VISTA: A Variable Length Genetic Algorithm and LSTM-Based Surrogate Assisted Ensemble Selection algorithm in Multiple Layers Ensemble System

Kate Han
Salford Business School, University of Salford
Manchester, UK

Truong Thanh Nguyen
School of Computing, Robert Gordon University
Aberdeen, UK

Viet Anh Vu
School of ICT, Hanoi University of Science and Technology
Hanoi, Vietnam

Alan Wee-Chung Liew
School of ICT, Griffith University
Gold Coast, Australia

Truong Dang, Tien Thanh Nguyen
National Subsea Center, Robert Gordon University
Aberdeen, UK

2024 IEEE Congress on Evolutionary Computation (CEC) | 979-8-3503-0836-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/CEC60901.2024.10612029

***Abstract*—We proposed a novel ensemble selection method called VISTA for multiple layers ensemble systems (MLES). Our ensemble model consists of multiple layers of ensemble of classifiers (EoC) in which the EoC in each layer is trained on the data generated by a concatenation of the original training data and the predictions by classifiers of the previous layer. The predictions of the EoC in the final layer are aggregated to obtain the final prediction. To enhance the accuracy of the MLES, we used the Variable-Length Genetic Algorithm (VLGA) to search for the optimal configuration of EoC in each layer. Since the optimisation process is computationally intensive, we use Surrogate-Assisted Evolutionary Algorithms (SAEA) to reduce the training time. Most surrogate models developed in the literature require a fixed-length input, which limits their applications when the encoding is of variable length. In this paper, we proposed to use a Long Short-Term Memory (LSTM)-based surrogate model, in which the LSTM transforms the variable-length encoding to a fixed-size representation which will then be used by the surrogate model to predict the fitness values in VLGA. For the surrogate model, we adopted Radial Basis Function (RBF) for surrogation. We first conducted experiments in comparing two types of LSTM converters, and the results suggest that the proposed chunk-based LSTM converter provides better results compared to the normal LSTM converter. Our experiments on 15 datasets show that VISTA outperforms several benchmark algorithms.**

***Index Terms*—Ensemble learning, ensemble selection, classifier selection, ensemble of classifiers, Surrogate model**

I. Introduction

In recent years, ensemble learning, a sub-field of machine learning which aims to aggregate multiple classifiers to improve prediction accuracy, has been successfully applied to many areas. By combining the prediction results of multiple single classifiers, an ensemble learning system can compensate for poor predictions of individual classifiers to achieve improved overall performance. An example which demonstrates the effectiveness of ensemble learning is shown in [1] in which the authors applied 179 machine learning methods on 121 datasets to show that ensemble-based methods perform the best.

In 2014, Zhou et a. proposed a deep ensemble system named gcForest which consists of multiple layers of Random Forest and Completely Random Tree Forest [2]. That study showed that gcForest is a subset of deep models, which can be constructed using non differentiable modules. Experimental results on several popular datasets showed that gcForest outperforms both DNNs and state-of-the-art ensemble methods [2] [3]. With the success of gcForest in general, in recent years there have been multiple works on deep and MLES, and most works so far have focused on selecting the best ensemble for each layer. Since MLES are composed of non-differentiable modules, optimisation methods such as Evolutionary Algorithms (EA) are usually used to find the optimal ensemble configuration for each layer. However, it is noted that optimising a MLES requires more time compared to a single layer ensemble [4]. A potential solution to reduce computation time is by using surrogate-assisted EA (SAEA) [5], which uses a function with lower computational overhead, called surrogate function, to predict the fitness value during the evolutionary process. However, it is noted that the surrogate functions assume a fixed-size input, while the optimal number of layers in a multiple layers ensemble is usually not known beforehand [3] [5].

In this paper, we propose a novel ensemble selection method to improve the performance and efficiency of MLES. The

configuration of the deep model is represented using a binary array, indicating which classifiers are selected or not. The representation's length is based on the number of layers and available classifiers used in MLES. The optimal set of classifiers is found by using a SAEA that maximises the classification accuracy of the MLES on a validation dataset. We introduce VISTA, a Variable-length Genetic Algorithm (VLGA) combined with a LSTM-based surrogation model to solve this optimisation problem to select the optimal EoC. Our contributions are as follows:

- We propose an ensemble selection method for MLES. The system is composed of multiple layers in which each layer contains an EoC. The output probability of each classifier in a layer is used as input for the classifiers in the next layer.
- We propose to represent all available classifiers in all layers in the MLES using a variable length approach. Since the number of classifiers in each layer can be different, and the optimal number of layers can be problem-dependent, this approach provides greater flexibility compared to a fixed-length approach.
- We develop a LSTM based-surrogation assisted VLGA for optimal configuration search of ensemble learning systems. Until now, SAEA-based approaches have only considered fixed-length inputs. LSTM is a popular method for time-series problems and can handle variable-length input, which makes it an ideal choice for converting from a variable-length to a fixed-length representation.
- Our experiments on 15 popular datasets show that the proposed method performs well compared to several benchmark algorithms.

II. Background and Related Works

A. Ensemble selection

Ensemble selection can improve predictive performance and computational efficiency by selecting a subset of classifiers from the entire ensemble. There are several approaches in selecting a subset of classifiers for an ensemble such as ordering-based, optimisation-based, and dynamic techniques. Dang et al. [6] proposed an ensemble of deep segmentation models in which the confidence of each prediction of the models are measured by a threshold to determine if the model is selected as a member of the ensemble. The optimal threshold is obtained by using Comprehensive Learning Particle Swarm Optimisation (CLPSO), a swarm intelligence algorithm. In [7], the authors proposed ECM-EFS, an ensemble feature selection based on an enhanced co-association matrix with a novel consensus strategy based on considering all results given by all base feature selectors, the importance of features, and the relationship between features. Ning et al. [8] proposed a novel sparse projection infinite selection ensemble for imbalanced classification, in which balanced versions of the datasets are iteratively sampled and the classifiers trained on these subsets are combined to create the final prediction. A graph-based approach combined with random sparse projection is used to adaptively sample diverse subsets of the original dataset. In [9], the authors proposed an ensemble selection method based on joint spectral clustering and structural sparsity, in which spectral clustering is proposed to learn pseudo cluster-labels on the transformed data while competent base classifiers are weighted by using structural analysis with regularization. Nguyen et al. [10] applied Ant Colony Optimisation (ACO) to search for both the optimal combining algorithm and the optimal set of the outputs of classifiers in the ensemble system.

Inspired by the successes of DNNs, several deep/MLES have been developed. gcForest [2] proposed in 2014 is the first system consisting of multiple layers, with each layer comprising of four Random Forest-based classifiers. After that, several MLESs were proposed including deep ensemble models of incremental classifiers [11], an ensemble of SVM classifiers [12], and deep ensemble models focusing on multi-label learning [13]. In [4], Nguyen et al. introduced MULES, a MLES which aims to select both optimal set of classifiers and the optimal set of features used by a selected classifier at each layer. The optimisation problem was formulated as a bi-objective problem to balance maximising the classification accuracy and increasing the diversity of the EoC in each layer. Although multiple layers ensemble can achieve better results than a single layer ensemble, constructing and optimizing a multiple layers ensemble requires much more computation time. Han et al. [14] proposed a variable length-based Genetic Algorithm to search for the optimal configuration of multiple layer ensemble. Dang et al. [15] proposed a two-layer ensemble of deep learning-based medical image segmentation models. The prediction of each pixel by each segmentation model is used as augmented data for the second layer, and the predictions in the second layer are then combined via a weight-based scheme in which each model has a different contribution to the final prediction.

B. Surrogate assisted evolutionary algorithms (SAEA)

In EA, a population is evolved through a number of generations to find the optimal solution. For each evolved candidate, it is necessary to evaluate its quality through a fitness function. For many real-world optimisation problems, evaluating the fitness function requires a lot of computation time [5] which shows a need of cost-saving fitness evaluation solutions when using EA. An approach to circumvent this problem is known as Surrogate-assisted evolutionary algorithm (SAEA) which uses low-cost surrogate models to evaluate the candidate solution. Common surrogate models which have been successfully used in SAEA are Polynomial response surface (PRS), Kriging, Radial basis function (RBF) and Support Vector Machine (SVM) [16].

In SAEA, as new candidates are created at each generation, it is necessary to update the surrogate model based on the fitness value of these new candidates. At each generation, it is necessary to find an appropriate scheme for choosing which candidates to use to update the surrogate model (known as model management). Generally, there are three types of model

management used to update the surrogate model: individual-based, generation-based, and hybrids [16]. Within generation-based management, all candidates undergo real fitness evaluation (FE). Following a certain number of generations, which can be either fixed or adaptive [17], the surrogate model is then updated. In contrast, only a small number of candidates are chosen for real FE at each generation in individual-based methods. The individuals can be chosen according to some criteria: either a random percentage of individuals are chosen, or the best is chosen for each generation, or the population might be clustered and the most representative individual in each cluster is chosen [18].

An innovative SAEA approach was developed in [19] to efficiently solve high-dimensional and computationally expensive optimisation problems. A generalized surrogate model is designed for this algorithm which is capable of handling both continuous and categorical variables and employs an effective update scheme for the surrogate model to improve its accuracy while reducing the computational cost. A dynamic SAEA framework was proposed in [20] to solve expensive structural optimisation problems. The framework incorporated a dynamic surrogate model to accelerate the optimisation process, and a multi-level EA to handle the complex and high-dimensional search space.

III. Proposed Approach

In this section, our proposed ensemble learning system called VISTA, which is based on surrogate-assisted VLGA with a LSTM network will be introduced.

A. Ensemble Selection for Multiple Layers Ensemble Systems

Suppose we have a set of $\mathcal{N}$ training observations, denoted as $\mathcal{D}$, where each observation has a feature vector x_n and its label $\hat{y}_n$ where $\hat{y}_n$ belongs to a label set $\mathcal{Y}$ of M labels. Our objective is to learn a hypothesis, represented by a classifier h, to approximate the unknown relationship $g : x_n \rightarrow \hat{y_n}$. For ensemble learning, we combine a set of K hypotheses to assign a label to each unlabelled instance.

The ensemble system introduced in this paper consists of multiple layers, and each layer consists of an EoC (an example is shown in Figure 1). In the first layer of a MLES, K learning algorithms are trained on the original training data to obtain a set of EoC. The Stacking algorithm is also used to generate input data for the second layer [4]. For each subsequent layer, the EoC of that layer is constructed using the output data from the previous layer concatenated with the original dataset. The predictions of the EoC of the last layer (s^{th} layer) are aggregated for a collaborative decision i.e., predictions corresponding to all class labels. For each instance, the label that corresponds to the maximum value among the predicted probabilities is assigned to it.

It is recognised that in each layer there exists a subset of EoC that performs competitively in comparison to the whole EoC [21]. In this study, we propose a novel ensemble selection approach to select the optimal subset of classifiers for each layer of MLES.

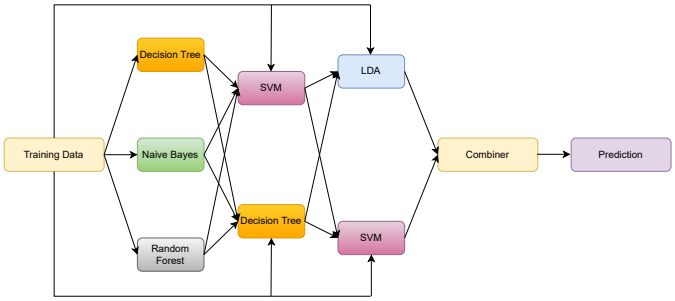

Fig. 1. An example of Multiple Layers Ensemble Systems

Layer 1				Layer 2				Layer 3			
0	1	0	0	1	1	0	1	1	0	0	1

Fig. 2. Chunk-based representation example

B. Optimisation Problems and Evolutionary Algorithm

The ensemble selection problem is modeled as a maximisation optimisation problem. Let's denote E as a representation of a configuration for an EoC which shows whether a classifier is selected or not while h_E is the combining model which combines the classifiers' predictions associated with E. The objective is to maximise classification accuracy when predicting labels for observations in a validation set $\mathcal{V}$:

$$max_E\{\frac{1}{|\mathcal{V}|}\sum_{x\in\mathcal{V}} \parallel h_E(x) = \hat{y} \parallel\} \tag{1}$$

In this study, we developed a surrogate-assisted variable length binary representation Genetic Algorithm (SA- VLGA) to search for the optimal set of classifiers for each layer in a MLES. The proposed SA-VLGA in this paper consists of two operators: CROSSOVER and MUTATION.

REPRESENTATION: We introduce a variable length binary representation method for multiple ensemble systems to facilitate ensemble selection. The representation's length is determined by both the number of layers s in the multiple layers ensemble and the quantity of available classifiers K in each layer. The value at each index in a representation is as the equation below:

$$h_k^{(i)} = \begin{cases} 1, k^{th} \text{ classifier at } i^{th} \text{ layer is selected, } i = \overline{1,s} \\ 0, otherwise \end{cases} \tag{2}$$

Figure 2 below provides a chunk-based representation for a three-layer ensemble system with four potential classifiers available in each layer, each chunk contains encoding for classifiers in a layer. For this configuration, the second classifier of the $1st$ layer, the first, second, and fourth classifier of the $2nd$ layer and the first and fourth classifier of the $3rd$ layer will be selected to construct the ensemble.

CROSSOVER: To control the crossover process, a crossover probability P_c is used. Crossover will be performed if the randomly generated crossover rate is less than P_c. In the

chunk-based crossover operation, for two selected parent candidates with s_1 and s_2 layers, two random numbers r_1 and r_2 are generated from the sets $\{K, 2 \times K, ..., s_1 \times K\}$ and $\{K, 2 \times K, ..., s_2 \times K\}$, respectively. Each parent exchanges its tail with the other while retaining its head.

MUTATION: A mutation probability parameter P_m is used to direct the mutation process. For each index of a representation, a flip mutation from 0 to 1 or 1 to 0 will occur when a randomly generated mutate rate is smaller than P_m.

C. LSTM based Surrogate model

To address the issue of the expensive computational cost of the search process for ensemble selection, we propose integrating a surrogate model with our VLGA approach. The surrogate model will be initialised and trained with individuals from several first generations. The trained surrogate model will then be used for fitness approximation for several next generations. It is noted that the surrogate model is designed as an adaptive model that will be continuously updated during the search process and it will only be retrained with newly collected individuals with actual fitness before being used for fitness approximation again.

Surrogate models typically use mathematical or statistical methods to approximate the relationship between the inputs and the outputs of a complex system. These methods often require a number of input variables. To handle the variable length encoding of the ensemble system in this paper, we proposed to use LSTM [22] to convert the variable length input to the fixed length input before feeding it to the surrogate model.

1) LSTM representation transformation: In this paper, we developed a chunk-based LSTM converter for the surrogate models. In [23], Zoph, Barret, et al. used a RNN (Recurrent Neural Network) to sample a string which encodes a DNN in a sequential manner. Deng et al. [24] proposed a predictive model for deep network performance before training to reduce expensive training cost. The suggested predictive model utilises the LSTM algorithm to capture information from each layer sequentially. The approach involves merging the encoded representations of individual layers into vectors, generating a unified description using LSTM, and subsequently feeding it into an MLP for network performance prediction.

Our study introduces a novel approach for optimising the search for multiple layers ensemble network structures. Unlike existing methods for predicting network performance, we propose using chunk-based LSTM to construct surrogate models. Additionally, our approach differs from prior research on surrogate models, as it applies surrogation to evolving variable length input data.

Suppose we have a representation E consisted of N chunks that we can denote E as $\{e_0, e_1, ...e_{N-1}\}$. The LSTM circular chunk-based converter is a multiple-layer LSTM algorithm that consists of n_{lstm} layers. Each LSTM layer in the chunk-based converter consists of a single unit only.

Each encoding chunk n representation is denoted as e_n, it is fed to single unit LSTM layer t, where $t = n\%n_{lstm}$. For each LSTM layer t, the output is denoted as O_t, the activation function is denoted as σ, W_t is the weighting matrix, and b_t is the bias vector. If only one chunk e_n is fed into an LSTM layer t, the output is $O_t = \sigma(W_t \times [e_n] + b_t)$. If more than one chunk $\{e_n, e_{n+n_{lstm}}, ..., e_{n+m\times n_{lstm}}\}$ is fed into an LSTM layer recurrently, starting from the second chunk in the set, the hidden state/output O_{t-1} is generated, $O_t = \sigma(W_t \times [e_n, O_{t-1}] + b_t)$ and is fed together with the next item in the set to the corresponding LSTM layer.

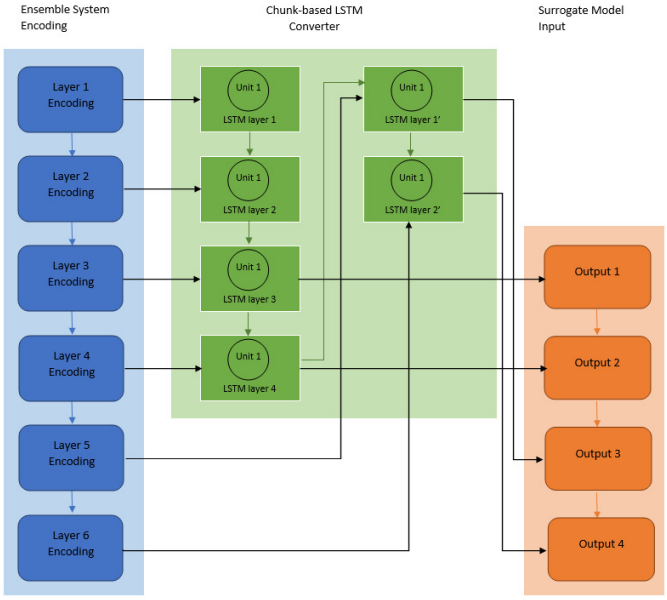

Fig. 3. LSTM chunk-based converter

In the developed chunk-based LSTM converter, tanh function is used as the activation method, which outputs zero-centred values that enable easy mapping of the output values as strongly negative, neutral, or strongly positive. The final outputs of all n_{lstm} LSTM layers are concatenated and formed as an input for the surrogate model.

An LSTM converter working scheme is demonstrated in Figure 3 in which a 6-chunk ensemble learning encodings is feeding into the 4 single unit LSTM layers chunk-based converter. From input encoding chunk 1 to chunk 4, they are fed into LSTM layers 1 to 4. Starting from input encoding chunk 5, the corresponding representation chunk is sent to LSTM layer 1 while encoding chunk 6 is sent to LSTM layer 2. The final output of the LSTM converter is a concatenate of single unit LSTM layer 3, LSTM layer 4, LSTM layer 1', and LSTM layer 2'.

We compare chunk-based converter to traditional LSTM converter using an example illustrated in Fig.4. Here an input encoding that reflects a 6-chunk ensemble network configuration is fed into a simple LSTM converter, comprising an LSTM layer with 100 units, another with 150 units, a third one with 50 units, and finally, a Dense layer with 50 units. The output encoding with a fixed length that matches the output size of the Dense layer (50) will be used for a surrogate model.

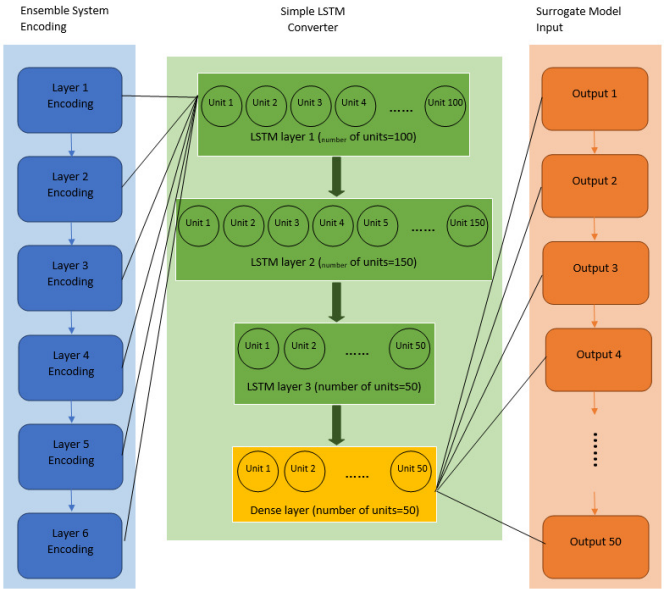

Fig. 4. LSTM simple converter

By contrast, in the chunk-based LSTM converter, there are multiple single-unit LSTM layers in a hierarchical structure. The encoding is divided into layer-based chunks and then sent to corresponding LSTM layers. Each single unit LSTM layer's output is concatenated to construct the final converted output. In the provided example of this simple LSTM converter, it's evident that the total number of units, spanning from the LSTM layers to the dense layers, is significantly greater compared to the subsequently proposed chunk-based LSTM converter. The converted fixed-length training data size for the surrogate model is also smaller.

The advantage behind the chunk-based LSTM is primarily in three folds. First, the chunk-based LSTM would be able to maintain layer-based information of the ensemble network structure when converting it into fixed-length training data for the proposed surrogate model. This includes the number of layers (chunks), the number of potential classifiers in each layer (chunk), and the structure of the variable-length encoding corresponding ensemble network. Secondly, the chunk-based LSTM converter in this research is single unit-based instead of the multiple unit-based structure of the original LSTM converter. The chunk-based LSTM converter thus has fewer units and hidden layers. Since we do not need the LSTM converter to learn extensive long-term memories, there is no need to maintain as many hidden layers as in the original LSTM converter. Finally, the chunk-based LSTM converter requires less parameter tuning, as we only need to decide the number of single unit layers, each layer will be designed to be the same.

2) Surrogate Model: In this paper, we developed an adaptive surrogate model to approximate the fitness value. Initially, the surrogate model will be trained with a specified number of generations defined by the parameter $surr_{train}$. After this training, the evaluation of individuals for the next set of generations defined by the parameter $surr_{estimate}$, will be replaced by surrogate model approximation. During each surrogate approximation iteration, the individual representation with the best approximate fitness will also be evaluated with the original objective function and then added to the training dataset for the surrogate model update when the $surr_{estimate}$ generations are completed. In the final generation of the search, each individual representation will be evaluated with the original objective function to obtain the actual fitness, and the best ensemble system configuration will be selected.

In this paper, we propose to build the surrogate model using the RBF (Radial Basis Function) [25] surrogate model. RBF is identified to be able to obtain better accuracy when dealing with nonlinear problems compared with Polynomial response surface (PRS) [26] and Kriging [27] in [5].

D. SA-LSTM-VLGA algorithm

1) Pseudo-code of SA-LSTM-VLGA is in Algorithm 1: Algorithm 1 is the pseudo-code of the proposed SA-LSTM-VLGA algorithm. The inputs to the algorithm include the training dataset $\mathcal{D}$, validation dataset $\mathcal{V}$, and parameters for the Genetic Algorithm including population size, number of generations, and crossover and mutation probability. The algorithm first generates a population of individuals randomly, and then calculates the actual fitness on the validation data using Algorithm 2. The evaluated candidates and fitness are converted using LSTM to generate data to train the surrogate model. Parents are selected through a tournament scheme, and if the chosen parents pass the crossover check, they will produce a pair of offspring. Subsequently, these offspring undergo mutation, where random positions in their genetic material are altered if a mutation occurs.

If the surrogate model is not used in the current generation, the fitness of each offspring on the validation data is also calculated using Algorithm 2. The newly evaluated population will be used to create new training data to train the surrogate model. If the surrogate model is used in the current generation, each offspring's fitness will be approximated by using the surrogate model. A new population is created based on these fitness approximations. The candidate with the best approximation fitness will be evaluated with true fitness function in Algorithm 2 and converted to training data to train the surrogate model. The genetic operation process is repeated until a new population of the same size as the original is generated. From the population of $2 * popSize$ individuals, the best individuals (in terms of fitness) are kept to use in the next generation. The algorithm iterates until it reaches the specified number of generations. Initially, the fitness of individuals in the final generation is computed, and the candidate with the highest fitness value is then selected as the solution to the problem.

2) Pseudo-code of fitness evaluation in Algorithm 2: Algorithm 2 is designed to compute the fitness and generate a multiple layers learning model corresponding to a given encoding. The encoding E specifies the number of layers and the classifiers used in each layer. On the i^{th} layer: (i) we train

Algorithm 1 SA-LSTM-VLGA

Require: Training dataset $\mathcal{D}$, Validation dataset $\mathcal{V}$, population size $popSize$, the number of generations $nGen$, crossover probability P_c, mutation probability: P_m, surrogate flag $surr$
Ensure: Optimal configuration of ensemble
1: Generate population and calculate the fitness of each individual in the population based on $\mathcal{V}$ using Algorithm 2
2: Initialize the surrogate model with the first population
3: Apply tournament selection
4: **for** $i \leftarrow nGen$ **do**
5: **if** $surr$ is $False$ **then**
6: $train_{iter} = train_{iter} + 1$
7: **else**
8: $estimate_{iter} = estimate_{iter}$+1
9: **end if**
10: **if** $train_{iter} == surr_{train}$ **then**
11: $surr \leftarrow$ **not** $surr$
12: $train_{iter} = 0$
13: **end if**
14: **if** $estimate_{iter} == surr_{estimate}$ **then**
15: $surr = True \rightarrow False, or False \rightarrow True$
16: $estimate_{iter} = 0$
17: **end if**
18: **while** $currentpopulationsize < 2 \times popSize$ **do**
19: Select a pair of individuals using tournament selection
20: Generate two random numbers $r_c, r_m \in [0, 1]$
21: **if** $r_c \leq P_c$ **then**
22: Partition the parents into head and tail segments, determined by two randomly chosen multiples of K
23: Exchange the tails of two parents to generate two new offspring
24: **end if**
25: **if** $r_m \leq P_m$ **then**
26: **for** each offspring **do**
27: Flip the binary value with mutation points
28: **end for**
29: **end if**
30: **if** $surr$ is $False$ **then**
31: Obtain the fitness of offspring using Algorithm 2
32: **else**
33: Calculate the fitness of the individual with the best fitness approximation
34: **end if**
35: Create new surrogate model training data with the new population using LSTM
36: **end while**
37: Use $popSize$ best individuals for the next generation
38: update surrogate model with updated training data
39: **end for**
40: **return** the best fitness individual from the last generation.

Algorithm 2 Fitness calculation and model generation based on an encoding

Require: Training dataset $\mathcal{D}$, Validation dataset $\mathcal{V}$ and encoding candidate E, number of folds T
Ensure: The fitness value of E and EoC associated with E
1: Retrieve the number of layer s and selected classifiers in each layer based on the representation in E
2: $\mathcal{L}_0 = D, \mathcal{V}_0 = V$
3: **for** $i \leftarrow 1, s$ **do**
4: Train selected classifiers $\{h_k^{(i)}\}$ for the i^{th} layer on $\mathcal{L}_{i-1}$
5: $\mathcal{P}_i = \varphi$
6: Use $\{h_k^{(i)}\}$ to predict for $\mathcal{V}_{i-1}$ to obtain $\mathcal{P}_i(\mathcal{V})$
7: **for** $t \leftarrow 1, T$ %generate the running data for the next layer **do**
8: $\mathcal{L}_{i-1} = \cup_{j=1}^{T} \mathcal{L}_{i-1}^{(j)}; \mathcal{L}_{i-1}^{(j1)} \cap \mathcal{L}_{i-1}^{(j2)} = \emptyset; |\mathcal{L}_{i-1}^{(j1)}| \approx |\mathcal{L}_{i-1}^{(j2)}|; 1 \leq j_1, j_2 \leq T, j_1 \neq j_2$
9: **for all** $\mathcal{L}_{i-1}^{(j)}$ **do**
10: Train selected classifiers on $\mathcal{L}_{i-1} - \mathcal{L}_{i-1}^{(j)}$
11: Utilize these classifiers to make predictions on $\mathcal{L}_{i-1}^{(j)}$ to obtain ${\mathcal{P}_i}^{(j)}$
12: $\mathcal{P}_i = \mathcal{P}_i \cup \mathcal{P}_i^{(j)}$ % add new predictions
13: **end for**
14: $\mathcal{L}_i = \mathcal{L}_0 \oplus \mathcal{P}_i$ %concatenation operation
15: **end for**
16: $\mathcal{V}_i = \mathcal{V}_0 \cup \mathcal{P}_i(\mathcal{V})$
17: **end for**
18: Apply Sum Rule method on $\mathcal{P}_s(\mathcal{V})$
19: Calculate fitness f of E by using (1)
20: **return** f and $h_k^{(i)} (i = 1, ..., s; k = 1, ..., K)$

the selected classifiers $\{h_k^{(i)}\}$ of this layer on the previous layer's generated training data $\mathcal{L}_{i-1}$ and (ii) we applied T-fold cross-validation procedure and the concatenation operator on predictions and original training data to generate training data for the $(i+1)^{th}$ layer denoted by $\mathcal{L}_i$ (Step 14). The classifier $\{h_k^{(i)}\}$ works on $\mathcal{V}_{i-1}$ to output the prediction $\mathcal{P}_i(\mathcal{V})$ which in fact contains the predictions for observations of $\mathcal{V}$ at the $(i-1)^{th}$ layer. The concatenation operator is applied to $\mathcal{P}_i(\mathcal{V})$ and $\mathcal{V}$ to generate the validation dataset for the $(i+1)^{th}$ layer. Upon completing the traversal of the final layer i.e. the s^{th} layer, we apply the Sum Rule [28] to combine the predictions in $\mathcal{P}_s(\mathcal{V})$ so as to gain the fitness value of encoding E. We also can determine the selected classifiers $\{h_k^{(i)}\}$ from E.

During the testing procedure, within every layer, the classifiers make predictions on the input test data and then combine the output with the original test sample. This generates new test data for the next layer. The final prediction is obtained by applying the combining function to the outputs of the classifiers of the last layer.

IV. EXPERIMENTAL STUDIES

A. Experimental Settings

The experiments were conducted on 15 different datasets gathered from various sources, including the UCI Machine Learning Repository and OpenML. Our method VISTA utilised five different classifiers in each layer. These classifiers in EoC were trained using 5 learning algorithms namely K Nearest Neighbour (K was set to 5), Naive Bayes classifiers (using Gaussian distribution), XgBoost (using 200 estimators), Random Forest (using 200 estimators), and Logistic Regression. The 5-fold Cross Validation procedure was used in each layer to populate the training data in multiple layers ensemble. We used 20% of the training data for validation. The maximum number of generations was set to 500, the population size was set to 100, and the crossover and mutation probabilities were set to 0.9 and 0.1, respectively based on the experiments in [29]. For the surrogate model, we used 100 generations for training and 10 generations for estimating by the set $surr_{train}$ to 100 and $surr_{estimate}$ set to 10. In each training iteration,

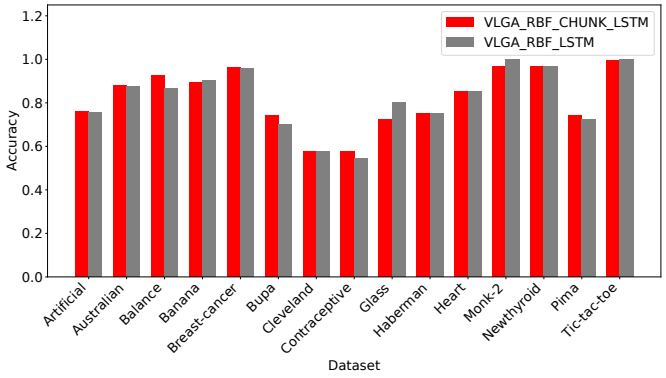

Fig. 5. The classification accuracy of chunk-based and conventional LSTM in VISTA

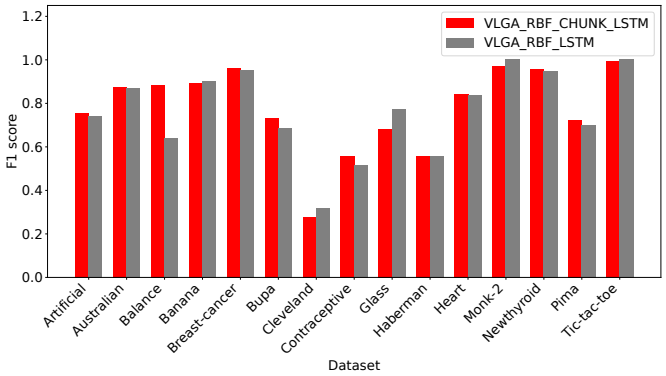

Fig. 6. The F1 scores of chunk-based and conventional LSTM in VISTA

each candidate along with their true fitness will be added as surrogate model training data. In each estimating iteration, the candidate with the best surrogate estimate approximate fitness will be evaluated with the original evaluation method and then added to the surrogate model as training data.

B. Comparison of the two LSTM converters

We compared the performances of the chunk-based LSTM converter and a conventional simple LSTM-based converter. The simple LSTM was implemented with two $ReLU$ layers (100 units and 150 units respectively), one $Tanh$ layer, and one $Dense$ layer. The chunk-based LSTM converter was implemented as described in section III-C1 where n_{lstm} is set to 10. Each component in the chunk-based LSTM converter was designed with 1 unit and $Tanh$ activation method.

The classification accuracies and F1 scores of two converter-based algorithms evaluated on 15 datasets are shown in Figure 5 and 6. Generally, the chunk-based LSTM performs better than the conventional LSTM. For classification accuracy, chunk-based LSTM is slightly better than conventional LSTM on 7 datasets. On some datasets, the differences in performance are significant, for example, on Balance (0.9255 vs. 0.8670) and Bupa (0.7404 vs. 0.7019). By contrast, conventional LSTM is better than chunk-based LSTM on 4 datasets Banana (0.9025 vs. 0.8943), Glass (0.8 vs. 0.7231), Monk-2

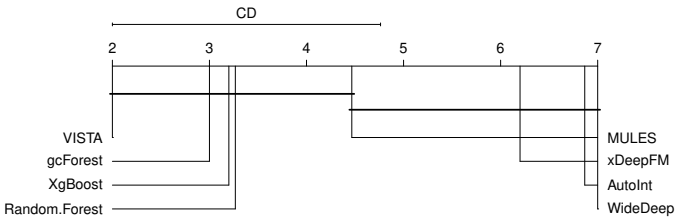

Fig. 7. The Nemenyi test result on classification accuracy

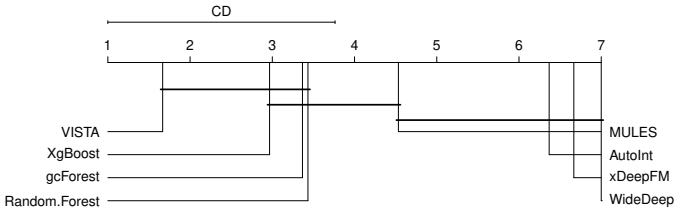

Fig. 8. The Nemenyi test result on F1 Score

(1 vs. 0.9692), and Tic-tac-toe (1 vs. 0.9931). For F1 score, the chunk-based LSTM is better and worse than conventional LSTM on 9 and 5 datasets, respectively. The results demonstrate the advantage of using the proposed chunk-based LSTM converter in generating training data for the surrogate model.

In the next section, we used the results of VISTA with the chunk-based approach to compare to those of benchmark algorithms.

C. Comparison to the benchmark algorithms

To evaluate VISTA's performance, we compared it with several ensemble methods and deep learning models. We used two popular ensemble methods namely Random Forest and XgBoost as benchmark algorithms; each consists of 200 learners. Additionally, we compared VISTA with two multiple layers learning models: gcForest (4 forests with 200 trees in each forest) [2] and MULES [4]. For MULES, we used the same parameter settings as in the original paper. We also compared VISTA to 3 deep learning models developed for tabular data namely WideDeep [30], xDeepFM [31], and AutoInt [32] in which we aim to show the outstanding performance of VISTA to the deep learning models. The performance of the experimental methods on 15 datasets was evaluated based on the Friedman test. Since the p-values corresponding to classification accuracy and F1 score of this test are 8.90e-12 and 3.61e-12, respectively which are smaller than a significant threshold of 0.05, we then rejected the null hypothesis of "no difference in the methods' performance". Subsequently, the Nemenyi post-hoc test was conducted to compare each pair of methods. The results of the Nemenyi test are shown in Figure 7 and 8. For accuracy, Nemenyi test result in Figure 7 shows that VISTA ranks first among all experimental methods and is better than xDeepFM, AutoInt, and WideDeep. gcForest ranks second and is also better than xDeepFM, AutoInt, and WideDeep. The top 4 methods ranked based on accuracy are VISTA, gcForest, XgBoost, and Random Forest in which the Nemenyi test shows that there are no differences in their

TABLE I
THE ACCURACY OF BENCHMARK ALGORITHMS AND VISTA ON EXPERIMENTAL DATASETS

Dataset	VISTA	WideDeep	xDeepFM	AutoInt	XgBoost	Random Forest	gcForest	MULES
Artificial	0.7619	0.6238	0.6381	0.6095	0.7619	**0.7905**	**0.7905**	0.7238
Australian	0.8792	0.8068	0.7874	0.7633	0.8744	**0.8889**	0.8792	0.8309
Balance	**0.9255**	0.8883	0.6702	0.5372	0.8457	0.8085	0.8564	0.8351
Banana	0.8943	0.5453	0.8597	0.4214	**0.8969**	0.8365	0.8654	0.8899
Breast-cancer	0.9610	0.6293	0.7902	0.9610	0.9561	0.9561	**0.9707**	0.9512
Bupa	**0.7404**	0.4615	0.5385	0.5865	0.7019	0.7308	0.7212	0.7019
Cleveland	0.5778	0.1333	0.1111	0.2889	0.5889	**0.6333**	0.6222	0.5889
Contraceptive	**0.5769**	0.3054	0.4457	0.5430	0.5701	0.5588	0.5566	0.5724
Glass	0.7231	0.2923	0.5231	0.2462	**0.7385**	0.7077	0.7077	0.6308
Haberman	**0.7500**	0.5435	0.7391	0.6630	0.7065	**0.7500**	0.7391	0.6630
Heart	**0.8519**	0.6049	0.6173	0.5926	0.7531	0.8025	0.8272	0.7654
Monk-2	0.9692	0.6231	0.6308	0.5538	**1.0000**	0.9615	0.9615	0.9615
Newthyroid	**0.9692**	0.7385	0.7231	0.9231	**0.9692**	0.9538	**0.9692**	0.9385
Pima	0.7403	0.5584	0.6840	0.6320	0.7186	**0.7576**	0.7359	0.6840
Tic-tac-toe	**0.9931**	0.6076	0.9688	0.7222	0.9861	0.8056	0.8160	0.9792

TABLE II
THE F1 SCORE OF BENCHMARK ALGORITHMS AND VISTA ON EXPERIMENTAL DATASETS

Dataset	VISTA	WideDeep	xDeepFM	AutoInt	XgBoost	Random Forest	gcForest	MULES
Artificial	0.7555	0.5654	0.5567	0.4543	0.7464	**0.7690**	0.7608	0.7123
Australian	0.8711	0.7972	0.7553	0.7460	0.8669	**0.8820**	0.8704	0.8165
Balance	**0.8802**	0.6217	0.5122	0.3759	0.5970	0.5664	0.6985	0.5863
Banana	0.8931	0.3529	0.8582	0.4177	**0.8956**	0.8341	0.8646	0.8890
Breast-cancer	0.9582	0.5041	0.7503	0.9582	0.9528	0.9533	**0.9691**	0.9474
Bupa	**0.7293**	0.3786	0.3846	0.3903	0.6937	0.7063	0.6899	0.6836
Cleveland	0.2746	0.0689	0.0400	0.1242	**0.3016**	0.2571	0.2559	0.2555
Contraceptive	**0.5542**	0.2440	0.3722	0.5251	0.5461	0.5204	0.5244	0.5224
Glass	0.6787	0.0754	0.2950	0.1662	**0.7540**	0.5765	0.5264	0.5113
Haberman	**0.5539**	0.5253	0.4250	0.3987	0.5237	**0.5539**	0.5221	0.5440
Heart	**0.8412**	0.3769	0.4939	0.5661	0.7271	0.7817	0.8056	0.7502
Monk-2	0.9685	0.4495	0.5604	0.5486	**1.0000**	0.9607	0.9607	0.9607
Newthyroid	**0.9548**	0.3839	0.3446	0.8713	0.9453	0.9260	0.9453	0.9052
Pima	0.7234	0.5570	0.5617	0.5696	0.6952	**0.7255**	0.7230	0.6478
Tic-tac-toe	**0.9921**	0.5362	0.9637	0.5682	0.9841	0.7383	0.7715	0.9763

performances. The 3 poorest methods are 3 deep learning-based methods.

In detail, VISTA ranks first on 7 datasets and ranks second on 6 datasets (see Table I). On some datasets like Balance and Heart, the accuracy of VISTA is about 3% better than the second-best method on this dataset (0.9255 vs. 0.8883 of WideDeep on Balance and 0.8519 vs. 0.8272 of gcForest). On some datasets like Bupa and Contraceptive, although VISTA performed well, the differences between VISA's performance and other top methods' performance are not significant. Meanwhile, our method is outperformed by Random Forest and gcForest on the Artificial and Cleveland datasets as our accuracy is 3% and 6% smaller than that of the first-ranked method.

For F1 score, the test result in Figure 8 shows that VISTA performed better than MULES, AutoInt, xDeepFM, and WideDeep. VISTA ranks first on 7 datasets and ranks second on the 7 datasets (see Table II). On the Cleveland dataset, although VISTA ranks fifth for accuracy, it ranks second for F1 score. It's important to highlight that the F1 score considers both False Negatives and False Positives in the detection results, as the metric represents the harmonic mean of Precision and Recall. This demonstrates VISTA's powerful ability to classify test samples with different class labels.

VISTA required a longer training duration compared to two deep ensemble models, namely gcForest and MULES. Taking the Breast-cancer dataset as an example, gcForest completed the training process in just 105 seconds, whereas VISTA took significantly more time, specifically 25341.82 seconds (with the maximum number of generations set to 500 and the population size of 100). Meanwhile, MULES (with the maximum number of generations set to 100 and the population size to 50) took its training process in 3154.86 seconds. It's worth mentioning that the training time for VISTA could be

further reduced through parallel implementation or implementing early stopping for VLGA. Conversely, the classification time of VISTA competes favorably with those of gcForest and MULES, as VISTA incorporates a limited number of classifiers in each layer. On the Breast-cancer dataset, for example, VISTA took only 0.18 second for classification in total. Meanwhile, gcForest completed the classification for all test instances in 0.25 second while MULES used 0.17 second for the same task.

V. CONCLUSIONS

In this study, we propose VISTA, a novel SAEA-based MLES. In each layer, the output of each classifier is concatenated with the original training data as the input of the next ensemble layer. A VLGA was proposed to search for the configurations of each layer i.e., which classifiers are present in the ensemble of a layer. Since the optimisation process requires high computational time, a LSTM-based surrogate model was used in this paper. The LSTM converts the variable-length configuration of each layer into a fixed-length representation, which allows a conventional surrogate model to predict the fitness value of a candidate in VLGA. In this paper, the Radial Basis Function (RBF) was chosen as the surrogate model. We performed experiments comparing two types of LSTM converters and the results show that the chunk-based LSTM converter consistently is slightly better than the normal LSTM converter. The experimental results on 15 popular datasets show that VISTA performs better than other benchmark algorithms, including two ensemble methods (Random Forest and XgBoost), two multiple layers ensemble methods (gcForest and MULES), and three deep learning methods for tabular data (WideDeep, xDeepFM, and AutoInt).

ACKNOWLEDGMENT

This work was supported by the Scottish Government through The RSE Scotland Asia Partnerships Higher Education Research (SAPHIRE) Fund [grant number 2970].

REFERENCES

[1] M. Fernández-Delgado, E. Cernadas, S. Barro et al., 'Do we need hundreds of classifiers to solve real world classification problems?', J. Mach. Learn. Res., vol. 15, no. 1, pp. 3133–3181, Jan. 2014.

[2] Z.-H. Zhou and J. Feng, 'Deep Forest: Towards An Alternative to Deep Neural Networks', Proceedings of IJCAI, pp. 3553–3559, 2017.

[3] T. T. Nguyen, M. T. Dang, V. A. Baghel, et al., 'Evolving interval-based representation for multiple classifier fusion', Knowl.-Based Syst., vol. 201–202, p. 106034, 2020.

[4] T.T. Nguyen, N. V. Pham, T. Dang, et al., 'Multi-layer heterogeneous ensemble with classifier and feature selection', in GECCO, 2020, pp. 725–733.

[5] C. He, Y. Zhang, D. Gong et al., 'A review of surrogate-assisted evolutionary algorithms for expensive optimization problems', Expert Systems with Applications, vol. 217, p. 119495, 2023.

[6] T. Dang, T. T. Nguyen, J. McCall et al., 'Ensemble Learning based on Classifier Prediction Confidence and Comprehensive Learning Particle Swarm Optimisation for Medical Image Segmentation', in IEEE Symposium Series on Computational Intelligence (SSCI), 2022, pp. 269–276.

[7] T. Wu, Y. Hao, B. Yang et al., 'ECM-EFS: An ensemble feature selection based on enhanced co-association matrix', Pattern Recognition, vol. 139, p. 109449, 2023.

[8] Z. Ning, Z. Jiang, and D. Zhang, 'Sparse projection infinite selection ensemble for imbalanced classification', Knowl.-Based Syst., vol. 262, p. 110246, 2023.

[9] Z. Wang, S. Zhao, Z. Li et al., 'Ensemble selection with joint spectral clustering and structural sparsity', Pattern Recognition, vol. 119, p. 108061, 2021.

[10] T. T. Nguyen, A. V. Luong, T. M. Van Nguyen et al., 'Simultaneous Meta-Data and Meta-Classifier Selection in Multiple Classifier System', in Proceedings of GECCO, 2019, pp. 39–46.

[11] A. V. Luong, T. T. Nguyen, and A. W.-C. Liew, 'Streaming Active Deep Forest for Evolving Data Stream Classification', CoRR, vol. abs/2002.11816, 2020, [Online].

[12] Z. Qi, B. Wang, Y. Tian et al., 'When Ensemble Learning Meets Deep Learning: a New Deep Support Vector Machine for Classification', Knowl.-Based. Syst., vol. 107, pp. 54–60, 2016.

[13] L. Yang, X.-Z. Wu, Y. Jiang et al., 'Multi-Label Learning with Deep Forest', CoRR, vol. abs/1911.06557, 2019, [Online].

[14] K., Han, T. Pham, T.H. Vu et al. VEGAS: a variable length-based genetic algorithm for ensemble selection in deep ensemble learning. ACIIDS, 2021, pp. 168–180.

[15] T. Dang, T. T. Nguyen, J. McCall et al., 'Two layer Ensemble of Deep Learning Models for Medical Image Segmentation', Cognitive Computation (2024). https://doi.org/10.1007/s12559-024-10257-5.

[16] Y. Jin, 'Surrogate-assisted evolutionary computation: Recent advances and future challenges', Swarm Evol. Comput., vol. 1, no. 2, pp. 61–70, 2011.

[17] Y. Jin, H. Wang, T. Chugh et al., 'Data-Driven Evolutionary Optimization: An Overview and Case Studies', IEEE Trans. Evol. Comput., vol. 23, no. 3, pp. 442–458, 2019.

[18] T. Dang, A. V. Luong, A. W. C. Liew et al., 'Ensemble of deep learning models with surrogate-based optimization for medical image segmentation', in IEEE CEC, 2022, pp. 1–8.

[19] X. Cai, L. Gao, and X. Li, 'Efficient Generalized Surrogate-Assisted Evolutionary Algorithm for High-Dimensional Expensive Problems', IEEE Trans. Evol. Comput., vol. 24, no. 2, pp. 365–379, 2020.

[20] M. Yu, X. Li, and J. Liang, 'A Dynamic Surrogate-Assisted Evolutionary Algorithm Framework for Expensive Structural Optimization', Struct. Multidiscip. Optim., vol. 61, no. 2, pp. 711–729, Feb. 2020.

[21] T.T. Nguyen, T.T.T Nguyen, X.C. Pham, A.W.C. Liew, 'A novel combining classifier method based on Variational Inference', Pattern Recognition, vol. 49, pp. 198-212, 2016.

[22] S. Hochreiter and J. Schmidhuber, 'Long Short-Term Memory', Neural Comput., vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[23] B. Zoph, V. Vasudevan, J. Shlens et al., 'Learning Transferable Architectures for Scalable Image Recognition', in CVPR, 2018, pp. 8697–8710.

[24] B. Deng, J. Yan, and D. Lin, 'Peephole: Predicting Network Performance Before Training', CoRR, vol. abs/1712.03351, 2017, [Online].

[25] M. D. Buhmann, 'Radial basis functions', Acta Numer., vol. 9, pp. 1–38, 2000.

[26] G. E. P. Box and K. B. Wilson, 'On the Experimental Attainment of Optimum Conditions', in Breakthroughs in Statistics: Methodology and Distribution, 1992, pp. 270–310.

[27] D. G. Krige, 'A statistical approach to some basic mine valuation problems on the Witwatersrand', Journal of the Southern African Institute of Mining and Metallurgy, vol. 52, no. 6, pp. 119–139, 1951.

[28] J. Kittler, M. Hatef et al., 'On combining classifiers', IEEE Trans. Pattern Anal. Machine Intell., vol. 20, no. 3, pp. 226–239, 1998.

[29] A. Hassanat, K. Almohammadi, E. Alkafaween et al., 'Choosing Mutation and Crossover Ratios for Genetic Algorithms—A Review with a New Dynamic Approach', Information, vol. 10, no. 12, 2019.

[30] H.-T. Cheng et al., 'Wide and Deep Learning for Recommender Systems', in Proceedings of DLRS, 2016, pp. 7–10.

[31] J. Lian, X. Zhou, F. Zhang et al., 'XDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems', in Proc. ACM SIGKDD Int., 2018, pp. 1754–1763.

[32] W. Song et al., 'AutoInt: Automatic Feature Interaction Learning via Self-Attentive Neural Networks', in Proc. ACM Int. Conf. Inf. Knowl. Manag., 2019, pp. 1161–1170.