Ahmed and S. A. Azeem, "On-line arabic handwriting recognition system based on hmm," in *2011 International Conference on Document Analysis and Recognition*. IEEE, 2011, pp. 1324–1328.

[31] H. A. Abd Alshafy and M. E. Mustafa, "Hmm based approach for online arabic handwriting recognition," in *2014 14th International Conference on Intelligent Systems Design and Applications*. IEEE, 2014, pp. 211–215.

[32] I. Hosny, S. Abdou, and A. Fahmy, "Using advanced hidden markov models for online arabic handwriting recognition," in *The First Asian Conference on Pattern Recognition*. IEEE, 2011, pp. 565–569.

[33] M. Kherallah, N. Tagougui, A. M. Alimi, H. El Abed, and V. Margner, "Online arabic handwriting recognition competition," in *2011 International Conference on Document Analysis and Recognition*. IEEE, 2011, pp. 1454–1458.

[34] N. Tagougui, H. Boubaker, M. Kherallah, and A. M. Alimi, "A hybrid nn/hmm modeling technique for online arabic handwriting recognition," *arXiv preprint arXiv:1401.0486*, 2014.

[35] F. Alwajih, E. Badr, S. Abdou, and A. Fahmy, "Deeponkhatt: An end-to-end arabic online handwriting recognition system," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 35, no. 11, p. 2153006, 2021.

[36] R. Maalej, N. Tagougui, and M. Kherallah, "Online arabic handwriting recognition with dropout applied in deep recurrent neural networks," in *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*. IEEE, 2016, pp. 417–421.

[37] R. Maalej and M. Kherallah, "Improving the dblstm for on-line arabic handwriting recognition," *Multimedia Tools and Applications*, vol. 79, no. 25, pp. 17 969–17 990, 2020.

[38] S. Jaeger, S. Manke, J. Reichert, and A. Waibel, "Online handwriting recognition: the npen++ recognizer," *International Journal on Document Analysis and Recognition*, vol. 3, no. 3, pp. 169–180, 2001.

[39] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems*, vol. 26, 2013.

[40] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[41] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1243–1252.

[42] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[43] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[44] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson, "Averaging weights leads to wider optima and better generalization," *arXiv preprint arXiv:1803.05407*, 2018.

[45] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[46] D. Wilson-Nunn, T. Lyons, A. Papavasiliou, and H. Ni, "A path signature approach to online arabic handwriting recognition," in *2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)*. IEEE, 2018, pp. 135–139.

[47] Y. Hamdi, H. Boubaker, and A. M. Alimi, "Data augmentation using geometric, frequency, and beta modeling approaches for improving multi-lingual online handwriting recognition," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 24, no. 3, pp. 283–298, 2021.

[48] Y. Hamdi, H. Boubaker, B. Rabhi, W. Ouarda, and A. Alimi, "Hybrid architecture based on rnn-svm for multilingual online handwriting recognition using beta-elliptic and cnn models," 2021.