Computer Systems Science and Engineering

Tech Science Press

DOI: 10.32604/csse.2023.034520

ARTICLE

Computational Linguistics Based Arabic Poem Classification and Dictarization Model

Manar Ahmed Hamza[1,*], Hala J. Alshahrani[2], Najm Alotaibi[3], Mohamed K. Nour[4], Mahmoud Othman[5], Gouse Pasha Mohammed[1], Mohammed Rizwanullah[1] and Mohamed I. Eldesouki[6]

[1]Department of Computer and Self Development, Preparatory Year Deanship, Prince Sattam bin Abdulaziz University, AlKharj, Saudi Arabia

[2]Department of Applied Linguistics, College of Languages, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh, 11671, Saudi Arabia

[3]Prince Saud AlFaisal Institute for Diplomatic Studies, Riyadh, Saudi Arabia

[4]Department of Computer Science, College of Computing and Information System, Umm Al-Qura University, Makkah, Saudi Arabia

[5]Department of Computer Science, Faculty of Computers and Information Technology, Future University in Egypt, New Cairo, 11835, Egypt

[6]Department of Information System, College of Computer Engineering and Sciences, Prince Sattam bin Abdulaziz University, AlKharj, Saudi Arabia

*Corresponding Author: Manar Ahmed Hamza. Email: ma.hamza@psau.edu.sa

Received: 19 July 2022 Accepted: 13 November 2022 Published: 26 January 2024

ABSTRACT

Computational linguistics is the scientific and engineering discipline related to comprehending written and spoken language from a computational perspective and building artefacts that effectively process and produce language, either in bulk or in a dialogue setting. This paper develops a Chaotic Bird Swarm Optimization with deep ensemble learning based Arabic poem classification and dictarization (CBSOEDL-APCD) technique. The presented CBSOEDL-APCD technique involves the classification and dictarization of Arabic text into Arabic poetries and prose. Primarily, the CBSOEDL-APCD technique carries out data pre-processing to convert it into a useful format. Besides, the ensemble deep learning (EDL) model comprising deep belief network (DBN), gated recurrent unit (GRU), and probabilistic neural network (PNN) are exploited. At last, the CBSO algorithm is employed for the optimal hyperparameter tuning of the deep learning (DL) models to enhance the overall classification performance. A wide range of experiments was performed to establish the enhanced outcomes of the CBSOEDL-APCD technique. Comparative experimental analysis indicates the better outcomes of the CBSOEDL-APCD technique over other recent approaches.

KEYWORDS

Computational linguistics; Arabic poetry; classification; dictarization; ensemble model; parameter tuning; fusion process

1 Introduction

Poetry frames a huge body of literature from several renowned languages. Counties, even predating literacy, utilized poetry as an interaction and communication means [1]. Arabic poetry is a thriving, conventional literature which has an origin dating to prior the 6th century. Arab people will pay more attention to their art and celebrates gifted poets. Arabic writers compose poems to explain ideas, express emotions, give wisdom, pride and ridicule, motivate, flirt, record events, and praise and defame. Traditional Arabic poetry contains 16 m that varies in rhythm and target purposes. Chanting a poem articulately needs knowledge of the poem's meter and acquiring a discretized version of its verses (letters engraved with short vowels); diacritics were repeatedly not engraved in Arabic texts. The Arabic language was not an exception in terms of poetry. Many findings regarding the early Islamic and pre-Islamic Arabs were found via their poetry [2,3]. Arabs employed poetry to exhibit all detail of their life. Poetry is so valuable to them that they bring 7 legendary poems on 'Al Kaaba' curtains, their holiest location. It is even employed to collect on periodical festivals for reciting and promulgating their poems [4,5]. The Arabic language was the 6th most spoken language across many counties.

Arabic script orientation will be from right to left [6]. It contains twenty-eight letters, which include twenty-five consonants and three long vowels. Moreover, it involves certain special glyphs named diacritics. The diacritics in Arabic are split into 4 categories respective to their functionalities [7,8]. The initial category of diacritics involves a short vowel after every letter. The second category attaches a constant letter at the word end [9]. The 'Shadda' diacritic formed the third category and was employed for producing stressed letters. The 4th category was the diacritic 'Sukun', which can be utilized for indicating vowel absence [10].

The majority of the solutions for the automatic identification of Poem classifiers depend on Natural Language Processing (NLP) techniques [11]; there has recently been a leaning against using pure machine learning (ML) approaches such as neural networks for that task [12,13]. NLP methods contain the disadvantage of their complexity and, to a greater extent, rely on the language utilized in the text. It offers a robust motivation to use other ML methods for classifier tasks [14]. Additionally, the prevailing automatic techniques rely upon employing pre-trained vectors (e.g., Word2Vec, Glove) as word embedding for better performance from the classifier method. That makes identifying hatred content impracticable in cases where users deliberately obfuscate their offensive terminologies with short slang words [15].

In [16], a Deep Belief Network (DBN) can be leveraged as a diacritizers for Arabic text. DBN refers to one deep learning (DL) method that seems highly effective for several ML complexes. The author assesses the usage of DBN as a classifier in automated Arabic text discretizations. The DBN is well-trained to categorize every input letter with the respective diacritized versions separately. Madhfar et al. [17] offered three deep learning (DL) techniques for recovering Arabic text diacritics concerning this study in a text-to-speech synthesis mechanism utilizing DL. The primary method was a baseline method for testing how a simple DL executes over the corpora. The next method was related to an encoder-decoder structure that looks like this text-to-speech synthesis method having several alterations to suit this issue. The latest method was related to the encoder share of the text-to-speech method that attains exciting performance in diacritic error rate metrics and word error rates.

Fadel et al. [18] provided numerous DL approaches for the automated discretization of Arabic text. This technique was constructed utilizing 2 key techniques, viz. embeddings, recurrent neural network (RNN) and Feed-Forward Neural Network (FFNN), and has numerous improvements like Conditional Random Field (CRF), Block-Normalized Gradient (BNG), and 100-hot encoding. The techniques were tested on the only easily accessible benchmark data, and the outcomes exhibit

that these techniques are either on par or better with others, even those demanding human-crafted language-dependent postprocessing stages dissimilar to ours. In [19], the long short term memory (LSTM) method was employed to investigate the efficacy of neural network (NN) in Arabic NLP. The method was explicitly practical for recognizing part-of-speech (POS) tags for morphemes and Arabic words taken from the Quranic Arabic Corpus (QAC) data. QAC becomes a renowned gold standard dataset organized by authors from Leeds varsity. In [20], the researchers employed a Gated Recurrent Unit (GRU) and recurrent neural network (RNN), applying a simple gating system to improve Arabic discretization processes. Assessment of GRU for discretization can be executed compared to the exiting outcomes acquired with LSTM, an influential RNN structure receiving the familiar fallouts in discretization.

This paper develops a Chaotic Bird Swarm Optimization with deep ensemble learning based Arabic poem classification and dictarization (CBSOEDL-APCD) technique. The presented CBSOEDL-APCD technique involves the classification and dictarization of Arabic text into Arabic poetries and prose. Primarily, the CBSOEDL-APCD technique carries out data pre-processing to convert it into a useful format. Besides, the EDL model comprising deep belief network (DBN), gated recurrent unit (GRU), and probabilistic neural network (PNN) are exploited. At last, the CBSO algorithm is employed for the optimal hyperparameter tuning of the DL models to enhance the overall classification performance. A wide range of experiments were performed to demonstrate the enhanced outcomes of the CBSOEDL-APCD technique.

The rest of the paper is organized as follows. Section 2 introduces the proposed model, and Section 3 offers the performance validation. Lastly, Section 4 concludes the study.

2 Design of CBSOEDL-APCD Technique

In this study, a new CBSOEDL-APCD algorithm was introduced for classifying and dictarization of Arabic text into Arabic poetries and prose. The presented CBSOEDL-APCD technique includes data pre-processing, fusion process, and parameter optimization. It can clean social media posts during data pre-processing to remove unwanted symbols and noise. Fig. 1 showcases the overall process of the CBSOEDL-APCD approach. This step aims to maximize the count of words whose embedded was defined in the pre-trained word embedded technique.

The steps followed to clean the Arabic comments are as follows:

- Eliminating the stop words with a list of stop words including MSA and Dialect Arabic, i.e., (بز like), (هذ this), and) تىح until).
- Normalize the words and eliminate unwanted punctuation marks and symbols.
- Eliminating elongation and utilizing a single event in its place.

Concerning step 2, as the word embedded method does not comprise representation for emoticons, it can be established a mapping amongst known emoticons to their equivalent emojis. This approach cannot fail the sentiment stated by individuals' emoticons. Emojis were tokenized by assigning spaces among them; all the emojis are looked upon individually from the word-embedded method, which supports that once a set of emojis, without spaces between, perform in a comment, this integration may not have an equivalent word embedded.

2.1 Process Involved in EDL Model

This study's EDL model encompasses DBN, GRU, and PNN models. A fusion of three models helps in accomplishing enhanced Arabic poetry classification performance.

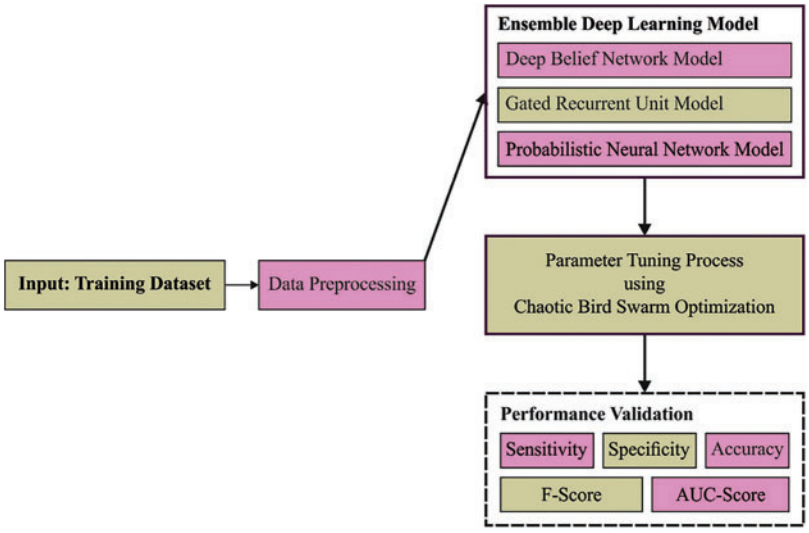

Figure 1: Overall process of CBSOEDL-APCD approach

2.1.1 DBN Model

DBNs can learn several layers of nonlinear features in unlabeled information. The high-order feature learned with the upper layer is extracting the hidden unit from the lower layer that is recognized with trained Restricted Boltzmann Machines (RBMs) from a greedy layer-wise approach by Contrastive Divergence technique and stacking them all over each other [21]. A generative DBN was capable of performing image in-painting and reconstruction.

Assume it takes an N-layer DBN whereas the visible input vector was x and *the* l^{th} layer of a hidden vector can be h^l $(l = 1, 2, \ldots, N)$; afterwards, the joint probability distribution for DBN takes the subsequent procedure

$$P\left(x,\ h^1, \ldots,\ h^N\right) = \left(\prod_{l=1}^{N-1} P\left(h^{l-1} | h^l\right)\right) P\left(h^{N-1},\ h^N\right) \tag{1}$$

whereas $x \triangleq h^o$, $P\left(h^{N-1},\ h^N\right)$ refers to joint distribution determined as top RBM, and $\prod_{l=1}^{N-1} P(h^{l-1}|h^l)$ implies the distribution of directed sigmoid belief network under.

And the conditional probability is

$$\begin{aligned} &P(h^{l-1}|h^l) = sigm\left(b^l + h^l W^l\right) \\ &P\left(h^l|h^{l-1}\right) = sigm\left(c^l + h^{l-1} W^{l^T}\right) \\ &P\left(h^{N-1},\ h^N\right) = \frac{1}{\sum_{h^{N-1}, h^N} e^{-E\left(h^{N-1}, h^N\right)}} e^{-E\left(h^{N-1}, h^N\right)} \end{aligned} \tag{2}$$

In which $E\left(h^{N-1},\ h^N\right)$ denotes the energy function of RBM on the top, for instance,

$$E\left(h^{N-1},\ h^N\right) = -h^{N-1} W^{N^T} h^{N^T} - h^{N-1} b^{N^T} - h^N c^{N^T}$$

and W^l refers to the weighted matrix, b^l and c^l stand for the visible and hidden bias vectors for l^{th} RBM correspondingly $(l = 1, 2, \ldots, N)$.

It can be complex to perform Gibbs sampling by conditional distribution $P(.)$ in Eq. (2) as it cannot be factorized. Thus, it can generally utilize the nearby posteriors represented by $Q(.)$ for sampling and model inference, viz., distribution of l^{th} RBM can be denoted by $Q\left(h^{l-1},\ h^{l}\right)$ in the pre-training phase and $Q(h^l|h^{l-1})$ was utilized for performing bottom-up inferences. It is noticeable only the posterior $Q(h^N|h^{N-1})$ is the same as true probability $P(h^N|h^{N-1})$ for top RBM, but the residual of $Q(.)$ is every estimate.

Algorithm 1: (Pre-training) a DBN from the layer-wise manner

Input: Training data $h^0 = x$; initializing $\theta^l = \left(W^l,\ b^l,\ c^l\right) = 0, l = 1, 2, \ldots,\ N$; rate of learning $\in$;.
Output: A DBN with N layers.
for $l = 1$ to N, do
training l the RBM with data h^{l-1} *by CD* approach;
develop the well-learned parameters W, b^l and c^l;
sample $h^l \sim Q(h^l|h^{l-1};\theta^l) = P(h^l|h^{l-1};\theta^l)$ by Eq. (2);
end for.

2.1.2 GRU Model

Like the LSTM model, GRU is intended for adoptively updating or resetting the memory contents with z^j and RJ reset and update gates like input and forget gates of LSTM: In contrast to LSTM, GRU has only 2 gates and doesn't have memory cells. The GRU activation h_t^j at time t is the linear interruption of candidate activation $\tilde{h}$ and preceding activation h_{t-1}^j.

To evaluate the state h_t^j of *jth* GRU at *the t* time step, we employ the following formula:

$$h_t^j = \left(1 - z_t^j\right) h_{t-1}^j + z_t^j \tilde{h}_t \tag{3}$$

In Eq. (3), $\tilde{h}_t^j$ and h_{t-1}^j correspond to the original candidate and preceding memory contents. z_t^j signifies the update gate, which permits the module to determine the quantity of the preceding dataset (from the preceding time step) conveyed to the upcoming and the quantity of original memory content to be included.

To evaluate the update gate z_t for the t time step, we employ the preceding hidden state h_{t-1} and the existing input tex_t as follows:

$$z_t = \sigma\left(W_z x_t + U_z h_{t-1}\right) \tag{4}$$

The new memory content $\tilde{h}_t^j$ is evaluated by using Eq. (5):

$$\tilde{h}_t = \tanh\left(W x_t + r_t \odot U h_{t-1}\right) \tag{5}$$

Now $\odot$ represents the Hadamard product, and r_t signifies the reset gates that are utilized for determining the quantity of dataset be forgotten from the initial state and then apply the following equation for computation:

$$r_t = \sigma\left(W_r x_t + U_r h_{t-1}\right) \tag{6}$$

GRU is fast than LSTM in training because GRU has a simple structure using lesser parameters and thus employs lesser memory. Fig. 2 depicts the infrastructure of the GRU approach.

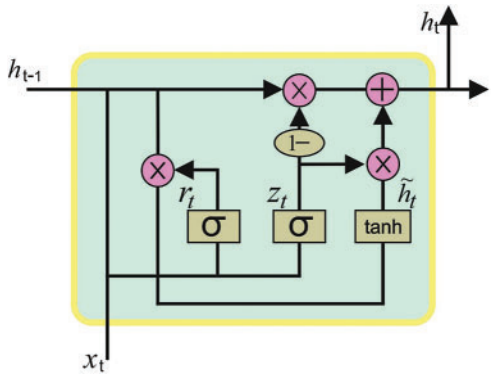

Figure 2: Structure of GRU

2.1.3 PNN Model

Consider an input vector $x \in \mathbb{R}^n$ belongs to the predetermined class $g = 1, 2, \ldots, G$. Here, the probability of vector x belonging to the class g equals p_g, the cost related to categorizing the vector into class g and that the probability density function: $y_1(x)$, $y_2(x)$, …, $y(x)$ for each class is known [22]. Next, based on the Bayes theorem, if $g \neq h$, the vector x was categorized into the class g when $p_g c_g y_g(x) > p_h c_h y_h(x)$. Generally, $p_g = p_h$ and $c_g = c_h$; therefore, one could infer that when $y_g(x) > y_h(x)$, then the vector x belongs to the class g.

In real-time data classification problems, dataset distribution is generally unknown, and a calculation of probability density function $y_g(x)$ should be defined. It is accomplished by the Parzen methodology, where the probability density function for more than one parameter is formulated by using Eq. (7):

$$y(x) = \frac{1}{l}\sum_{i=1}^{l} W_i(x,\ x_i) \tag{7}$$

Now, $W_i(x,\ x_i) = \sigma_1^{-1} \ldots \sigma_n^{-1} F\left(\sigma_1^{-1}(x_{i1} - x_1),\ \ldots,\ \sigma_n^{-1}(x_{in} - x_n)\right)$, $F(.)$ represents the weighting function that should be properly chosen [19], l indicates the input pattern count, and $\sigma_1, \ldots, \sigma_n$ denotes standard deviation related to the mean of n parameters $x_1, \ldots, x_n$. Generally, the Gaussian function is a collective decision for weighting.

Eq. (7) describes the architecture and the process of PNN. Then, assume a Gaussian function as an activation for the probability density function and consider that this function can be evaluated for class g as follows:

$$y_g(x) = \frac{1}{l_g (2\pi)^{n/2} (\det \Sigma_g)^{1/2}} \sum_{i=1}^{lg} \exp\left(-\frac{1}{2}(x_{g,i} - x)^T \Sigma_g^{-1}(x_{g,i} - x)\right) \tag{8}$$

In Eq. (8), $\Sigma_g = \text{diag}\left(\sigma_{g,1}^2,\ \ldots,\ \sigma_{g,n}^2\right)$ refers to the covariance matrix, LG denotes the example count of class g, $\sigma_{g,j}$ denotes the smoothing variable related to *aj-th* parameter and *g-th* e classes, and $x_{g,i}$ indicates the *i-th* training vectors ($i = 1,\ \ldots,\ LG$) from the g class. The equation given in (8) offers $g = 1, \ldots, G$ summation neuron of PNN architecture. The components of the previous layer, the pattern neuron, feed the component to the sum evaluated through every example of *g-th* e class. Then, LG hidden neuron constitutes the input for *g-th* e summation neuron. At last, the output layer defines output for vector x according to Bayes's decision rule.

$$G^*(x) = \arg\ \max_g \left\{y_g(x)\right\} \tag{9}$$

In Eq. (9), $G^*(x)$ signifies the expected class for pattern x. Therefore, the pattern layer needs $l = l_1 + \ldots + l_G$ nodes.

Here, a single smoothing variable for every class and attribute is employed. The selection of variation of σ choice imposes, based on Eq. (8), the predictability of saving a $G \times n$ matrixes of σ's. Therefore, the g-th summation neuron produces to decision layer of the output signal (8), however, with $\sigma_{g,j}$ as an intrinsic variable. Thus, the smoothing variable is evaluated for the j-th parameter of every g class. This technique allows for highlighting the similarity of vectors belonging to a similar class. The conjugate gradient methodology defines the value of σ.

2.2 Hyperparameter Optimization

For optimal hyperparameter tuning of the DL models, the CBSO algorithm is exploited in this work. The BSO algorithm is a robust optimization procedure with the features of the simplest technique, better expandability, etc. Deliberate N virtual birds fly and forage for food [23]. Supposing x_i^t ($i \in [1, 2, \cdots, N]$) expresses the position of an ith bird at t. The bird behaviour is defined below:

1. Foraging behaviour can be defined by Eq. (10):

$$x_{i,j}^{t+1} j = x_{i,j}^t + \left(p_{i,j} - x_{i,j}^t\right) \times C \times rand\,(0,1) + \left(g_{i,j} - x_{i,j}^t\right) \times S \times rand\,(0,1) \tag{10}$$

2. Vigilance behaviour can be defined by Eq. (11):

$$x_{i,j}^{t+1} = x_{i,j}^t + A_1 \left(mean_j - x_{i,j}^t\right) \times rand\,(0,1) + A_2 \left(p_{i,j} - x_{i,j}^t\right) \times rand\,(-1,1) \tag{11}$$

which A_1 and A_2 are arithmetically defined below:

$$A_1 = a_1 \times \exp\left(-\frac{p_{Fit_i}}{sumFit + \varepsilon} \times N\right)$$

$$A_2 = a_2 \times \exp\left(\left(\frac{p_{Fit_i} - p_{Fit_k}}{\left|p_{Fit_k} - p_{Fit_i}\right| + \varepsilon}\right) \times \frac{N \times p_{Fit_k}}{sumFit + \varepsilon}\right)$$

a_1 and a_2 denote constants in $[0, 2]$. ε refers to a small constant.

3. Flight behaviour is defined below:

$$x_{i,j}^{t+1} = x_{i,j}^t + randn\,(0,1) \times x_{i,j}^t \tag{12}$$

$$x_{i,j}^{t+1} = x_{i,j}^t + \left(x_{k,j}^t - x_{i,j}^t\right) \times FL \times randn\,(0,1) \tag{13}$$

Now, FL lies within $[0, 2]$. The chaotic method has a sensitive property to primary conditions. The chaotic signal produced using the determinist system has the superiority of genus-randomness. The curve is defined using the primary values and chaos mapping parameter. In the CBSO algorithm, logistic mapping is widely applied. The Logistic chaotic scheme has complicated dynamic behavior and is defined below:

$$\lambda_{i+1} = \mu \times \lambda_i \times (1 - \lambda_i) \tag{14}$$

$\lambda \in [0, 1]$, $i = 0, 1, 2, \cdots$, μ is in [1,4]. The study suggested that μ is near 4 and λ near the average distribution within [0, 1]. While the scheme is chaotic if μ is 4. The early population is a noteworthy fragment in the intelligent optimization approach that impacts the convergence speed and the final solution quality. Here, Logistic chaotic mapping is applied for population initialization that exploits the solution space to improve the model's accuracy.

3 Results and Discussion

In this section, the Arabic poetry classification outcomes of the CBSOEDL-APCD model are tested using an Arabic poetry dataset comprising 34000 samples under 17 class labels, as depicted in Table 1.

Table 1: Dataset details

Class	Description	No. of instances
1	Tawil	2000
2	Kamil	2000
3	Basit	2000
4	Khafif	2000
5	Wafer	2000
6	Rajaz	2000
7	Ramal	2000
8	Mutaqarib	2000
9	Sari	2000
10	Munsarih	2000
11	Mujtathth	2000
12	Madrid	2000
13	Hazaj	2000
14	Mutadarik	2000
15	Muqtadab	2000
16	Mudari	2000
17	Prose	2000
Total number of instances		**34000**

Fig. 3 reports the confusion matrix generated by the CBSOEDL-APCD model on the entire dataset. The figure indicated that the CBSOEDL-APCD model had recognized 1887 samples into class 1, 1928 samples into class 2, 1911 samples into class 3, 1885 samples into class 4, 1917 samples into class 5, 1893 samples into class 6, 1911 samples into class 7, and so on.

A detailed poetry classification outcome of the CBSOEDL-APCD model under the entire dataset is represented in Table 2 and Fig. 4. The results reported that the CBSOEDL-APCD model had enhanced outcomes under all class labels. For instance, the CBSOEDL-APCD model has identified class 1 samples with *an* $accu_y$ of 99.39%, $sens_y$ of 94.35%, $spec_y$ of 99.71%, F_{-score} of 94.82%, and AUC_{score} of 97.03%. Also, the CBSOEDL-APCD algorithm has identified class 2 samples with $accu_y$ of 99.59%, $sens_y$ of 96.40%, $spec_y$ of 99.78%, F_{score} of 96.47%, and AUC_{score} of 98.09%. In addition, the CBSOEDL-APCD algorithm has identified class 3 samples with $accu_y$ of 99.48%, $sens_y$ of 95.55%, $spec_y$ of 99.73%, F_{-score} of 95.60%, and AUC_{score} of 97.64%. Then, the CBSOEDL-APCD technique identified class 4 samples with an $accu_y$ of 99.39%, $sens_y$ of 94.25%, $spec_y$ of 99.71%, F_{score} of 94.80%, and AUC_{score} of 96.98%.

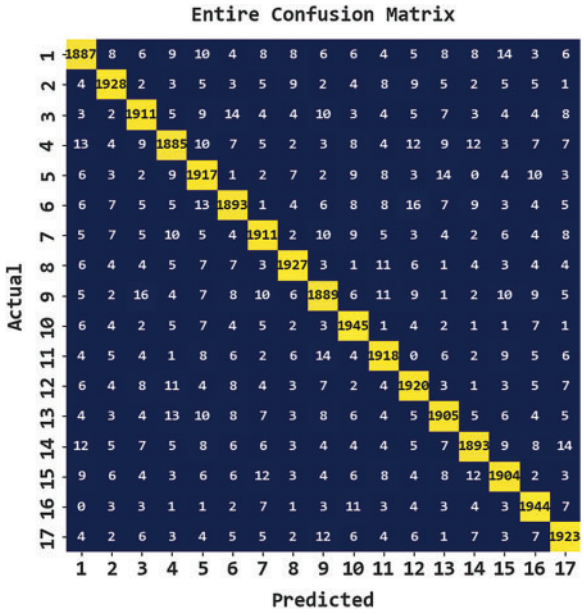

Figure 3: Confusion matrix of CBSOEDL-APCD approach under the entire dataset

Table 2: Result analysis of the CBSOEDL-APCD approach with distinct measures under the entire dataset

Entire dataset					
Labels	Accuracy	Sensitivity	Specificity	F-Score	AUC Score
1	99.39	94.35	99.71	94.82	97.03
2	99.59	96.40	99.78	96.47	98.09
3	99.48	95.55	99.73	95.60	97.64
4	99.39	94.25	99.71	94.80	96.98
5	99.42	95.85	99.64	95.11	97.75
6	99.41	94.65	99.71	94.98	97.18
7	99.49	95.55	99.73	95.62	97.64
8	99.59	96.35	99.80	96.54	98.07
9	99.39	94.45	99.70	94.78	97.07
10	99.56	97.25	99.71	96.33	98.48
11	99.49	95.90	99.72	95.68	97.81
12	99.48	96.00	99.70	95.62	97.85
13	99.47	95.25	99.73	95.46	97.49
14	99.47	94.65	99.77	95.44	97.21
15	99.46	95.20	99.73	95.44	97.47
16	99.58	97.20	99.72	96.43	98.46
17	99.51	96.15	99.72	95.84	97.93
Average	**99.48**	**95.59**	**99.72**	**95.59**	**97.66**

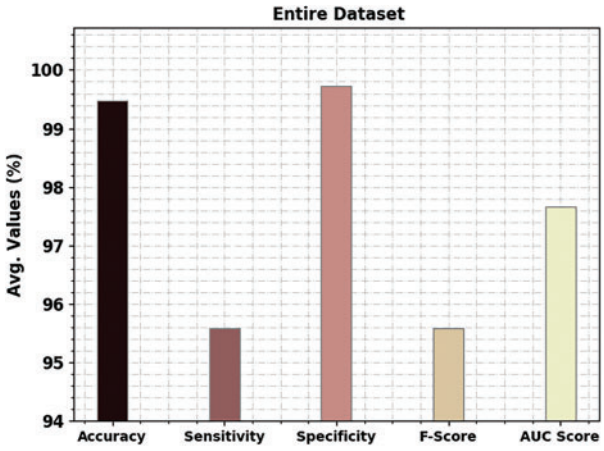

Figure 4: Average analysis of the CBSOEDL-APCD approach under the entire dataset

Fig. 5 portrays the confusion matrix generated by the CBSOEDL-APCD algorithm on 70% of training (TR) data. The figure represented the CBSOEDL-APCD approach has recognized 1272 samples into class 1, 1341 samples into class 2, 1336 samples into class 3, 1346 samples into class 4, 1306 samples into class 5, 1324 samples into class 6, 1392 samples into class 7, and so on.

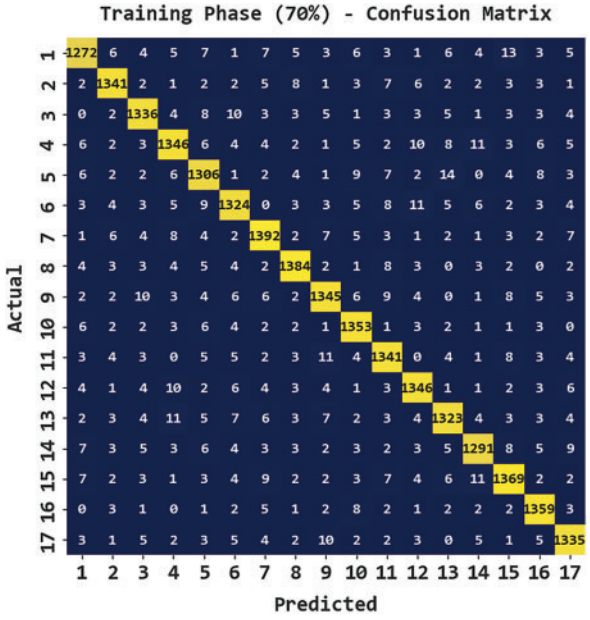

Figure 5: Confusion matrix of CBSOEDL-APCD approach under 70% of TR data

A brief poetry classification outcome of the CBSOEDL-APCD approach under 70% of training (TR) is represented in Table 3 and Fig. 6. The results reported that the CBSOEDL-APCD algorithm had exhibited enhanced outcomes under all class labels. For example, the CBSOEDL-APCD approach has identified class 1 samples with *an accu*$_y$ of 99.43%, *sens*$_y$ of 94.15%, *spec*$_y$ of 99.75%, F_{-score} of 94.96%, and *an* AUC_{score} of 96.95%. Further, the CBSOEDL-APCD technique has identified class 2 samples with *accu*$_y$ of 99.60%, *sens*$_y$ of 96.41%, *spec*$_y$ of 99.79%, F_{score} of 96.54%, and AUC_{score} of 98.10%.

Additionally, the CBSOEDL-APCD approach has identified class 3 samples with *an* $accu_y$ of 99.51%, $sens_y$ of 95.84%, $spec_y$ of 99.74%, F_{score} of 95.84%, and AUC_{score} of 97.79%. Then, the CBSOEDL-APCD algorithm identified class 4 samples with an $accu_y$ of 99.39%, $sens_y$ of 94.52%, $spec_y$ of 99.71%, F_{score} of 94.92%, and AUC_{score} of 97.11%.

Table 3: Result analysis of CBSOEDL-APCD approach with distinct measures under 70% of TR data

Training phase (70%)					
Labels	Accuracy	Sensitivity	Specificity	F-Score	AUC Score
1	99.43	94.15	99.75	94.96	96.95
2	99.60	96.41	99.79	96.54	98.10
3	99.51	95.84	99.74	95.84	97.79
4	99.39	94.52	99.71	94.92	97.11
5	99.38	94.84	99.66	94.67	97.25
6	99.41	94.71	99.70	94.94	97.20
7	99.49	96.00	99.71	95.80	97.86
8	99.61	96.78	99.79	96.72	98.28
9	99.44	94.99	99.72	95.29	97.35
10	99.57	97.20	99.71	96.33	98.46
11	99.45	95.72	99.69	95.38	97.70
12	99.52	96.07	99.74	95.94	97.91
13	99.44	94.91	99.72	95.21	97.32
14	99.47	94.79	99.76	95.38	97.27
15	99.44	95.27	99.70	95.33	97.49
16	99.61	97.49	99.75	96.73	98.62
17	99.52	96.18	99.72	95.87	97.95
Average	**99.49**	**95.64**	**99.73**	**95.64**	**97.68**

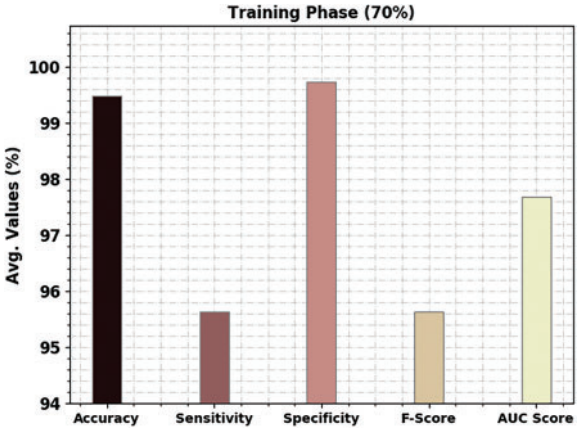

Figure 6: Average analysis of CBSOEDL-APCD approach under 70% of TR data

Fig. 7 presents the confusion matrix generated by the CBSOEDL-APCD algorithm on 30% of testing (TS) data. The figure denoted the CBSOEDL-APCD approach has recognized 615 samples into class 1, 587 samples into class 2, 575 samples into class 3, 539 samples into class 4, 611 samples into class 5, 569 samples into class 6, 519 samples into class 7, and so on.

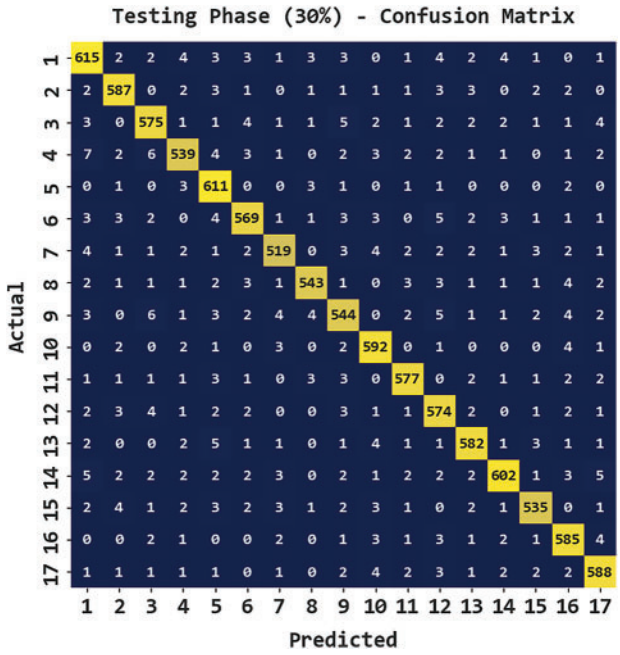

Figure 7: Confusion matrix of CBSOEDL-APCD approach under 30% of TS data

A brief poetry classification outcome of the CBSOEDL-APCD approach under 30% of TS is depicted in Table 4 and Fig. 8. The results reported that the CBSOEDL-APCD approach had outperformed enhanced outcomes under all class labels. For example, the CBSOEDL-APCD technique has identified class 1 samples with an $accu_y$ of 99.30%, $sens_y$ of 94.76%, $spec_y$ of 99.81%, F_{score} of 94.54%, and AUC_{score} of 97.19%. Additionally, the CBSOEDL-APCD approach has identified class 2 samples with $accu_y$ of 99.56%, $sens_y$ of 96.39%, $spec_y$ of 99.76%, F_{score} of 96.31%, an AUC_{score} of 98.07%. The CBSOEDL-APCD technique has also identified class 3 samples with $accu_y$ of 99.41%, $sens_y$ of 94.88%, $spec_y$ of 99.70%, F_{-score} of 95.04%, and AUC_{score} of 97.29%. Then, the CBSOEDL-APCD approach identified class 4 samples with $accu_y$ of 99.38%, $sens_y$ of 93.58%, $spec_y$ of 99.73%, F_{score} of 94.48%, and AUC_{score} of 96.65%.

Table 4: Result analysis of CBSOEDL-APCD approach with distinct measures under 30% of TS data

	Testing phase (30%)				
Labels	Accuracy	Sensitivity	Specificity	F-Score	AUC Score
1	99.30	94.76	99.61	94.54	97.19
2	99.56	96.39	99.76	96.31	98.07
3	99.41	94.88	99.70	95.04	97.29
4	99.38	93.58	99.73	94.48	96.65

(Continued)

Table 4 (continued)

	Testing phase (30%)				
Labels	Accuracy	Sensitivity	Specificity	F-Score	AUC Score
5	99.51	98.07	99.60	96.07	98.84
6	99.42	94.52	99.73	95.07	97.12
7	99.48	94.36	99.77	95.14	97.07
8	99.57	95.26	99.82	96.11	97.54
9	99.26	93.15	99.64	93.55	96.39
10	99.56	97.37	99.70	96.34	98.53
11	99.58	96.33	99.78	96.41	98.05
12	99.39	95.83	99.61	94.88	97.72
13	99.53	96.04	99.75	96.04	97.89
14	99.45	94.36	99.79	95.56	97.07
15	99.53	95.03	99.79	95.71	97.41
16	99.49	96.53	99.68	95.74	98.11
17	99.49	96.08	99.71	95.77	97.89
Average	**99.47**	**95.44**	**99.72**	**95.46**	**97.58**

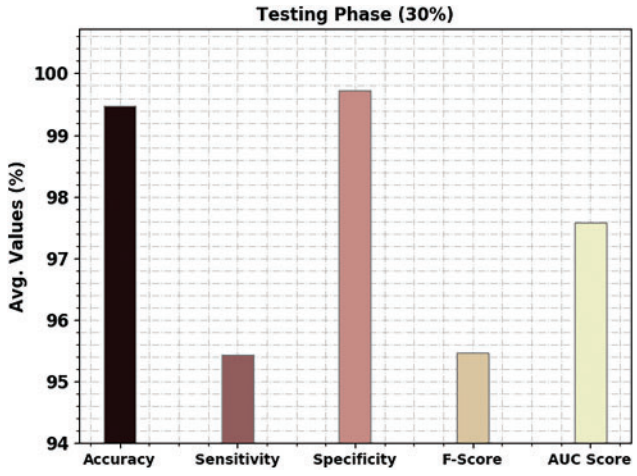

Figure 8: Average analysis of CBSOEDL-APCD approach under 30% of TS data

The training accuracy (TRA) and validation accuracy (VLA) gained by the CBSOEDL-APCD method on the test dataset is shown in Fig. 9. The experimental outcome represented the CBSOEDL-APCD algorithm has attained maximum values of TRA and VLA. In Particular, the VLA is greater than TRA.

The training loss (TRL) and validation loss (VLL) gained by the CBSOEDL-APCD algorithm on the test dataset were exhibited in Fig. 10. The experimental outcome indicates the CBSOEDL-APCD approach has accomplished least values of TRL and VLL. Seemingly, the VLL is lesser than TRL. A clear precision-recall analysis of the CBSOEDL-APCD methodology on the test dataset is given in Fig. 11. The figure specified that the CBSOEDL-APCD algorithm has resulted in enhanced

values of precision-recall values under all classes. A detailed ROC analysis of the CBSOEDL-APCD technique on the test dataset is portrayed in Fig. 12. The results implicit that the CBSOEDL-APCD algorithm has outperformed its ability to categorise distinct classes on the test dataset. For assuring the enhancements of the CBSOEDL-APCD model, a comparative $accu_y$ examination is made in Table 5 [24]. The experimental values indicated that the expert system and bidirectional long short term memory (BiLSTM) models had reduced $accu_y$ of 96.43% and 96.27%, respectively.

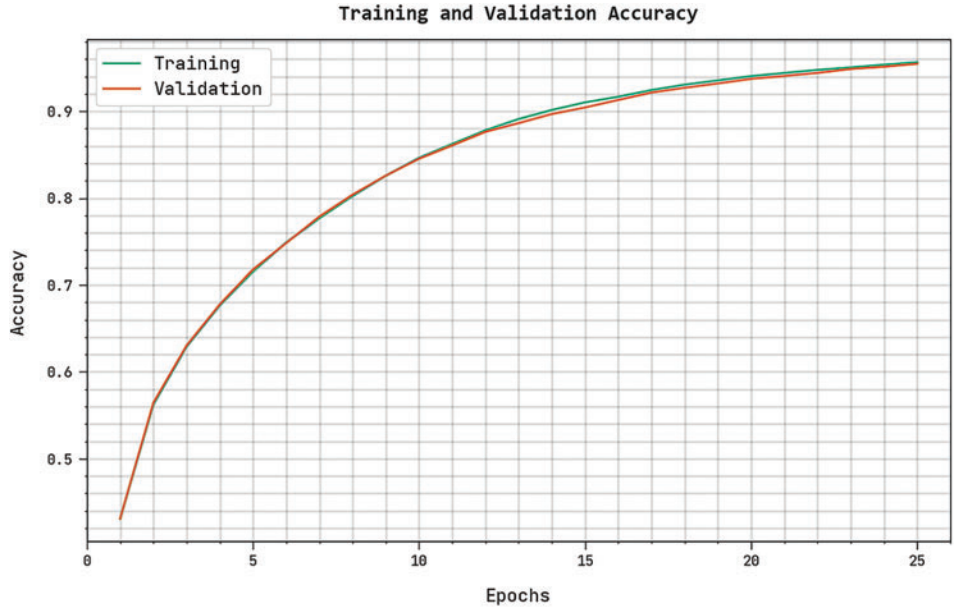

Figure 9: TRA and VLA analysis of the CBSOEDL-APCD approach

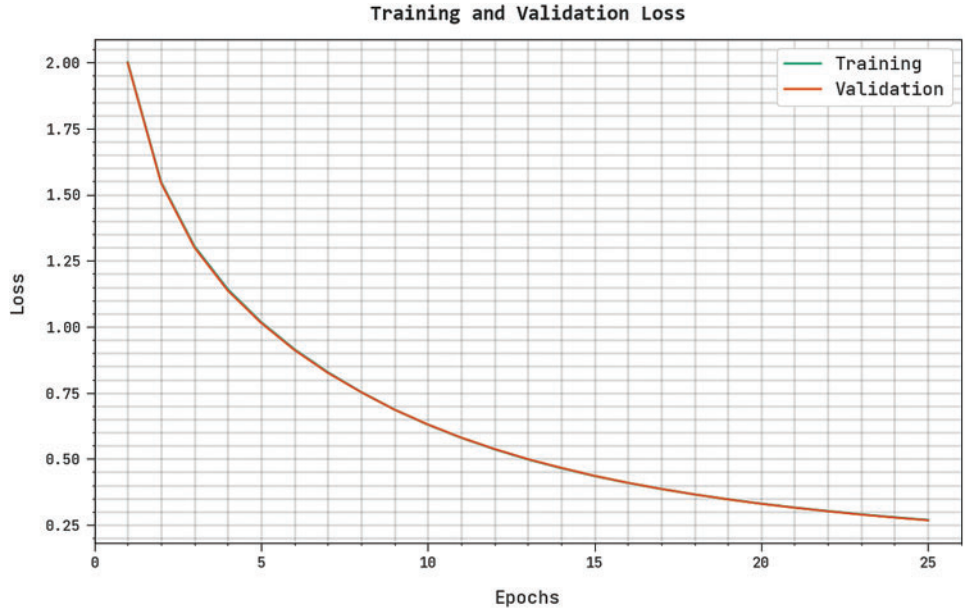

Figure 10: TRL and VLL analysis of the CBSOEDL-APCD approach

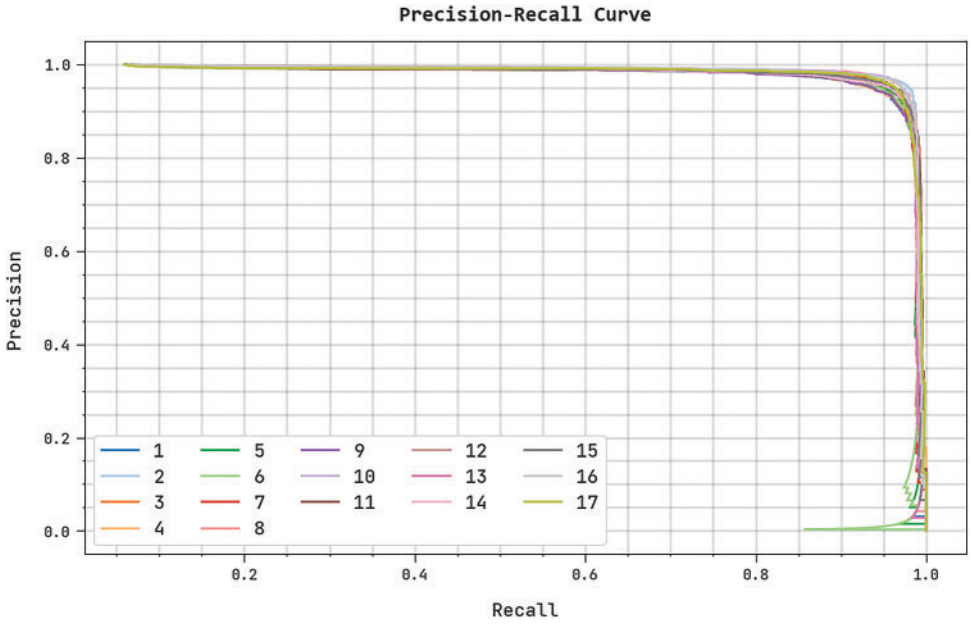

Figure 11: Precision-recall analysis of the CBSOEDL-APCD approach

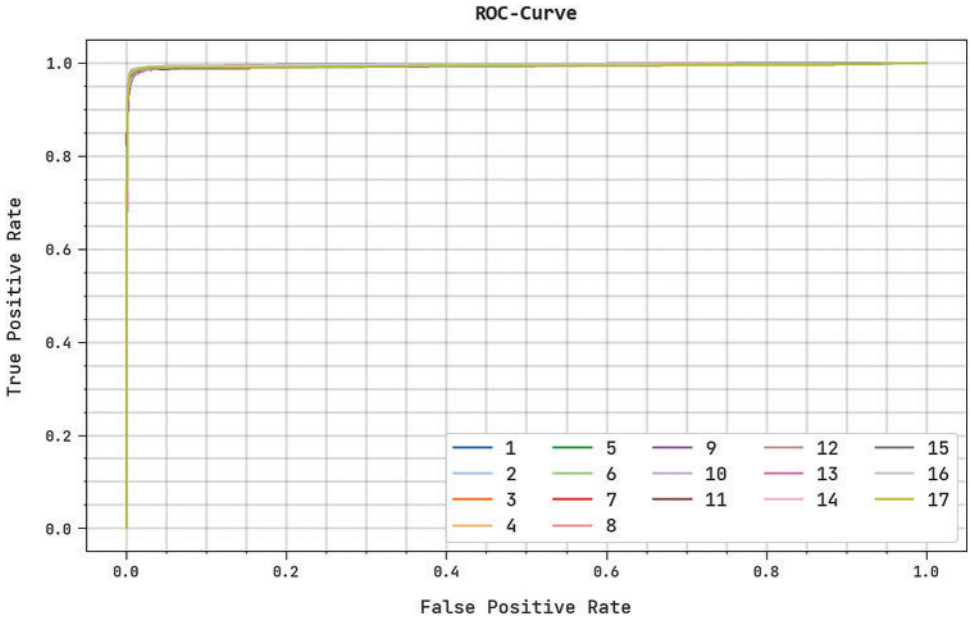

Figure 12: ROC analysis of the CBSOEDL-APCD approach

Table 5: Comparative analysis of the CBSOEDL-APCD approach with recent algorithms

Methods	Accuracy
CBSOEDL-APCD	99.48
Expert system	96.43

(Continued)

Table 5 (continued)

Methods	Accuracy
Context-free grammar	97.59
Rule-based algorithm	98.38
BiGRU	97.14
BiLSTM	96.27

Next, the bidirectional gated recurrent unit (BiGRU) and Context free grammar models have obtained slightly enhanced $accu_y$ of 97.14% and 97.59%, respectively. However, the CBSOEDL-APCD model has showcased a higher $accu_y$ of 99.48%. These values reassured that the CBSOEDL-APCD model had obtained effectual SA outcomes.

4 Conclusion

This study introduced a new CBSOEDL-APCD algorithm for classifying and dictarization of Arabic text into Arabic poetries and prose. Primarily, the CBSOEDL-APCD technique carries out data pre-processing to convert it into a useful format. Besides, the EDL model encompasses DBN, GRU, and PNN models. At last, the CBSO algorithm is employed for the optimal hyperparameter tuning of the DL models to enhance the overall classification performance. An extensive range of experiments was performed to demonstrate the enhanced outcomes of the CBSOEDL-APCD technique. A wide-ranging experimental analysis indicates the better outcomes of the CBSOEDL-APCD approach over other recent ones. Thus, the CBSOEDL-APCD technique can be employed to classify Arabic poems.

Acknowledgement: The authors express their gratitude to Princess Nourah bint Abdulrahman University Researchers Supporting Project, Princess Nourah bint Abdulrahman University Riyadh Saudi Arabia.

Funding Statement: Princess Nourah bint Abdulrahman University Researchers Supporting Project Number (PNURSP2022R281), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. The authors thank the Deanship of Scientific Research at Umm Al-Qura University for supporting this work by Grant Code: 22UQU4310373DSR42.

Author Contributions: Conceptualization, Hala J. Alshahrani; Methodology, Hala J. Alshahrani; Najm Alotaibi, Software, Mohamed K. Nour; Validation, Mahmoud Othman and Manar Ahmed Hamza Investigation, Hala J. Alshahrani; Data curation, Gouse Pasha Mohammed; Writing – original draft, Manar Ahmed Hamza, Hala J. Alshahrani, Najm Alotaibi, Mohamed K. Nour, Mahmoud Othman, Gouse Pasha Mohammed, Mohammed Rizwanullah and Mohamed I. Eldesouki; Writing – review & editing, Manar Ahmed Hamza, Mahmoud Othman, Gouse Pasha Mohammed, Mohammed Rizwanullah; Visualization, Mohamed I. Eldesouki; Project administration, Manar Ahmed Hamza; Funding acquisition, Hala J. Alshahrani. All authors have read and agreed to the published version of the manuscript.

Availability of Data and Materials: Data sharing not applicable to this article as no datasets were generated during the current study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Masmoudi, C. Aloulou, A. Abdellahi and L. Belguith, "Automatic discretization of Tunisian dialect text using SMT model," *International Journal of Speech Technology*, vol. 25, no. 1, pp. 89–104, 2021.

[2] M. A. A. Rashwan, A. A. Al Sallab, H. M. Raafat and A. Rafea, "Deep learning framework with confused sub-set resolution architecture for automatic Arabic discretization," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 505–516, 2015.

[3] G. Abandah and A. Karim, "Accurate and fast recurrent neural network solution for the automatic discretization of Arabic text," *Jordanian Journal of Computers and Information Technology*, vol. 6, no. 2, pp. 103–121, 2020.

[4] F. N. Al-Wesabi, "A hybrid intelligent approach for content authentication and tampering detection of Arabic text transmitted via internet," *Computers, Materials & Continua*, vol. 66, no. 1, pp. 195–211, 2021.

[5] O. Hamed, "Automatic discretization as prerequisite towards the automatic generation of Arabic lexical recognition tests," in *Proc. of the 3rd Int. Conf. on Natural Language and Speech Processing*, Trento, Italy, pp. 100–106, 2019.

[6] F. N. Al-Wesabi, "Proposing high-smart approach for content authentication and tampering detection of Arabic text transmitted via internet," *IEICE Transactions on Information and Systems*, vol. E103.D, no. 10, pp. 2104–2112, 2020.

[7] M. S. Al-shaibani, Z. Alyafeai and I. Ahmad, "Meter classification of Arabic poems using deep bidirectional recurrent neural networks," *Pattern Recognition Letters*, vol. 136, pp. 1–7, 2020.

[8] F. N. Al-Wesabi, "A smart English text zero-watermarking approach based on third-level order and word mechanism of markov model," *Computers, Materials & Continua*, vol. 65, no. 2, pp. 1137–1156, 2020.

[9] A. Fadel, I. Tuffaha, B. Al-Jawarneh and M. Al-Ayyoub, "Arabic text discretization using deep neural networks," in *2nd Int. Conf. on Computer Applications & Information Security (ICCAIS)*, Riyadh, Saudi Arabia, pp. 1–7, 2019.

[10] F. N. Al-Wesabi, "Entropy-based watermarking approach for sensitive tamper detection of Arabic text," *Computers, Materials & Continua*, vol. 67, no. 3, pp. 3635–3648, 2021.

[11] G. A. Abandah, A. Graves, B. Al-Shagoor, A. Arabiyat, F. Jamour *et al.,* "Automatic discretization of Arabic text using recurrent neural networks," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 18, no. 2, pp. 183–197, 2015.

[12] A. A. Karim and G. Abandah, "On the training of deep neural networks for automatic Arabic-text discretization," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 8, pp. 276–286, 2021.

[13] F. N. Al-Wesabi, A. Abdelmaboud, A. A. Zain, M. M. Almazah and A. Zahary, "Tampering detection approach of Arabic-text based on contents interrelationship," *Intelligent Automation & Soft Computing*, vol. 27, no. 2, pp. 483–498, 2021.

[14] A. Shahrour, S. Khalifa and N. Habash, "Improving Arabic discretization through syntactic analysis," in *Proc. of the Conf. on Empirical Methods in Natural Language Processing*, Lisbon, Portu, pp. 1309–1315, 2015.

[15] I. H. Ali, Z. Mnasri and Z. Laachri, "Gemination prediction using DNN for Arabic text-to-speech synthesis," in *16th Int. Multi-Conf. on Systems, Signals & Devices (SSD)*, Istanbul, Turkey, pp. 366–370, 2019.

[16] W. Almanaseer, M. Alshraideh and O. Alkadi, "A deep belief network classification approach for automatic discretization of Arabic text," *Applied Sciences*, vol. 11, no. 11, pp. 5228, 2021.

[17] M. A. H. Madhfar and A. M. Qamar, "Effective deep learning models for automatic discretization of Arabic text," *IEEE Access*, vol. 9, pp. 273–288, 2021.

[18] A. Fadel, I. Tuffaha and M. Al-Ayyoub, "Neural Arabic text discretization: State-of-the-art results and a novel approach for Arabic nlp downstream tasks," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 21, no. 1, pp. 1–25, 2022.
[19] A. Alharbi, M. Taleb and M. Kalkatawi, "Deep learning in Arabic sentiment analysis: An overview," *Journal of Information Science*, vol. 47, no. 1, pp. 129–140, 2019.
[20] R. Moumen, R. Chiheb, R. Faizi and A. El, "Evaluation of gated recurrent unit in Arabic discretization," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 11, pp. 360–3664, 2018.
[21] W. Deng, H. Liu, J. Xu, H. Zhao and Y. Song, "An improved quantum-inspired differential evolution algorithm for deep belief network," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 10, pp. 7319–7327, 2020.
[22] G. Capizzi, G. L. Sciuto, C. Napoli, D. Polap and M. Wozniak, "Small lung nodules detection based on fuzzy-logic and probabilistic neural network with bioinspired reinforcement learning," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 6, pp. 1178–1189, 2020.
[23] K. Mishra and S. K. Majhi, "A binary bird swarm optimization based load balancing algorithm for cloud computing environment," *Open Computer Science*, vol. 11, no. 1, pp. 146–160, 2021.
[24] G. A. Abandah, M. Z. Khedher, M. R. A. Majeed, H. M. Mansour, S. F. Hulliel *et al.,* "Classifying and diacritizing Arabic poems using deep recurrent neural networks," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 6, pp. 3775–3788, 2022.