Weighted ensemble of deep learning models based on comprehensive learning particle swarm optimization for medical image segmentation.

DANG, T., NGUYEN, T.T., MORENO-GARCIA, C.F., ELYAN, E. and MCCALL, J.

2021

Weighted Ensemble of Deep Learning Models based on Comprehensive Learning Particle Swarm Optimization for Medical Image Segmentation

Truong Dang[1], Tien Thanh Nguyen[2], Carlos Francisco Moreno-García[3], Eyad Elyan[4] and John McCall[5]
School of Computing, Robert Gordon University
Aberdeen, UK
Email: [1]t.dang1@rgu.ac.uk, [2]t.nguyen11@rgu.ac.uk, [3]c.moreno-garcia@rgu.ac.uk, [4]e.elyan@rgu.ac.uk, [5]j.mccall@rgu.ac.uk

***Abstract*—In recent years, deep learning has rapidly become a method of choice for segmentation of medical images. Deep neural architectures such as UNet and FPN have achieved high performances on many medical datasets. However, medical image analysis algorithms are required to be reliable, robust, and accurate for clinical applications which can be difficult to achieve for some single deep learning methods. In this study, we introduce an ensemble of classifiers f or s emantic segmentation of medical images. The ensemble of classifiers h ere i s a set of various deep learning-based classifiers, a iming t o achieve better performance than using a single classifier. W e propose a weighted ensemble method in which the weighted sum of segmentation outputs by classifiers i s u sed t o c hoose t he final segmentation decision. We use a swarm intelligence algorithm namely Comprehensive Learning Particle Swarm Optimization to optimize the combining weights. Dice coefficient, a popular performance metric for image segmentation, is used as the fitness criteria. Experiments conducted on some medical datasets of the CAMUS competition on cardiographic image segmentation show that our method achieves better results than both the constituent segmentation models and the reported model of the CAMUS competition.**

***Index Terms*—image segmentation, deep neural networks, ensemble learning, ensemble method, particle swarm optimization**

I. Introduction

Image segmentation is the process of partitioning an input image into regions which correspond to different objects or parts of an object. Segmentation of medical images is considered very important in providing noninvasive information about human body structure [40], which have a vital role in numerous biomedical imaging applications, such as tissue volumes quantification, d iagnosis, p athology localization, study of anatomical structure, treatment planning, and computer-integrated surgery [42]. Automation of segmentation to integrate into clinical processes is therefore desirable. With the success of deep learning in image classification [14] in 2012, practitioners in medical image analysis took notice of these developments and applied it to segmentation of medical images. It is well known that localization and interpolation of anatomical structures in medical images, which is a key step in radiological workflow, was performed by handcrafting image filters to extract anatomical signatures [15]. This is a time-consuming process and requires expert knowledge. By applying deep learning, practitioners have been able to achieve many successes such as organs detection using 3D dynamic contrast-enhanced MRI scans over a period of time [17] and automatic segmentation in brain images using 3D convolutional deep learning architecture on mini-batches of multiple cubes of brain data [18]. Deep learning methods are highly effective when the number of available samples are large during a training stage. For example, in ImageNet Large Scale Visual Recognition Challenge (ILSVRC), the dataset contained 1 million annotated images. Medical datasets meanwhile are considerably smaller, typically less than 1,000 images [15]. This poses a problem for creating deep models for medical imaging which are robust against overfitting. Another problem is that the process of training deep neural networks using popular optimizers such as Stochastic Gradient Descent (SGD) generally require much manual tuning of optimization parameters such as learning rates and convergence criteria [41]. In recent years there have been many alternative optimization methods for deep learning which require less parameter tuning, such as Adam [43]. However, these methods do not generalize as well compared to traditional methods such as SGD [44]. The manual parameter tuning causes a challenge in selecting suitable deep models for a specific problem. A solution to these difficulties is to combine multiple deep learning models trained on medical image datasets which would guarantee better predictions compared to using individual deep models.

Ensemble learning is a popular machine learning technique in which multiple learning methods are combined to solve a computational intelligence problem. [48] tested 179 classifiers on 121 datasets and the results indicated that ensemble-based methods achieved the top ranks. In this study, we introduce an ensemble of deep learning methods for the problem of semantic medical image segmentation. The ensemble includes a number of different segmentation algorithms in which their outputs are combined by a combining algorithm to obtain the collaborated prediction. It is recognized that different segmentation algorithms will perform well on different subsets of examples because of the nature and size of training sets they have been exposed to and because of method-intrinsic factors. Therefore we focus on improving the effectiveness of

the ensemble by using a weight-based combining method. On a particular problem, some segmentation algorithms will contribute more to the final combining result by associating them larger weights than those of other ones. The final prediction is made by using a weighted sum on the outputs of segmentation algorithms. The weights are chosen to maximize the Dice coefficient, which is a popular performance metric in segmentation, based on a cross-validation procedure on the training data. We empirically compared our proposed ensemble with some well-known deep learning benchmark algorithms on several medical datasets of the CAMUS competition on cardiographic image segmentation [38]. In section 2, we briefly introduce ensemble learning, weighted combining model, Particle Swarm Optimization and Comprehensive Learning, and techniques for medical image segmentation problem. In section 3, we give a detailed description of the proposed ensemble. Experimental studies on a number of datasets are provided in Section 4, followed by conclusions in Section 5.

II. Background and Related Work

A. Ensemble System and Weighted Combining Model

Ensemble systems are typically built by generating diverse classifiers and then combine them to make a final decision. The first stage is done by training a learning algorithm on multiple training sets generated from the original training data or training different learning algorithms on the original training data to generate Ensemble of Classifiers (EoC) [1], [2]. The second stage uses a combining method working on the predictions of the generated classifiers for the final decision. Fixed combining methods are frequently applied to the predictions of classifiers to predict class labels. Popular fixed combining methods use fixed combining rules such as the Sum Rule, Product Rule, Min Rule, Max Rule, Median Rule, and Majority Vote Rule [3]. In simple fixed combining rules, all classifiers are treated equally in the aggregation step, i.e. all classifiers make an equal contribution in the final collaborated prediction. It is recognized that the equal contribution of classifiers may downgrade the performance of EoC because classifiers perform differently on a particular dataset and some classifiers need to contribute more than the others. Weighted combining model, in contrast, assumes that each classifier puts a different weight on the combining result. The weights and predictions are used to generate a set of combinations associated with the class labels. The predicted class label for a sample is then decided by selecting the maximum value among these combinations. There are some techniques to obtain the combining weights. Nguyen et al. [1] searched for the weights by minimizing the distance between these combinations computed on the training data and the class label of training observations given in the crisp form. Zhang and Zhou [4] proposed using linear programming to find the weights. Sen et al. [5] searched for the combining weights by minimizing the hinge loss function of the combination and the class labels of training data. Pacheco et al. [29] performed ensemble selection and pruning of deep learning classifiers by learning the Dirichlet distribution of the output probabilities and optimizing the weights dynamically using a loss function based on Mahalanobis distance.

B. Particle Swarm Optimization and Comprehensive Learning

Evolutionary Computation (EC) is a family of algorithms inspired by biological evolution for global optimization. One of the most popular methods of EC is Particle Swarm Optimization (PSO) [6], a swarm-based algorithm inspired by the emergent motion of a flock of birds searching for food. This algorithm simultaneously performs a local exploitation within each particle and global exploration among the whole swarm. For a U-dimension optimization problem, PSO maintains a number of particles whose positions are defined by $x_i = (x_i^1, x_i^2, ..., x_i^U), i = 1, ..., N$ where N is the number of particles. A velocity $v_i = (v_i^1, v_i^2, ..., v_i^U)$ is associated with each particle x_i. PSO ensures each particle learns from the whole swarm during its search by updating each particle's velocity based on its current velocity, local best position, and global best position.

Because all particles learn from the global best position, PSO can converge prematurely at a local optimum [8]. In 2006, Liang et al. proposed Comprehensive Learning PSO (CLPSO) [8] which addresses this shortcoming by having each particle learn from all particles' local best position. Specifically, each particle with U-dimension will also have a U-dimension exemplar vector $e_i = (e_i^1, e_i^2, ..., e_i^U)$ for comprehensive learning. The exemplar vector is introduced for a particle to learn from the local best $(pbest)$ of itself as well as all the other particles. For example, a particle with the position $(0.13, 0.43, 0.22, 0.74, 0, 11)$, the velocity $(0.48, 0.25, 0.52, 0.13,$ -0.15), and the exemplar $(6, 8, 4, 8, 4)$ would learns/updates the 3rd dimension position value based on the 3rd dimension position value of the 4th particle's $pbest$.

A particle is assigned randomly with an exemplar vector at initialization. When a particle's $pbest$ does not improve after a number of iterations, the exemplar will be updated. In order to choose which particle to learn from for each dimension, the algorithms selects randomly two different particles and the one with higher fitness value will be assigned as the exemplar for the updated particle on the corresponding dimension [8], [9]. Therefore, only one acceleration of constant c is needed. The updated equation is given by:

$$v_i^u \leftarrow a \times v_i^u + c \times r_1 \times (pbest_{e_i^u}^u - x_i^u) \tag{1}$$

in which a is the inertia weight which controls the velocity speeding rate, c is an acceleration constant used to control the learning rate of the exemplars' local best, $pbest_{e_i^u}^u$ is the u^{th} dimension of particle's best position referring to the u^{th} dimension of exemplar e_i, and r_1 is a random number drawn from a uniform distribution over $[0, 1]$. Considering that CLPSO has demonstrated state-of-the-art global search capabilities in various applications [45], such as optimizing reactive power dispatch [46] and [47] optimizing network security, in this paper we use CLPSO as an optimization routine for our proposed method.

C. Medical Image Segmentation

Many research efforts have been made to apply deep learning to medical image segmentation. An example is UNet [20] which consists of an equal number of upsampling and downsampling layers. Each downsampling layer has a skip connection which concatenates its output feature map with the input of the corresponding upsampling layer. This allows the network to take into the full context of the whole image, which is beneficial in performing segmentation task. Other authors have extended this architecture to handle 3D medical data, such as VNet [22], which performs 3D image segmentation using 3D convolutional layers with an objective function based on Dice coefficient. Although these specific architectures achieved remarkable results, many authors have also obtained excellent segmentation results via patch-based deep neural networks. One of the earliest papers on applying deep learning to medical image segmentation performed pixel-wise segmentation of membranes in electron microscopy imagery in a sliding window fashion [25]. More recent papers use architectures based on Fully Convolutional Neural Network (fCNN) [26] over sliding-window due to computational efficiency. A notable examples is vertebral body segmentation in MR images using 3D fCNNs to generate vertebral body likelihood maps for deformable models [27]. Some researchers have also applied graphical models such as Markov Random Fields (MRFs) [28] and Conditional Random Fields (CRFs) [19] on top of the likelihood maps produced by fCNNs to act as label regularizers.

III. Proposed method

Let $\mathbf{D}$ be the training set of N observations $\{(\mathbf{I}_n, \mathbf{Y}_n)\}_{n=1}^{N}$, where $\mathbf{I}_n = \mathbf{I}_n(i,j), 1 \le i \le W, 1 \le j \le H$ is an image in the training set and $\mathbf{Y}_n$ be its corresponding ground truth. Each image is given with a number of channels. In this study, we work on grayscale images which have only one channel. The ground truth $\mathbf{Y}_n$ is also an image with size $W \times H, \mathbf{Y}_n = \mathbf{Y}_n(i,j)$ showing which label each pixel belongs to $\mathbf{Y}_n(i,j) \in \mathcal{Y}$, where $\mathcal{Y} = \{y_m\}_{m=1}^{M}$ is a set of labels. Totally, we have $N \times W \times H$ pixels and their corresponding labels. For the semantic image segmentation problem, we aim to learn a hypothesis $\mathbf{h}$ (i.e., classifier) based on the relationship between each pixel $\mathbf{I}_n(i,j)$ and its corresponding label $\mathbf{Y}_n(i,j)$ of the training data and then use this hypothesis to assign a label on each pixel of an unsegmented image. The classifier $\mathbf{h}$ is obtained by training a segmentation algorithm on the training data $\mathbf{D}$. Given an image, $\mathbf{h}$ assigns a class label to each pixel, and the segmentation result for all pixels of the input image constitutes the segmented image.

We develop an EoC for solving the image segmentation problem. We denote $\mathbf{K} = \{\mathcal{K}_k\}_{k=1}^{K}$ as the set of K segmentation algorithms. In the ensemble, we train an EoC including K different classifiers $\{\mathbf{h}_k\}_{k=1}^{K}$ and then use a combining algorithm $\mathbf{C}$ to form the final decision making: $\hat{\mathbf{h}} = \mathbf{C}\{\mathbf{h}_k\}_{k=1}^{K}$. The EoC $\{\mathbf{h}_k\}_{k=1}^{K}$ is generated by training K segmentation algorithms on the training set $\mathbf{D}$. We then generate the predictions of pixels in training images and then train the combining algorithm on these predictions. In detail, we use the Stacking algorithm [2] to generate the predictions for pixels of training images. First, we divide training set $\mathbf{D}$ into T disjoint parts $\{\mathbf{D}_1, ..., \mathbf{D}_T\}$, where $\mathbf{D} = \mathbf{D}_1 \cup ... \cup \mathbf{D}_T, \mathbf{D}_i \cap \mathbf{D}_j = \emptyset (i \ne j), |\mathbf{D}_1| \approx ... \approx |\mathbf{D}_T|$, and their corresponding $\{\tilde{\mathbf{D}}_1, ..., \tilde{\mathbf{D}}_T\}$ in which $\tilde{\mathbf{D}}_t = \mathbf{D} - \mathbf{D}_t$. The segmentation algorithm $\mathcal{K}_j$ trains on $\tilde{\mathbf{D}}_i$ to obtain a classifier ${C_j}^i$. ${C_j}^i$ works on the images in $\mathbf{D}_i$ to output the probability reflecting how supportive a classifier is to a class label for each pixel. The predictions for an image $\mathbf{I}$ is given in an $(W \times H) \times (M \times K)$ matrix $\mathbf{P}(\mathbf{I})$:

$$\mathbf{P}(\mathbf{I}) = \begin{bmatrix} P_1(y_1|\mathbf{I}(1,1)) & \cdots & P_1(y_M|\mathbf{I}(1,1)) & \cdots & P_K(y_1|\mathbf{I}(1,1)) & \cdots & P_K(y_M|\mathbf{I}(1,1)) \\ P_1(y_1|\mathbf{I}(1,2)) & \cdots & P_1(y_M|\mathbf{I}(1,2)) & \cdots & P_K(y_1|\mathbf{I}(1,2)) & \cdots & P_K(y_M|\mathbf{I}(1,2)) \\ \vdots & \cdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ P_1(y_1|\mathbf{I}(W,H)) & \cdots & P_1(y_M|\mathbf{I}(W,H)) & \cdots & P_K(y_1|\mathbf{I}(W,H)) & \cdots & P_K(y_M|\mathbf{I}(W,H)) \end{bmatrix} \quad (2)$$

in which $P_k(y_m|\mathbf{I}(i,j))$ is the probability that the pixel $\mathbf{I}(i,j)$ belongs to the class label y_m given by the classifier generated by using $\mathcal{K}_k$ for each $k = 1, ..., K; m = 1, ..., M$ and $\sum_{m=1}^{M} P_k(y_m|\mathbf{I}(i,j)) = 1$ [12], [13]. The prediction for all images in the training set $\mathbf{D}$ is given by a $(N \times W \times H) \times (M \times K)$ matrix

$$\mathcal{P} = \begin{bmatrix} \mathbf{P}(\mathbf{I}_1) \\ \mathbf{P}(\mathbf{I}_2) \\ \cdots \\ \mathbf{P}(\mathbf{I}_N) \end{bmatrix} \quad (3)$$

The next step is to train the combining algorithm on $\mathcal{P}$. There are two combining models developed for the ensemble systems, namely representation-based model and weighted combining-based model [13]. The representation-based model creates M representations for M class labels on the predictions of the training data and then assigns class label which is associated with the biggest value among similarities (or the smallest value among dissimilarities) between the prediction for each test sample and the M representations [2], [12], [13]. Meanwhile, in the weighted combining-based model, classifiers contribute differently to combining by using different combining weights. The weights may vary for each classifier or among pairwise of classifier – class label. In this study, we use a weighted combining-based model which is based on the weight matrix $\mathbb{W} = \{w_{k,m}\}$ in which $w_{k,m}$ is the weight of the k^{th} classifier on the m^{th} class $(k = 1, ..., K; m = 1, ..., M)$. Since the ground truths of the training images are given in advance, the weights of classifiers on the class labels can be obtained by discovering the relationship between predictions $\mathcal{P}$ and the class labels of the pixels of the training images. First, the class membership of a pixel $\mathbf{I}(i,j)$ associated with the class y_m is obtained by a linear combination of the predictions and the associated weights as:

$$CM_m(\mathbf{I}(i,j)) = \sum_{k=1}^{K} w_{k,m} P_k(y_m|\mathbf{I}(i,j)) = \mathbb{P}_m \mathbb{W}_m \quad (4)$$

with $\mathbb{P}_m = [P_1(y_m|\mathbf{I}(i,j)), P_2(y_m|\mathbf{I}(i,j))..., P_K(y_m|\mathbf{I}(i,j))]$ and $\mathbb{W}_m = [w_{1,m}, ..., w_{K,m}]^T$. We then compare the class memberships associated with the class labels and assign the

class label y_s to pixel $\mathbf{I}(i,j)$ if its associated class membership is the biggest among all memberships.

$$\mathbf{I}(i,j) \in y_s \text{ if } s = argmax_{m=1,...,M} CM_m(\mathbf{I}(i,j)) \quad (5)$$

In this study, we propose an approach to search for the combining weights $\mathbb{W}$ by maximizing the Dice coefficient computed on the predictions of the proposed ensemble with the combining weights $\mathbb{W}$ on training data. Let **pred** and **ground** denote the final predictions and ground truths of all training pixels:

$$\textbf{pred} = \{pred_1, pred_2 ... pred_M\} \quad (6)$$

$$\textbf{ground} = \{ground_1, ground_2 ..., ground_M\} \quad (7)$$

in which $pred_m$ is the vector of size $(N \times W \times H, 1)$ in which its element is the prediction for each pixel belonging to the class label y_m in the form of crisp label i.e. in $\{0, 1\}$. Likewise $ground_m$ is the vector of size $(N \times W \times H, 1)$ associated with the class label y_m which is the ground truth of each pixel in the form of crisp label i.e. in $\{0, 1\}$. $pred_m$ is obtained based on the classification rule in 5 while $ground_m$ is obtained from the ground truths $\{\mathbf{Y}_n\}$. The Dice coefficient associated with the class label y_m is given by:

$$DC_m = \frac{2 \times pred_m^T ground_m}{||pred_m||^2 + ||ground||^2} \quad (8)$$

The Dice coefficient is the average of all Dice coefficients associated with the class labels.

$$DC_{avg} = \frac{1}{M} \sum_{m=1}^{M} DC_m \quad (9)$$

We maximize the Dice coefficient to find the $\mathbb{W}$. This optimization problem is solved by using the CLPSO method.

$$\begin{aligned} \max_{\mathbb{W}} \quad & DC_{avg} \\ \text{s.t.} \quad & 0 \leq w_{k,m} \leq 1 \end{aligned} \quad (10)$$

In this study, we use three popular segmentation algorithms namely UNet, LinkNet, and Feature Pyramid Network (FPN) to train the EoC. It is widely recognized that most segmentation algorithms based on deep learning are inspired by Fully Convolutional Network (FCN) [26]. This architecture adapts an existing classification network, such as VGG16, to the segmentation problem by replacing the fully connected layers with convolutional layers, followed by upsampling to produce dense pixel-level result. Deep networks specifically designed for medical image segmentation have also been introduced. A notable example is UNet [20], which consists of a contracting path and an expanding path. The contracting path consists of a number of downsampling operations on the input image in order to extract useful features, while the expanding path upsample the image back to its original size for the final prediction. In order to help with localization, high resolution features from the contracting path are concatenated with the upsampled output. This is an example of *encoder-decoder architecture*, in which an image goes through an encoder which contracts the image size, and is then decoded back to the original size to get the segmentation result. Other examples include LinkNet [21] which takes the sum of the upsampled output and the corresponding features in the contracting path, and FPN [32] which concatenates features of all levels in the expanding path to help with the final prediction.

Algorithm 1 Training process

Input: Training images $\mathbf{D}$, K segmentation algorithms $\{\mathcal{K}_k\}_{k=1}^K$, parameters for the CLPSO: maximum number of iteration $maxT$, population size $nPop$, acceleration constant c

Output: The optimal weights $\hat{\mathbb{W}}$ and $\{\mathbf{h}_k\}_{k=1}^K$

1: Learn K classifiers $\{\mathbf{h}_k\}_{k=1}^K$ on $\mathbf{D}$ using $\{\mathcal{K}_k\}_{k=1}^K$
2: $\mathcal{P} = \emptyset$
3: $\mathbf{D} = \mathbf{D}_1 \cup ... \cup \mathbf{D}_T, \mathbf{D}_i \cap \mathbf{D}_j = \emptyset (i \neq j)$
4: **for** each $\mathbf{D}_i$ **do**
5: $\quad \tilde{\mathbf{D}}_i = \mathbf{D} - \mathbf{D}_i$
6: $\quad$ Learn ensemble of classifiers on $\tilde{\mathbf{D}}_i$ using $\{\mathcal{K}_k\}_{k=1}^K$
7: $\quad$ Classify images in $\mathbf{D}_i$ by these classifiers
8: $\quad$ Add outputs on samples in $\mathbf{D}_i$ to $\mathcal{P}$ 3
9: Use the CLPSO method: for each candidate $\mathbb{W}$, compute the associated Dice coefficient using Algorithm 2
10: Select the optimal $\hat{\mathbb{W}}$ with the best Dice coefficient
11: **return** $\hat{\mathbb{W}}$ and $\{\mathbf{h}_k\}_{k=1}^K$

The pseudo-code of the training process of the proposed system is present in Algorithm 1. The algorithm gets the inputs including the training images $\mathbf{D}$, K segmentation algorithm $\{\mathcal{K}_k\}_{k=1}^K$, and parameters for the CLPSO (the population size $popSize$, the number of iterations $iter$, and learning rate controller C). First, we train K segmentation algorithms $\{\mathcal{K}_k\}_{k=1}^K$ on $\mathbf{D}$ to create classifiers $\{\mathbf{h}_k\}_{k=1}^K$. Then we generate the prediction $\mathcal{P}$ for all pixels of training images by using the Stacking algorithm (Step 2-8). For each candidate $\mathbb{W}$ generated in the CLPSO, we call Algorithm 2 to calculate its associated Dice coefficient. In Algorithm 2, for each row of $\mathcal{P}$ i.e. the predictions of K classifiers for a pixel, we compute the class memberships associated with the class labels by using 4 and then assign a class label to this pixel by using 5. On the prediction result for all pixels of $\mathcal{P}$, we can obtain the final predictions **pred** in the form of crisp labels. By using the ground truth of all pixels in the training set, we can calculate the Dice coefficient associated with each class label and the average Dice coefficient. The CLPSO runs until it reaches the number of iterations. From the last swarm, we select the candidate $\hat{\mathbb{W}}$ which is associated with the best Dice coefficient as the solution of the problem.

In the classification process, we assign the class label to an unsegmented image $\mathbf{I}$. We first obtain the predictions $\mathbf{P}(\mathbf{I})$ for all pixels of $\mathbf{I}$ by using the EoC $\{\mathbf{h}_k\}_{k=1}^K$. The M class memberships of each pixel then are calculated by using these predictions and the optimal weight $\hat{\mathbb{W}}$ (Step 2-5). The classification rule in 5 is applied to these class memberships of this pixel to give the final prediction. The predictions for all pixels of $\mathbf{I}$ constitute its segmentation result.

Algorithm 2 Compute the Dice coefficient for each weight candidate generated in the CLPSO algorithm

Input: Candidate $\mathbb{W}$, Predictions $\mathcal{P}$

Output: The Dice coefficient associated with $\mathbb{W}$

1: **for** each row $\mathbf{I}_n(i,j)$ of $\mathcal{P}$ **do**
2: **for** $m \leftarrow 1$ to M **do**
3: Compute $CM_m(\mathbf{I}_n(i,j))$ by using 4
4: Assign class label to $\mathbf{I}_n(i,j)$ by using 5
5: Generate **pred**
6: Compute DC_{avg} by 9
7: **return** DC_{avg}

Algorithm 3 Classification process

Input: Unsegmented image $\mathbf{I}$, the optimal weights $\hat{\mathbb{W}}$ and $\{\mathbf{h}_k\}_{k=1}^{K}$

Output: Segmented result for $\mathbf{I}$

1: Obtain the prediction $\mathbf{P}(\mathbf{I})$ by using $\{\mathbf{h}_k\}_{k=1}^{K}$
2: **for** each pixel of $\mathbf{I}$ **do**
3: **for** $m \leftarrow 1$ to M **do**
4: Compute $CM_m(\mathbf{I}(i,j))$ by using $\mathbb{P}_m$ getting from $\mathbf{P}(\mathbf{I})$ and $\hat{\mathbb{W}}_m$ from $\hat{\mathbb{W}}$
5: Assign label to $\mathbf{I}(i,j)$ by using 5
6: **return** Segmented result for $\mathbf{I}$

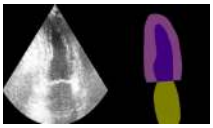

Fig. 1. An image (left) and ground truth (right) of the CAMUS competition.

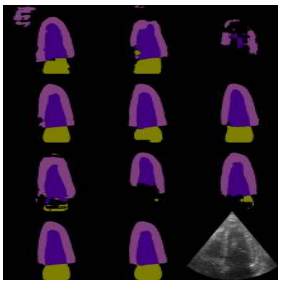

Fig. 2. Example result for datasets of the CAMUS competition. From left to right, top to bottom: UNet-VGG16, LinkNet-VGG16, FPN-VGG16, UNet-ResNet34, LinkNet-ResNet34, FPN-ResNet34, UNet-ResNet101, LinkNet-ResNet101, FPN-ResNet101, Proposed ensemble (6), Proposed ensemble (9), test image (ground truth not available)

IV. Experimental Studies

A. Experimental Settings

Two performance metrics were used for the evaluation of the base segmentation algorithms and the proposed ensemble: Dice coefficient and Mean Absolute Distance (MAD). Dice coefficient, defined in Equation 8, is one of the most popular metrics for medical image segmentation. However, its shortcoming is that it is a measure for total volume difference, without taking into account local discrepancies between contours, which is important in the context of medical image analysis [36]. Therefore, we also used another distance measure between geometrical contours for the evaluation. Let GT_m and PR_m be the set of coordinate vectors of the ground truth contour and prediction contour with respect to class y_m respectively. The MAD for class y_m [37] is defined as follows:

$$MAD_m = \frac{1}{|GT_m| + |PR_m|}\Big(\sum_{gt \in GT_m} \min_{pr \in PR_m} ||gt - pr|| + \sum_{pr \in PR_m} \min_{gt \in GT_m} ||pr - gt||\Big) \quad (11)$$

To evaluate the effectiveness of our proposed ensemble compared to the benchmark algorithms, we participated in the Cardiac Acquisitions for Multi-structure Ultrasound Segmentation (CAMUS) challenge [38], which is a competition for accurate segmentation of 2D echocardiographic images. The datasets provided by the competition consists of clinical exams from 500 patients. For each patient, 2D apical four-chamber and two-chamber cardiographic images and segmentation were recorded at two cardiographic positions, End Diastolic (ED) and End Systolic (ES), making a total of 4 datasets. Three expert cardiologists were involved in the manual segmentation of the datasets. Segmentation ground truth is provided for 450 patients, while the segmentation of the other 50 patients are not publicly available, and participants have to submit the results to a server for evaluation [1]. The datasets have three classes: Left ventricle, Myocardium and Left atrium, with an additional background class. Example images for two-chamber and four-chamber cases and their corresponding ground truths are shown in Figure 1. The evaluation server reports the aggregate results for ED and ES for both four-chamber and two-chamber cases. We reported the best results achieved by the author of this competition and the results of constituent classifiers as benchmark algorithms. For proposed ensemble, we set $T = 5$ for the T-fold cross-validation procedure in the Stacking algorithm. For CLPSO, we set $c = 1.494$ as in [8], and $maxT = 600$, $nPop = 10$. The predictions generation on the training set of one case (e.g. two-chamber ED) took approximately 18 hours using the GPU running in parallel. The optimization for each of the four datasets in the CAMUS competition using the CLPSO meanwhile was run on the CPU and took approximates 26 hours. This can be considered a reasonable time, compared to other similar works such as [49] in which the authors took 61 hours to optimize DNN hyperparameters for medical image segmentation.

B. Influence of Using Different Number of Segmentation Algorithms

We first explored the influence of using different number of segmentation algorithms on the performance of the proposed ensemble. We used the following architectures: UNet [20], LinkNet [21] and Feature Pyramid Network (FPN) [32] with two backbones VGG16 [33] and ResNet34 [34] to obtain the ensemble of 6 segmentation algorithms (denoted by Proposed ensemble (6)). We then used these 3 architectures with

[1] https://www.creatis.insa-lyon.fr/Challenge/camus/scientificInterests.html

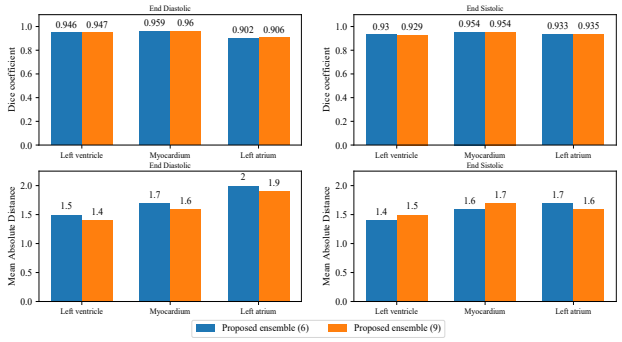

Fig. 3. The performance of proposed ensemble using 6 and 9 segmentation algorithms.

backbone ResNet101 to generate 3 more segmentation algorithms for the ensemble (denoted by Proposed Method (9)). All segmentation algorithms were run for 300 epochs when training classifiers. Figure 3 shows the comparison between the performance of Proposed ensemble (6) and Proposed ensemble (9).

With respect to the Dice coefficient, it can be seen that both ensemble give similar result. For the ED case, Proposed ensemble (6) achieves a Dice coefficient of 0.946 and 0.959 on Left ventricle and Myocardium class respectively, while Proposed ensemble (9) gives a higher result of 0.1% in both classes. For the Left atrium class, Proposed ensemble (6) has a Dice coefficient of 0.902, which is lower than that of Proposed ensemble (9) by 0.4%. Proposed ensemble (6) is slightly better than Proposed ensemble (9) for the ES case, with Dice coefficient of 0.930 and 0.929 respectively. In contrast, Proposed ensemble (6) achieves a Dice coefficient of 0.933 on the Left atrium class which is lower than that of Proposed ensemble (9) by 0.2%. Both ensembles achieve the same result on the Myocardium class at 0.954.

With respect to the MAD, Proposed ensemble (9) achieves better result compared to Proposed ensemble (6) for the ED case by a margin of 0.1 on all three classes (from 1.5 to 1.4 on Left ventricle, 1.7 to 1.6 on Myocardium and 2.0 to 1.9 on Left atrium). This can also be observed for the ES case, Left atrium class (from 1.7 to 1.6). It is observed that adding 3 segmentation algorithms with ResNet101 backbone increases MAD on the two other classes for this case (from 1.4 to 1.5 on Left ventricle and 1.6 to 1.7 on Myocardium).

C. Comparison with Benchmark Segmentation Algorithms

We compared the performance of the Proposed ensemble (9) with the benchmark algorithms. Tables I and II shows the Dice coefficient and MAD measured for the ED Case. It can be seen that the proposed ensemble achieves the best Dice coefficient for all three classes compared to other benchmarks. For the Left ventricle class, the proposed ensemble achieves a score of 0.947 which is slightly higher than that of the second best by UNet-ResNet34 (0.946). Meanwhile, the author's best achieves a score of only 0.936. For the two other classes, Myocardium

TABLE I
DICE COEFFICIENT FOR THE DATASETS OF THE CAMUS COMPETITION, END DIASTOLIC CASE

	Left ventricle	Myocardium	Left atrium
Author's best	0.936	0.956	0.889
UNet-VGG16	0.307	0.3	0.244
UNet-ResNet34	0.946	0.958	0.9
UNet-ResNet101	0.943	0.957	0.892
LinkNet-VGG16	0.203	0.2	0.197
LinkNet-ResNet34	0.942	0.958	0.897
LinkNet-ResNet101	0.887	0.899	0.843
FPN-VGG16	0.354	0.356	0.279
FPN-ResNet34	0.945	0.958	0.899
FPN-ResNet101	0.925	0.938	0.876
Proposed ensemble (9)	**0.947**	**0.96**	**0.906**

TABLE II
MEAN ABSOLUTE DISTANCE FOR DATASETS OF THE CAMUS COMPETITION, END DIASTOLIC CASE

	Left ventricle	Myocardium	Left atrium
Author's best	1.6	1.7	2.2
UNet-VGG16	6.4	6.7	3.4
UNet-ResNet34	1.5	1.7	2
UNet-ResNet101	1.6	1.7	2.2
LinkNet-VGG16	**1.2**	1.9	**1.2**
LinkNet-ResNet34	1.6	1.7	2.1
LinkNet-ResNet101	1.5	1.7	2
FPN-VGG16	1.3	2	2.5
FPN-ResNet34	1.5	1.7	2
FPN-ResNet101	1.5	1.8	2.1
Proposed ensemble (9)	1.4	**1.6**	1.9

and Left atrium, the proposed ensemble has a Dice coefficient of 0.96 and 0.906 respectively, which is better than the second best benchmarks by 0.2% and 0.7% respectively. Most of the contributions to the proposed ensemble are from the segmentation algorithms with ResNet34 and ResNet101 backbone, while the ones having VGG16 backbone achieve a very low Dice coefficient at just around 0.2 and 0.3. However, with the MAD, the proposed ensemble only achieved the best result for the Myocardium class at 1.6, while for Left ventricle and Left atrium it only achieved 1.4 and 1.9 respectively compared to LinkNet-VGG16 which was at 1.2 for both classes. Other benchmarks achieve slightly higher MAD values for all of the three classes. This can be explained by the observation in [36] that in the case where the prediction curvature has a high degree of winding and low similarity compared to the reference curvature, it is possible for measures based on segmentation contours, as opposed to using global information (such as with Dice coefficient) to miscalculate.

The result for ES Case are shown in Tables III and IV. As with the ED case, the proposed ensemble achieved the best Dice coefficient on all three classes, and the benchmarks using VGG16 backbone performed poorly. For the Left ventricle class, the proposed ensemble achieved a score of 0.929 which was higher than the second best (LinkNet-ResNet34) by 0.1%. For the Myocardium class, the proposed ensemble obtained the same Dice coefficient as the second best benchmark (LinkNet-ResNet34) at 0.954. UNet-ResNet34 and FPN-ResNet34 also achieved slightly lower scores (0.952 and 0.953 respectively) while the other benchmarks obtained lower scores from 0.93

TABLE III
DICE COEFFICIENT FOR DATASETS OF THE CAMUS COMPETITION, END SYSTOLIC CASE

	Left ventricle	Myocardium	Left atrium
Author's best	0.913	0.946	0.918
UNet-VGG16	0.295	0.305	0.244
UNet-ResNet34	0.925	0.952	0.927
UNet-ResNet101	0.923	0.949	0.918
LinkNet-VGG16	0.106	0.113	0.119
LinkNet-ResNet34	0.928	**0.954**	0.922
LinkNet-ResNet101	0.871	0.894	0.868
FPN-VGG16	0.317	0.317	0.241
FPN-ResNet34	0.927	0.953	0.926
FPN-ResNet101	0.905	0.93	0.888
Proposed ensemble (9)	**0.929**	**0.954**	**0.935**

TABLE IV
MEAN ABSOLUTE DISTANCE FOR DATASETS OF THE CAMUS COMPETITION, END SYSTOLIC CASE

	Left ventricle	Myocardium	Left atrium
Author's best	1.7	1.9	2
UNet-VGG16	2.1	4.8	3.1
UNet-ResNet34	1.6	**1.7**	1.7
UNet-ResNet101	1.6	**1.7**	1.8
LinkNet-VGG16	3.6	4.1	3.6
LinkNet-ResNet34	**1.5**	**1.7**	1.9
LinkNet-ResNet101	**1.5**	**1.7**	1.7
FPN-VGG16	1.7	3	3.3
FPN-ResNet34	**1.5**	**1.7**	1.8
FPN-ResNet101	**1.5**	1.8	2
Proposed ensemble (9)	**1.5**	**1.7**	**1.6**

(FPN-ResNet101) to 0.49 (UNet-ResNet101). The proposed ensemble achieved for the Left atrium class a Dice score of 0.935, which was higher than the second best (UNet-ResNet34) by a margin of 0.6%. With respect to MAD, the proposed ensemble only achieved better score on the Left atrium class at 1.6, which was better than the second best (UNet-ResNet34 and LinkNet-ResNet101) by a difference of 0.1. For the Left ventricle and the Myocardium class, the proposed ensemble achieved a score of 1.5 and 1.7 respectively, which was the same as with several benchmarks. It should be noted that even though the improvement was not very high, this was the average result across 50 patients, while there are cases in which there was noticeable improvement which is very important in clinical situations. Table V shows the comparison of Dice and MAD result for patient 19 between UNet-ResNet101 and Proposed ensemble (9) on Left atrium class. It can be seen that for this patient, UNet-ResNet101 has a ED Dice score of 0.903, compared to 0.926 by Proposed ensemble (9), which was an increase of more than 2%. Similarly, there is an improvement of 1% for ES Dice score (0.942 to 0.952). For MAD score, the proposed ensemble has a better score by a margin of around 0.3.

Figure 2 shows an example of prediction made by the benchmarks and the proposed ensemble. It can be seen that FPN-VGG16 (first row, second column) failed to make a correct prediction, while LinkNet-VGG16 did not segment the bottom left part of the Myocardium, and made mistake on a part of the Left ventricle for Myocardium. UNet-VGG16 wrongly predicted an empty part in the top left as Myocardium, and leaves a small hole in the Left atrium contour. For the benchmarks using ResNet34 backbone (second row), UNet-ResNet34 obtained an unsegmented hole in the left of the Myocardium contour, while LinkNet-ResNet34 predicted correctly the Left ventricle and Myocardium class but failed to segment a lower part in the left of the Left atrium class. On the other hand, FPN-ResNet34 obtained an unsegmented area in the area between the Left ventricle and the Myocardium. The benchmarks using ResNet101 backbone (third row) failed to segment the Left atrium class altogether. In contrast, both Proposed ensemble (6) and Proposed ensemble (9) improved on the base segmentation algorithms to achieve the better segmentation result. Table VI shows the optimal weights found by CLPSO for the two-chamber ED case. It can be seen that overall the ResNet-based algorithms are assigned a higher weights compared to the VGG16-based algorithms, however there are cases where the VGG16-based algorithms are assigned relatively high weights. For example, with respect to the Left ventricle class, LinkNet-VGG16 and FPN-VGG16 were assigned a weight of 0.766 and 0.675 respectively. This shows that the weights of the proposed ensemble are not biased towards well-performing methods. Instead, all the constituent segmentation algorithms contribute to the ensemble.

TABLE V
COMPARISON OF DICE AND MAD RESULT FOR PATIENT 19 BETWEEN UNET-RESNET101 AND PROPOSED ENSEMBLE (9), LEFT ATRIUM CLASS

	ED Dice	ES Dice	ED MAD	ES MAD
UNet-ResNet101	0.903	0.942	1.9	1.5
Proposed ensemble (9)	0.926	0.952	1.4	1.2

TABLE VI
OPTIMAL WEIGHTS FOUND BY CLPSO FOR THE TWO-CHAMBER ED CASE

	Left ventricle	Myocardium	Left atrium	Background
UNet-VGG16	0.469	0.061	0.390	0.010
UNet-ResNet34	0.358	0.816	0.982	0.267
UNet-ResNet101	0.362	0.640	0.874	0.125
LinkNet-VGG16	0.766	0.126	0.449	0.151
LinkNet-ResNet34	0.815	0.705	0.473	0.682
LinkNet-ResNet101	0.232	0.761	0.708	0.283
FPN-VGG16	0.675	0.368	0.499	0.004
FPN-ResNet34	0.573	0.391	0.970	0.156
FPN-ResNet101	0.891	0.506	0.771	0.321

V. CONCLUSION

In this paper, we presented a novel weighted ensemble of deep learning models for the problem of medical image segmentation. The probability predictions by the segmentation algorithms are combined based on weighted combining for a final prediction. Comprehensive Learning Particle Swarm Optimization (CLPSO), a swarm intelligence algorithm, was used to find the combining weights which gave the best fitness value over a five-fold cross-validation procedure. Dice coefficient, a popular metrics for medical image segmentation, was used as the fitness criteria. Our result on the datasets of CAMUS competition shows that the proposed ensemble achieves an overall improvement compared to several benchmark algorithms.

ACKNOWLEDGEMENT

Funding was provided by the Newton Fund Institutional Links program, project 527639907, in collaboration with Universidad Nacional Autonoma de Mexico (UNAM), granted by The British Council, UK, and Secretaría de Tecnología e Innovación (SECTEI), Mexico City, Mexico.

REFERENCES

[1] T.T. Nguyen, M.T. Dang, A.W.C. Liew et al., A weighted multiple classifier framework based on random projection, Information Sciences, 490 (2019), pp. 36-58.

[2] T.T. Nguyen, T.T.T. Nguyen, X.C. Pham et al., A novel combining classifier method based on Variational Inference, Pattern Recognition. 49 (2016), pp. 198-212.

[3] J. Kittler, M. Hatef, R.P.W. Duin et al., On Combining Classifiers, IEEE IEEE Trans. Pattern Anal. Mach. Intell. 20(3) (1998), pp. 226-239.

[4] Y. Zhang, S. Burer, W. Nick Street, Ensemble pruning via semidefinite programming, J. Mach. Learn. Res. 7 (2006), pp. 1315-1338.

[5] M. U. Sen, H. Erdogan, Linear classifier combination and selection using group sparse regularization and hinge loss, Pattern Recognition Letters. 34(3) (2013), pp. 265–274.

[6] J. Kennedy, R. Eberhart, Particle Swarm Optimization, in: Proceedings of ICNN'95, 1995, pp. 1942-1948.

[7] Y. Zhang, S. Wang, G. Ji, A Comprehensive Survey on Particle Swarm Optimization Algorithm and Its Applications, Mathematical Problems in Engineering. 2015.

[8] J.J. Liang, A.K. Qin, P. N. Suganthan et al., Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions, IEEE Trans Evo Comp. 10 (3) (2006), pp. 281-295.

[9] B. Tran, B. Xue, M. Zhang, Variable-Length Particle Swarm Optimization for Feature Selection on High-Dimensional Classification, IEEE Trans Evo Comp. 23 (3) (2019), pp. 473-487.

[10] X. Yu, X. Zhang, Enhanced comprehensive learning particle swarm optimization, Applied Mathematics and Computation 242 (2014), pp. 265-276.

[11] N. Lynn, P.N. Suganthan, Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation, Swarm and Evolutionary Computation. 24 (2015), pp. 11-24.

[12] T.T. Nguyen, M.D. Dang, V.A. Baghel et al., Evolving interval-based representation for heterogeneous classifier fusion, Knowledge Based Systems, 2020.

[13] T.T. Nguyen, A.V. Luong, M.T. Dang et al., Evolving an Optimal Decision Template for Combining Classifiers, in Proceedings of ICONIP, 2019, pp. 608-620.

[14] A. Krizhevsky, S. Ilya, H. Geoffrey, ImageNet classification with deep convolutional neural networks, in Commun. ACM 60, 2017, pp. 84-90.

[15] Shen, D., Wu, G., Suk, H.I, Deep learning in medical image analysis, in Annu. Rev. Biomed. Eng. 19, 2017, pp. 221–248.

[16] Guo, Y., Gao, Y., Shen, D., Deformable MR Prostate Segmentation via Deep Feature Learning and Sparse Patch Matching, in IEEE Trans Med Imaging. 2016, pp. 1077-89.

[17] Shin, HC., Orton, MR., Collins, DJ. et al., Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data, in IEEE Trans Pattern Anal Mach Intell. 2013, 35(8), pp. 1930-43.

[18] Kleesiek, J., Urban, G., Hubert, A. et al., Deep MRI brain extraction: A 3D convolutional neural network for skull stripping, in Neuroimage. 2016, 129, pp. 460-469.

[19] Litjens, G., Kooi, T., Bejnordi, BE. et al., A survey on deep learning in medical image analysis, in Med Image Anal. 2017, 42:pp.60-88.

[20] Ronneberger O., Fischer P., Brox T., U-Net: Convolutional Networks for Biomedical Image Segmentation, in MICCAI, 2015.

[21] A. Chaurasia, E. Culurciello, LinkNet: Exploiting encoder representations for efficient semantic segmentation, in IEEE Visual Communications and Image Processing, 2017, pp. 1-4.

[22] F. Milletari, N. Navab, S. Ahmadi, V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation, in Fourth International Conference on 3D Vision, 2016, pp. 565-571.

[23] Drozdzal, M., Vorontsov, E., Chartrand, G. et al., The importance of skip connections in biomedical image segmentation, in Proceedings of Deep Learning in Medical Image Analysis. 2016, vol. 10008, pp. 179–187.

[24] Andermatt, S., Pezold, S., Cattin, P., Multi-dimensional gated recurrent units for the segmentation of biomedical 3D-data. in Proceedings of Deep Learning in Medical Image Analysis, 2016, vol. 10008, pp. 142–151.

[25] Ciresan, D., Giusti, A., Gambardella, L.M. et al., Deep neural networks segment neuronal membranes in electron microscopyimages, in Proceedings of NIPS, 2016, pp. 2843–2851.

[26] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in IEEE CVPR, 2015, pp. 3431-3440.

[27] Korez, R., Likar, B., Pernus, F. et al.. Model-based segmentation of vertebral bodies from MR images with 3D CNNs, in Proceedings of MICCAI, 2016, pp. 433–441.

[28] Shakeri, M., Tsogkas, S., Ferrante, E. et al., Sub-cortical brain structure segmentation using F-CNNs, in Proceedings of the IEEE International Symposium on Biomedical Imaging, 2016, pp. 269-272.

[29] A. G. C. Pacheco, T. Trappenberg, R. A. Krohling, Learning dynamic weights for an ensemble of deep models applied to medical imaging classification, in IJCNN, 2020, pp. 1-8.

[30] D. Guo, Y. Pei, K. Zheng et al., "Degraded Image Semantic Segmentation With Dense-Gram Networks," in IEEE Trans Image Process, vol. 29, 2020, pp. 782-795.

[31] Jingru Yi, Pengxiang Wu, Menglin Jiang et al., Attentive neural cell instance segmentation, Med Image Anal, vol. 55, 2019, pp. 228-240.

[32] T. Lin, P. Dollar, R. Girshick et al., Feature Pyramid Networks for Object Detection, in IEEE CVPR, 2017, pp. 936-944.

[33] S. Liu, W. Deng, Very deep convolutional neural network based image classification using small training sample size, in 3rd IAPR Asian Conference on Pattern Recognition, 2015, pp. 730-734.

[34] K. He, X. Zhang, S. Ren et al., Deep Residual Learning for Image Recognition, in IEEE CVPR, 2016, pp. 770-778.

[35] Q. Liu, X. Tang, D. Guo et al., Multi-class Gradient Harmonized Dice Loss with Application to Knee MR Image Segmentation, in MICCAI, 2019, pp. 86–94.

[36] H. Kim, S. Park, S. Lo et al., Bidirectional local distance measure for comparing segmentations, in Medical Physics, 2019, vol. 39, no. 11, pp. 6779–6790.

[37] A. Taha, A. Hanbury, Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool, in BMC Medical Imaging, vol. 15, Aug. 2015.

[38] S. Leclerc, E. Smistad, J. Pedrosa, et al., Deep Learning for Segmentation Using an Open Large-Scale Dataset in 2D Echocardiography, in IEEE Trans Med Imaging, 2019, vol. 38, no. 9, pp. 2198–2210.

[39] Farag A.A., Ahmed M.N., El-Baz A. et al., Advanced Segmentation Techniques. In: Handbook of Biomedical Image Analysis. International Topics in Biomedical Engineering. Springer, 2015.

[40] Elnakib A., Gimel'farb G., Suri J.S. et al., Medical Image Segmentation: A Brief Survey. In: Multi Modality State-of-the-Art Medical Image Segmentation and Registration Methodologies. Springer, 2011.

[41] Quoc V. Le, Jiquan Ngiam, Adam Coates et al., On optimization methods for deep learning. In Proceedings of ICML, 2011, pp. 265–272.

[42] Pham D.L., Xu C, Prince J.L, Current methods in medical image segmentation. Annual Review of Biomedical Engineering, 2000.

[43] Kingma, Diederik P., and Ba, J., Adam: A Method for Stochastic Optimization. ICML, 2015.

[44] Ashia C. Wilson, Rebecca Roelofs, Mitchell Stern et al., The marginal value of adaptive gradient methods in machine learning. In Proceedings of NIPS, 2017, pp. 4151–4161.

[45] Z. Hu, Y. Bao, T. Xiong, Comprehensive learning particle swarm optimization based memetic algorithm for model selection in short-term load forecasting using support vector regression. In Appl. Soft Comput., 2014, pp. 15–25.

[46] K.Mahadevan, P.S. Kannan, Comprehensive learning particle swarm optimization for reactive power dispatch. In Appl. Soft Comput., 2010, pp. 641-652.

[47] H. Ali, F.A. Khan, Attributed multi-objective comprehensive learning particle swarm optimization for optimal security of networks. In Appl. Soft Comput., 2013, pp. 3903–3921.

[48] 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. 15, 1, 2014, pp. 3133–3181.

[49] Baldeon Calisto, M., Lai-Yuen, S., AdaResU-Net: Multiobjective adaptive convolutional neural network for medical image segmentationaldeon-Calisto and Susana K Lai-Yuen. In Neurocomputing 2020, pp. 325-340.