Two layer ensemble of deep learning models for medical image segmentation.

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

2021

Two layer Ensemble of Deep Learning Models for Medical Image Segmentation

Truong Dang, Tien Thanh Nguyen, John McCall, Eyad Elyan, Carlos Francisco Moreno-García
School of Computing Robert Gordon University, Aberdeen, UK

***Abstract*—In recent years, deep learning has rapidly become a method of choice for the segmentation of medical images. Deep Neural Network (DNN) architectures such as UNet have achieved state-of-the-art results on many medical datasets. To further improve the performance in the segmentation task, we develop an ensemble system which combines various deep learning architectures. We propose a two-layer ensemble of deep learning models for the segmentation of medical images. The prediction for each training image pixel made by each model in the first layer is used as the augmented data of the training image for the second layer of the ensemble. The prediction of the second layer is then combined by using a weights-based scheme in which each model contributes differently to the combined result. The weights are found by solving linear regression problems. Experiments conducted on two popular medical datasets namely CAMUS and Kvasir-SEG show that the proposed method achieves better results concerning two performance metrics (Dice Coefficient and Hausdorff distance) compared to some well-known benchmark algorithms.**

I. Introduction

Segmentation is the process of partitioning an image into multiple segments to locate objects and boundaries. Before the rise of Deep Neural Networks (DNN), most of the successful segmentation algorithms used hand-crafted features combined with a machine learning classifier such as Random Forest [1] or Support Vector Machine [2]. Even though subsequent research have achieved noticeable improvements by incorporating richer context information [3] or by applying structured prediction techniques [4], [5], the performance of these systems remained limited because the hand-crafted features are not representative enough for real-world usage. With the success of DNNs in image classification in 2012 [6], researchers began to apply this new architecture to segmentation. Some notable results in this direction include Fully Connected Networks (FCN) [7] and SegNet [8]. Applying deep learning techniques to medical imaging has brought many successes, such as the introduction of a novel architecture called Unet and successfully applied it to the segmentation of neuronal structures in electron microscopic stacks [9]. This network continues to be widely used for segmentation. Another notable example is in [10] which used T1-weighted, T2-weighted, and fractional anisotropy image patches of 13x13 in size as input to a Convolutional Neural Network (CNN) for segmentation of infant brains which are considered to be much more difficult than adult brains. This approach outperforms other commonly used segmentation algorithms when tested on a set of manually segmented isointense stage brain images. Deep learning methods are highly effective for cases when the dataset is large. For example, the first success in deep learning was a network trained on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset [6], which contained 1 million annotated images. However, medical image datasets are much smaller, usually about 1,000 images[11]. This creates an important challenge for creating deep medical models which are robust against overfitting. Another problem is that popular optimizers for training deep neural networks such as Stochastic Gradient Descent (SGD) generally require much manual tuning of optimization parameters [12]. Despite the fact that there has been some alternative methods which require less parameter tuning, such as Adam [13], these methods do not generalize as well as SGD [14]. The manual parameter tuning causes a challenge in selecting suitable deep models for a specific problem. Therefore, because medical image analysis requires reliable predictions from automated systems due to its critical nature, it is essential to leverage the strong points of multiple segmentation algorithms to improve on the final results.

Ensemble learning is a popular technique in which multiple learners are combined to make a collaborated decision. The key challenge is to build an effective ensemble method to combine the results of segmentation algorithms. The paper is organized as follows. In section 2, we briefly review the existing approaches relating to segmentation in medical image analysis and the ensemble learning. In section 3, we propose a novel two-layer ensemble method to combine the results of segmentation algorithms. Because segmentation gives a pixel-level output, the prediction results by the segmentation algorithms are concatenated with the original image as input to segmentation algorithms in the second layer. Dice Coefficient and Hausdorff distance are used as the evaluation metrics. The details of experimental studies on two public datasets are described in section 4. Finally, the conclusion is given in section 5.

II. Background and Related Work

A. Semantic segmentation in medical image analysis

With the success of [6] in applying deep Convolutional Neural Network (CNN) to the problem of image classification, deep learning has become the most popular approach in computer vision. Since then, many notable deep architectures have been proposed to solve vision problems. For example, VGG16 [15] was a deep CNN for image classification using a stack of convolution layers with small receptive fields in the first layers instead of few layers with big receptive fields

like previous models. This allows the model to have much fewer parameters and more non-linearity, which makes the decision function more discriminative and the model easier to train. VGG16 managed to achieve a top-5 accuracy of 92.7% on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)-2013 dataset. Another notable model is ResNet [16], which was motivated by the problem of training a really deep architecture. The network uses shortcut connections in order to perform identity mapping, i.e. instead of learning a function, the layers having shortcut connections learn the residual mapping. This allows Resnet to have a very deep network at 152 layers while achieving 96.4% accuracy on the ILSVRC-2016 competition.

Generally, deep image classification models are trained on large datasets, such as ImageNet [17] which have around 1 million images. However, in the problem of semantic segmentation, in which a model must predict the class of each pixel in the image, the scale of available datasets is not as large as in image classification [18]. To overcome this limitation, practitioners usually use a pre-trained classification network and finetune it for segmentation. Most deep learning based semantic segmentation architectures are inspired by Fully Convolutional Network (FCN) [7], which creates a segmentation network by using an existing classification network and replace the fully connected layers with convolutional ones to output spatial maps instead of classification scores. Those maps are then upsampled to produce dense pixel-level output. This architecture is considered the cornerstone of deep learning applied to semantic segmentation [18]. Another notable example is DeepLab [19] which makes use of Conditional Random Fields (CRF) [20] as a post-processing step for the refinement of the segmentation result. The proposed architecture models each pixel as a node in the random field and employs a fully connected factor graph in which one pairwise term is used for each pixel pair irrespective of their distance. This allows the model to incorporate both short-range and long-range information into account, facilitating the restoration of detailed structures in the segmentation process that was lost due to the spatial invariance of CNN.

Segmentation is considered one of the most essential medical imaging process as it extracts the region of interest (ROI) which is then used in clinical applications. Therefore, it has seen the widest variety of proposed methodology, including deep architectures specifically designed to tackle problems in medical image analysis. A notable example is UNet [9] which consists of a contracting path and an expanding path designed symmetrically. To help with localization, high resolution features from the contracting path are combined with the upsampled output. An important difference of UNet compared to previous architectures is that the upsampling part also has a large number of feature, channels, which allow the network to propagate context information to higher resolution layers. The network does not have any fully connected layer and therefore can be trained on images of arbitrary size via an overlap-tile strategy. In recent years, Recurrent Neural Networks (RNNs) have also become widely used for medical image segmentation. For example, in [21] a spatial clockwork RNN was used to segment perimysium in histopathology images. The authors applied the RNN four times in different orientations in order to incorporate bidirectional information from left/top and right/bottom neighbors. For 3D brain segmentation, [22] trained a 3D-CNN by using mini-batches of multiple cubes, whose size was larger than the input size. Their proposed model could take an arbitrary-sized 3D patch as input and would output a block of predictions per input, which is similar to FCN. Over four different brain segmentation datasets, their proposed method achieved the highest average specificity measure, with no significant loss in sensitivity. Some researchers have also used graphical models such as Conditional Random Fields as a post-processing step to refine the segmentation results [23].

B. Ensemble learning

Ensemble learning is a popular approach in machine learning for combining a collection of classifiers for the collaborative decision. Designing an ensemble system requires two stages, namely ensemble generation and ensemble integration. In the ensemble generation, multiple classifiers are generated by using either a homogeneous strategy (training a learning algorithm on multiple training sets generated from the original training data) [24], [25] or a heterogeneous strategy (training different learning algorithms on the original training data) [26], [27]. A combining method is then used to aggregate the predictions of the constituent classifiers in the ensemble integration stage to obtain the collaborated prediction. Several top-performing methods for classification have been reported including Random Forest [28], XgBoost [29], and Rotation Forest [30].

Recently, there is increasing interest in the ensemble generation inspired by the success of DNNs. Instead of using only one layer like in traditional ensemble models, the ensemble systems were made to train deeply through multiple layers. The first deep ensemble system was proposed by Zhou and Feng [31] (called gcForest), containing multiple layers of two Completely-Random Tree Forests and two Random Forests in each layer. Each forest in a layer outputs a class vector, which is then concatenated to the original data as the input data to the next layer. Utkin et al. [32] proposed a weighted average approach for gcForest by associating each tree with a weighted vector for its class distribution vector. The optimal weight vectors of each trees in one layer are found by minimizing the distance between the class label vector in a binary encoding scheme and the weighted prediction vector of this forest. The authors proposed to set only a weight vector for each group in order to reduce the computational overhead. Nguyen et al. [33] proposed MULES, a deep ensemble system with classifier and feature selection in each layer. The optimal configuration of each layer is found by using a bi-objective optimization problem in which the two objectives to be maximized are classification accuracy and diversity of the ensemble in each layer. Qi et al. [34] introduced a deep ensemble model in which each layer consists of an ensemble of Support Vector

Machine (SVM) classifiers [35]. The model parameters, such as the kernel functions of the SVM classifiers, the number of classifiers, and the weights of the features are found by AdaBoost [36].

III. Proposed Method

Our proposed method is inspired by multi-layer ensemble learning architectures, in which the segmentation algorithms in one layer train the segmentation models of this layer on the new training data generated by the preceding layer [31]. Applied to segmentation of medical images, this facilitates the successive refinement of segmentation results through each layer. It is recognized that the most successful segmentation algorithms in recent years have been based on DNNs [37], and even though deep learning models can be trained in parallel using GPU, a multi-layer ensemble model of deep learning-based segmentation algorithms would require a lot of computational resources. Therefore, an important question arises: How many layers should a deep ensemble model extend? [33] showed that on some datasets, the number of layers of multi-layer ensemble obtained was 2 or 3 only. Based on this observation, we introduce a novel two-layer ensemble model for segmentation of medical images. Figure 1 shows the high-level overview of our proposed method.

A. Two-layer ensemble for segmentation

Let $\mathbf{D} = \{\mathbf{I}_n, \mathbf{Y}_n\}_{n=1}^N$ be the training set where N is the number of images, $\mathbf{I}_n$ is an input image of size (W, H, C) with H being the image height, W the image width, and C is the number of channels ($C = 1$ for grayscale, $C = 3$ for color image). The mask $\mathbf{Y}_n$ is also an image of size (W, H), with each entry $\mathbf{Y}_n(i,j)(i = 1, ..., W; j = 1, ..., H)$ showing which group the pixel $\mathbf{I}_n(i,j)$ belongs to, i.e $\mathbf{Y}_n(i,j) \in \mathcal{Y}$, where $\mathcal{Y} = \{y_m\}, m = 1, ..., M$ is the set of all classes and M is the number of classes.

We aim to learn a hypothesis $\mathbf{h} : \mathbf{I}_n \rightarrow \mathbf{Y}_n$ (i.e segmentation model) to approximate the unknown relationship between each image and its corresponding mask, and then use this hypothesis to assign a label for each unsegmented image. We also denote $\{\mathcal{K}_k\}_{k=1}^K$ as the set of K segmentation algorithms. Each segmentation algorithm $\mathcal{K}_k$ learns on $\mathbf{D}$ to obtain a trained segmentation model $\mathbf{h}_k$. In ensemble learning, we train segmentation algorithms $\{\mathcal{K}_k\}_{k=1}^K$ on $\mathbf{D}$ to get K trained segmentation models $\{\mathbf{h}_k\}_{k=1}^K$

In the next step, we generate the training data for the second layer of ensemble. Based on the results of [33] and the stacking generalization model [26], we propose a two-layer deep ensemble architecture for segmentation in medical image analysis (Figure 1). Firstly, the training set $\mathbf{D}$ is divided into T disjoint parts $\{\mathbf{D}_1, \mathbf{D}_2, ..., \mathbf{D}_T\}$, where $\mathbf{D} = \mathbf{D}_1 \cup \mathbf{D}_2 \cup ... \cup \mathbf{D}_T, \mathbf{D}_{t_1} \cap \mathbf{D}_{t_2} = \emptyset (t_1, t_2 = 1, ..., T, t_1 \neq t_2)$. Then for each part $\mathbf{D}_t (t = 1, ..., T)$, the segmentation algorithms $\{\mathcal{K}_k\}_{k=1}^K$ will learn on its complementary $\mathbf{D} \setminus \mathbf{D}_t$ to obtain segmentation models $\mathbf{h}_{k,t}$. The images in $\mathbf{D}_t$ are then segmented by using these segmentation models. Let $P_k(y_m|\mathbf{I}_n(i,j))$ be probability prediction that $\mathbf{h}_{k,t}$ assigns pixel $\mathbf{I}_n(i,j)$ to be in class y_m. The prediction of $\mathbf{h}_{k,t}$ showing the probability all pixels of image $\mathbf{I}_n$ belonged to class y_m is given by a matrix:

$$\mathbf{P_k}(y_m|\mathbf{I}_n) = \begin{bmatrix} P_k(y_m|\mathbf{I}_n(1,1)) & P_k(y_m|\mathbf{I}_n(1,2)) & \cdots & P_k(y_m|\mathbf{I}_n(1,H)) \\ \cdots & \cdots & \cdots & \cdots \\ P_k(y_m|\mathbf{I}_n(W,1)) & P_k(y_m|\mathbf{I}_n(W,2)) & \cdots & P_k(y_m|\mathbf{I}_n(W,H)) \end{bmatrix} \quad (1)$$

For each image $\mathbf{I}_n$, there will be $M \times K$ prediction matrices $\mathbf{P_k}(y_m|\mathbf{I}_n)$ illustrated in Figure 2. In this study, we propose augment the training data for the second layer of ensemble by concatenating these $M \times K$ prediction matrices to the original training images to create new images $\mathbf{I}_n^*$. The prediction matrix $\{\mathbf{P_k}(y_m|\mathbf{I}_n)\}$ serves as an additional channel of the original image $\mathbf{I}_n$. In total, the new images $\mathbf{I}_n^*$ will have $C + M \times K$ channels:

$$\mathbf{I}_n^* = \mathbf{I}_n \cup \{\mathbf{P_k}(y_m|\mathbf{I}_n)\}, k = 1, ..., K, m = 1, ..., M \quad (2)$$

The new training data for the second layer of ensemble will be given as follows:

$$\mathbf{D}^* = \{\mathbf{I}_n^*, \mathbf{Y}_n\}_{n=1}^N \quad (3)$$

For second layer of the ensemble, we train $\{\mathcal{K}_k\}_{k=1}^K$ on $\mathbf{D}^*$ to get trained segmentation models $\{\mathbf{h}_k^*\}_{k=1}^K$. We then need to train a combiner $\mathbf{C}$ to combine the trained models $\hat{\mathbf{h}} = \mathbf{C}(\{\mathbf{h}_k^*\}_{k=1}^K)$ for final decision making. The training of combiner will conduct on the predictions for all pixels of training images in $\mathbf{D}^*$. Once again, the new training data $\mathbf{D}^*$ is divided into disjoint parts $\{\mathbf{D}_1^*, \mathbf{D}_2^*, ..., \mathbf{D}_T^*\}$. Then for each part $\mathbf{D}_t^* (t = 1, ..., T)$, the segmentation algorithms $\{\mathcal{K}_k\}_{k=1}^K$ will learn on $\mathbf{D}^* \setminus \mathbf{D}_t^*$ to obtain segmentation models $\mathbf{h}_{k,t}^*$. These models will then predict on $\mathbf{D}_t^*$. The second-layer probability prediction for all images in $\mathbf{D}^*$ is given as follows:

$$\mathbf{L}^* = \begin{bmatrix} P_1(y_1|\mathbf{I}_1^*(1,1)) & P_1(y_2|\mathbf{I}_1^*(1,1)) & \cdots & P_K(y_M|\mathbf{I}_1^*(1,1)) \\ P_1(y_1|\mathbf{I}_1^*(1,2)) & P_1(y_2|\mathbf{I}_1^*(1,2)) & \cdots & P_K(y_M|\mathbf{I}_1^*(1,2)) \\ \cdots & \cdots & \cdots & \cdots \\ P_1(y_1|\mathbf{I}_1^*(W,H)) & P_1(y_2|\mathbf{I}_1^*(W,H)) & \cdots & P_K(y_M|\mathbf{I}_1^*(W,H)) \\ P_1(y_1|\mathbf{I}_2^*(1,1)) & P_1(y_2|\mathbf{I}_2^*(1,1)) & \cdots & P_K(y_M|\mathbf{I}_2^*(1,1)) \\ \cdots & \cdots & \cdots & \cdots \\ P_1(y_1|\mathbf{I}_2^*(W,H)) & P_1(y_2|\mathbf{I}_2^*(W,H)) & \cdots & P_K(y_M|\mathbf{I}_2^*(W,H)) \\ \cdots & \cdots & \cdots & \cdots \\ P_1(y_1|\mathbf{I}_N^*(W,H)) & P_1(y_2|\mathbf{I}_N^*(W,H)) & \cdots & P_K(y_M|\mathbf{I}_N^*(W,H)) \end{bmatrix} \quad (4)$$

Normally, a learning algorithm trains the combiner on $\mathbf{L}^*$ with given labels of each pixel to combine the prediction of segmentation models for the final prediction. It is noted that each row of $\mathbf{L}^*$ is the probability predictions by K segmentation models on a pixel of each training image. Therefore $\mathbf{L}^*$ will be a matrix of $N \times W \times H$ rows and $M \times K$ columns. With a large training set and large image sizes, the size of $\mathbf{L}^*$ will be very large. For example, on Kvasir-SEG dataset of 800 training images with image size of $(640, 544)$, the matrix $\mathbf{L}^*$ will have $800 * 640 * 544 = 278528000$ rows. The large size of $\mathbf{L}^*$ causes a challenge for conventional machine learning algorithms to train the combiner on all data at once. In this paper, we use a weight-based combining method on the segmentation algorithms $\{\mathbf{h}_k^*\}_{k=1}^K$, in which each segmentation algorithm has its own weight in the combiner. The weights are found via an optimization method. This approach is practical to train the combiner on the whole $\mathbf{L}^*$ at once.

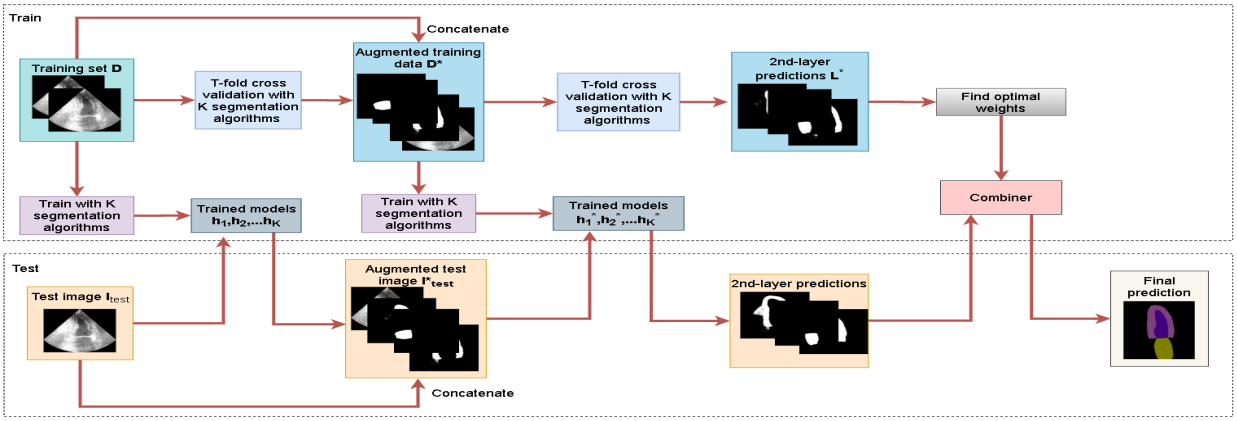

Fig. 1. High-level overview of the proposed method.

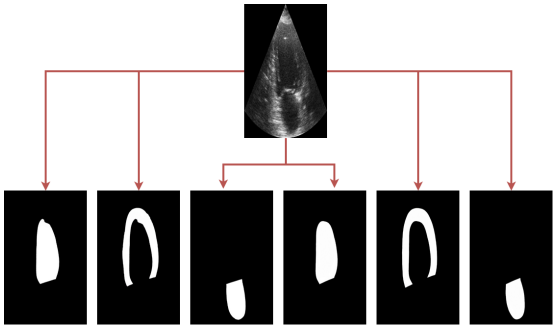

Fig. 2. Example of prediction results on CAMUS dataset. Top: Original image. Bottom is the predictions for Left ventricle, Myocardium and Left atrium classes, made by UNet and LinkNet with backbones ResNet34 and VGG16, respectively. The result has been multiplied by 255 for visualization.

B. Combining method

Let $\mathbb{W} = \{w_{k,m}\}$ be the weight matrix, in which $w_{k,m}$ is the weight associated with the segmentation model $\mathbf{h}_k^*$ and class $y_m (k = 1, ..., K, m = 1, ..., M)$. Since the class labels of the training observations are known in advance, the weights $\mathbb{W}$ can be obtained by exploring the relationship between the second-layer probability predictions in $\mathbf{L}^*$ and the class labels of the training pixels. The weight matrix is found by minimizing the difference between the prediction for pixel $\mathbf{I}_n(i,j)$ and its true class label. From the second-layer probability prediction matrix $\mathbf{L}^*$, we extract the probabilities associated with class y_m to create matrix of size $(N \times W \times H, K)$:

$$\mathbf{L}_m^* = \begin{bmatrix} P_1(y_m|\mathbf{I}_1^*(1,1)) & P_2(y_m|\mathbf{I}_1^*(1,1)) & \cdots & P_K(y_m|\mathbf{I}_1^*(1,1)) \\ P_1(y_m|\mathbf{I}_1^*(1,2)) & P_2(y_m|\mathbf{I}_1^*(1,2)) & \cdots & P_K(y_m|\mathbf{I}_1^*(1,2)) \\ \cdots & \cdots & \cdots & \cdots \\ P_1(y_m|\mathbf{I}_1^*(W,H)) & P_2(y_m|\mathbf{I}_1^*(W,H)) & \cdots & P_K(y_m|\mathbf{I}_1^*(W,H)) \\ P_1(y_m|\mathbf{I}_2^*(1,1)) & P_2(y_m|\mathbf{I}_2^*(1,1)) & \cdots & P_K(y_m|\mathbf{I}_2^*(1,1)) \\ \cdots & \cdots & \cdots & \cdots \\ P_1(y_m|\mathbf{I}_2^*(W,H)) & P_2(y_m|\mathbf{I}_2^*(W,H)) & \cdots & P_K(y_m|\mathbf{I}_2^*(W,H)) \\ \cdots & \cdots & \cdots & \cdots \\ P_1(y_m|\mathbf{I}_N^*(W,H)) & P_2(y_m|\mathbf{I}_N^*(W,H)) & \cdots & P_K(y_m|\mathbf{I}_N^*(W,H)) \end{bmatrix} \quad (5)$$

We also define crisp label vector having size $(N \times W \times H, 1)$ associated with class y_m as follows:

$$\mathbb{Y}_m = \begin{bmatrix} \mathbb{I}[\mathbf{Y}_1(1,1) = y_m] \\ \cdots \\ \mathbb{I}[\mathbf{Y}_1(W,H) = y_m] \\ \cdots \\ \mathbb{I}[\mathbf{Y}_n(1,1) = y_m] \\ \cdots \\ \mathbb{I}[\mathbf{Y}_n(W,H) = y_m] \end{bmatrix} \quad (6)$$

where $\mathbb{I}[.]$ is the indicator function. The weight vector $\mathbb{W}_m = \{w_{k,m}\}, k = 1, ..., K$ of size $(K, 1)$ for class y_m is then found by solving a linear regression problem:

$$\min_{\mathbb{W}_m} ||\mathbf{L}_m^* \mathbb{W}_m - \mathbb{Y}_m||_2 \quad (7)$$

$\mathbb{W}_m$ can be imposed with different constraints, such as Non-Negative Least Squares, i.e. $w_{k,m} \geq 0$ [38], [39], Bounded Variable Least Squares, i.e. $l_{k,m} \leq w_{k,m} \leq u_{k,m}$ in which $l_{k,m}$ and $u_{k,m}$ are lower and upper bounds [40], respectively, and Bounded Variable with Constant Sum, i.e. $-1 < w_{k,m} < 1, \sum_{k=1}^{K} w_{k,m} = 1$ [41]. In this study we simply constrain the weights between 0 and 1, i.e. $0 \leq w_{k,m} \leq 1$. By solving M different linear regression problems, we will get the optimal weight matrix $\mathbb{W} = \{\mathbb{W}_m\}_{m=1}^{M}$.

Given an unsegmented image $\mathbf{I}_{test}$, it is segmented firstly by $\{\mathbf{h}_k\}_{k=1}^{K}$ to get the prediction matrices $\{\mathbf{P_k}(y_m|\mathbf{I}_{test})\}(k = 1, ..., K, m = 1, ..., M)$. Then the augmented data is created for $\mathbf{I}_{test}$ by concatenating it with $\{\mathbf{P_k}(y_m|\mathbf{I}_{test})\}$ as additional image channels.

$$\mathbf{I}_{test}^* = \mathbf{I}_{test} \cup \{\mathbf{P_k}(y_m|\mathbf{I}_{test})\}, k = 1, ..., K, m = 1, ..., M \quad (8)$$

The trained segmentation models of the second layer $\{\mathbf{h}_k^*\}_{k=1}^{K}$ are then applied on $\mathbf{I}_{test}^*$ to get the prediction matrices $\{\mathbf{P_k}(y_m|\mathbf{I}_{test}^*)\}(k = 1, ..., K, m = 1, ..., M)$. The class memberships of an image pixel $\mathbf{I}_{test}^*(i,j)$ are found via linear

combination of the prediction probabilities and the associated weights as:

$$CM_m(\mathbf{I}_{test}(i,j)) = \sum_{k=1}^{K} w_{k,m} P_k(y_m|\mathbf{I}^*_{test}(i,j)) = \mathbb{P}_m(\mathbf{I}^*_{test}(i,j))\mathbb{W}_m \quad (9)$$

in which $\mathbb{P}_m(\mathbf{I}^*_{test}(i,j))$ and $\mathbb{W}_m$ are defined as follows :

$$\mathbb{P}_m(\mathbf{I}^*_{test}(i,j)) = [P_1(y_m|\mathbf{I}^*_{test}(i,j)), ..., P_K(y_m|\mathbf{I}^*_{test}(i,j))] \quad (10)$$

$$\mathbb{W}_m = [w_{1,m}, w_{2,m}, ..., w_{K,m}]^T \quad (11)$$

Finally, the predicted class label is obtained by getting the label corresponding to the maximum value of class memberships:

$$\hat{m} = argmax_{m=1,...,M} CM_m\{\mathbf{I}_{test}(i,j)\} \quad (12)$$

$$\mathbf{I}_{test}(i,j) \in y_{\hat{m}} \quad (13)$$

The combining and training procedure is described in Algorithm 1. Algorithm 1 receives inputs including training set $\mathbf{D} = \{\mathbf{I}_n, \mathbf{Y}_n\}_{n=1}^N$ and segmentation algorithms $\{\mathcal{K}_k\}_{k=1}^K$. Lines 2-7 create the probability matrices via T-fold cross-validation procedure. Line 8 creates the augmented input data for the second layer via equations 2. Lines 10-14 create the second-level predictions for all training pixels $\mathbf{L}^*$ via T-fold cross-validation procedure. Lines 16-20 find the optimal weight matrix via equation 7. Lines 21-24 train the segmentation models on the original training data and the augmented data respectively. Line 25 returns the trained models and the optimal weight matrix.

The testing procedure inputs an image $\mathbf{I}_{test}$, the trained models and the optimal weight matrix (see Algorithm 2). Lines 1-2 creates the probability matrix, while in line 3, the augmented input to the second layer is created by using equations 8. Lines 4-5 create second-level probability matrix from augmented input. Line 6-7 use equations 9, 12 and 13 to combine the second-level predictions of segmentation models by using the optimal weight matrix $\mathbb{W}$. Finally line 8 returns the final segmentation result.

Algorithm 1 Two-layer ensemble for segmentation

Input: Training set $\mathbf{D} = \{\mathbf{I}_n, \mathbf{Y}_n\}_{n=1}^N$, segmentation algorithms $\{\mathcal{K}_k\}_{k=1}^K$
Output: Trained segmentation models $\{\mathbf{h}_k\}_{k=1}^K$, $\{\mathbf{h}^*_k\}_{k=1}^K$ and optimal weights $\mathbb{W}$
1: (Posterior probability generation)
2: $\{\mathbf{D}_1, \mathbf{D}_2, ..., \mathbf{D}_T\} = T - partition(\mathbf{D})$
3: **for** $t \leftarrow 1$ to T **do**
4: **for** $k \leftarrow 1$ to K **do**
5: $\mathbf{h}_{k,t} = Learn(\mathcal{K}_k, \mathbf{D} \setminus \mathbf{D}_t)$
6: **for** $\mathbf{I}$ in $\mathbf{D}_t$ **do**
7: $\{\mathbf{P_k}(y_m|\mathbf{I})\}_{m=1}^M = Segment(\mathbf{h}_{k,t}, \mathbf{I})$
8: $\mathbf{D}^* = \{\mathbf{I}^*_n, \mathbf{Y}_n\}_{n=1}^N$,where $\mathbf{I}^*_n$ is defined as equations 2
9: (2nd-level probability generation)
10: $\mathbf{L}^* = \emptyset, \{\mathbf{D}^*_1, \mathbf{D}^*_2, ..., \mathbf{D}^*_T\} = T - partition(\mathbf{D}^*)$
11: **for** $t \leftarrow 1$ to T **do**
12: **for** $k \leftarrow 1$ to K **do**
13: $\mathbf{h}^*_{k,t} = Learn(\mathcal{K}_k, \mathbf{D}^* \setminus \mathbf{D}^*_t)$
14: $\mathbf{L}^* = \mathbf{L}^* \cup Segment(\mathbf{h}^*_{k,t}, \mathbf{D}^*_t)$
15: (Weight vector generation)
16: **for** $m \leftarrow 1$ to M **do**
17: Get $\mathbf{L}^*_m$ by equation 5
18: Get $\mathbb{Y}_m$ by equation 6
19: Find $\mathbb{W}_m = \{w_{k,m}\}, k = 1, ..., K$ by solving equation 7
20: $\mathbb{W} = \{\mathbb{W}_m\}_{m=1}^M$
21: (Base segmentation algorithms generation)
22: **for** $k \leftarrow 1$ to K **do**
23: $\mathbf{h}_k = Learn(\mathcal{K}_k, \mathbf{D})$
24: $\mathbf{h}^*_k = Learn(\mathcal{K}_k, \mathbf{D}^*)$
25: **return** $\{\mathbf{h}_k\}_{k=1}^K$, $\{\mathbf{h}^*_k\}_{k=1}^K$, and $\mathbb{W}$

Algorithm 2 Test process for two-layer ensemble for segmentation

Input: Test image $\mathbf{I}_{test}$, trained segmentation models $\{\mathbf{h}_k\}_{k=1}^K, \{\mathbf{h}^*_k\}_{k=1}^K$ and the weight $\mathbb{W}$
Output: Prediction for $\mathbf{I}_{test}$
1: **for** $k \leftarrow 1$ to K **do**
2: $\{\mathbf{P_k}(y_m|\mathbf{I}_{test})\}_{m=1}^M = Segment(\mathbf{h}_k, \mathbf{I}_{test})$
3: $\mathbf{I}^*_{test}$ created from $\mathbf{I}_{test}$ and $\{\mathbf{P_k}(y_m|\mathbf{I}_{test})\}_{m=1}^M$ using 8
4: **for** $k \leftarrow 1$ to K **do**
5: $\mathbf{P}^*_\mathbf{k}(y_m|\mathbf{I}_{test}) = \mathbf{P}^*_\mathbf{k}(y_m|\mathbf{I}_{test}) \cup Segment(\mathbf{h}^*_k, \mathbf{I}^*_{test}); m = 1, ..., M$
6: Use 9 to combine the predictions $\{\mathbf{P}^*_\mathbf{k}(y_m|\mathbf{I}_{test})\}; m = 1, ..., M; k = 1, ..., K$
7: Use 12 and 13 to get the final prediction
8: **return** The final prediction.

IV. Experimental Studies

In this experiment, we used UNet [9], LinkNet [42] and Feature Pyramid Network (FPN) [43], which are three popular segmentation architectures. The backbones used were VGG16 [15] and ResNet34 [16], pretrained on the ImageNet dataset [17]. In total, there were 6 segmentation models used in the experiments. All segmentation algorithms were run for 300 epochs. The number of folds in the cross-validation procedure was set to 5. We compared the performance of the proposed ensemble to the 6 segmentation algorithms and one layer ensemble system with weights-based combiner, denoted by OLE in the tables.

A. Performance metrics

The performance of our proposed method and the related benchmarks were evaluated using two popular segmentation metrics. Suppose there are M classes, and there are N images each having size (W, H). Let $\mathbf{P}$ and $\mathbf{G}$ be the prediction of a segmentation model on these images and the corresponding ground truth:

$$\mathbf{P} = [p_1, p_2, ..., p_M], \mathbf{G} = [g_1, g_2, ..., g_M] \quad (14)$$

where p_m is a vector with size $(N \times W \times H, 1)$ associated with class label y_m in which its element is the prediction for each pixel in the form of crisp label i.e. belonging to $\{0, 1\}$. Likewise, g_m is a vector with size $(N \times W \times H, 1)$ associated with class label y_m in which each element which is the ground

truth of each pixel in the form of crisp label i.e. belonging to $\{0, 1\}$. Dice coefficient for the m^{th} class is then defined as follows [44]:

$$DC_m = \frac{2\mathbf{p}_m^{\mathbf{T}}\mathbf{g}_m}{||\mathbf{p}_m||^2 + ||\mathbf{g}_m||^2} \tag{15}$$

In the context of medical image analysis, local discrepancies between contours are often of interest as well. For example, radiation treatment planning applications require quantified errors in geometric displacement to ensure target coverage, normal tissue avoidance, and similar analyses [45]. We therefore reported one measure based on distance between geometrical contours. 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 Hausdorff distance HD associated with class y_m is calculated as follows [46] :

$$HD_m = max(d(GT_m, PR_m), d(PR_m, GT_m)) \tag{16}$$

where $d(A, B)$ is the directed Hausdorff distance:

$$d(A, B) = \frac{1}{|A|}\sum_{a\in A}\min_{b\in B}||a - b|| \tag{17}$$

It is noted that the low Hausdorff distance or high Dice coefficient shows the good segmentation result.

B. Kvasir-SEG dataset

The first dataset used in this paper is Kvasir-SEG [47], which consists of 1000 gastrointestinal polyp images, 200 of which is used for testing. The task is to segment the polyps in the images. Comparative evaluation of the segmentation models and the proposed method in Dice coefficient and Hausdorff distance is shown in Table I. The methods having VGG16 as backbone perform poorly, with Dice measure at just 0.0. In contrast, UNet-ResNet34, LinkNet-ResNet34 and FPN-ResNet34 achieve a Dice coefficient at 0.878, 0.879 and 0.887 respectively, while OLE achieves 0.888, which is roughly the same as FPN-ResNet34. The proposed method achieves a score of 0.892, which is an increase of 0.4% compared to the second best (OLE). For the Hausdorff distance, LinkNet-VGG16 has a very high score at around 271.7, while UNet-VGG16 achieves a score of 10.402 and FPN-VGG16 has a score of 0.0 (detect nothing). On the other hand, among the methods using ResNet34 backbone, UNet-ResNet34 has the highest Hausdorff score at 55.591, followed by LinkNet-ResNet34 at 51.241, FPN-ResNet34 at 50.321 and OLE at 49.38 . The proposed method achieves a Hausdorff distance of 48.831, which is better than the OLE by a difference of 0.55.

Figure 3 shows the result of six segmentation models, OLE, the proposed ensemble, the mask of test image and the original test image. The results made by methods using backbone VGG16 are not shown because they could not predict anything. All the segmentation algorithms segmented correctly the left part of the polyp. However, for the right part, UNet-ResNet34 and FPN-ResNet34 obtained a big hole in the lower and upper part respectively, while LinkNet-ResNet34 and OLE failed to segment the right part. The proposed ensemble correctly segmented both the left and the right part of the polyp, with the exception of a relatively small hole in the middle. The reason of better performance of the proposed ensemble is that it takes into consideration information not only from the input image but also from the predictions in generating the segmentation models.

TABLE I
KVASIR-SEG RESULT FOR DICE AND HAUSDORFF MEASURE

Segmentation algorithm	Dice	Hausdorff
UNet-VGG16	0	10.402
LinkNet-VGG16	0.001	271.674
FPN-VGG16	0	0
UNet-ResNet34	0.878	55.591
LinkNet-ResNet34	0.879	51.241
FPN-ResNet34	0.887	50.321
OLE	0.888	49.38
Proposed ensemble	**0.892**	**48.831**

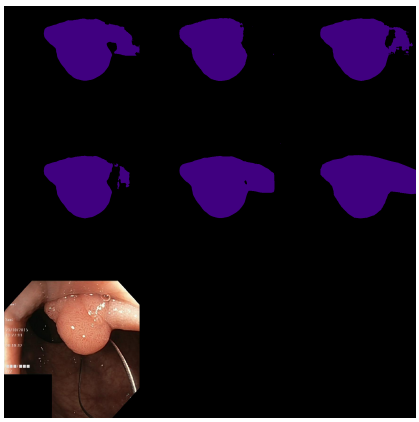

Fig. 3. Example result for Kvasir-SEG dataset. From left to right, top to bottom: UNet-ResNet34, LinkNet-ResNet34, FPN-ResNet34, OLE, proposed method, ground truth mask, and test image. The results made by segmentation algorithms using backbone VGG16 are not shown because they were not able to detect the polyps.

The proposed ensemble has higher training time than the benchmark algorithms. Compared with OLE which took about 2 days for training on this dataset, our two-layer ensemble trained for 4 days. In our training process, we solved Equation 7 to find the combining weights. Even though the optimisation problem in Equation 7 works on L_m^* matrix with 278528000 rows, it took only 5 minutes to find the weights by using *sklearn* library[1], which was the same as with OLE. Meanwhile, the testing time of proposed ensemble for 200 test images was 11 seconds, while OLE took 7 seconds.

C. CAMUS dataset

The second dataset used in this paper was the Cardiac Acquisitions for Multi-structure Ultrasound Segmentation (CAMUS) dataset [48], which is a dataset provided by a competition for accurate segmentation of 2D echocardiographic

[1]https://scikit-learn.org/

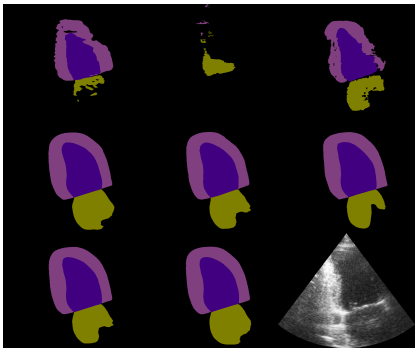

Fig. 4. Example result for CAMUS dataset. From left to right, top to bottom: UNet-VGG16, LinkNet-VGG16, FPN-VGG16, UNet-ResNet34, LinkNet-ResNet34, FPN-ResNet34, OLE, proposed method, and test image (mask image not available).

images. The dataset consists of cardiographic images and segmentation of 500 patients, acquired from clinical exams at the University Hospital of St Etienne, recorded at two cardiographic positions namely End Diastolic (ED) and End Systolic (ES). Three expert cardiologists were involved in the segmentation of the images. There are three classes: Left ventricle, Myocardium and Left atrium. The data of 50 patients are withheld for testing in which the submission link for evaluation is available here[2].

Table II and III shows the result of the segmentation models and the proposed ensemble. We included the author's best results for each measure on this dataset [48]. It can be seen that with respect to the Dice measure, the proposed method achieved best result on all cases. For the ED case, the proposed method achieved best result on the Myocardium and Left atrium class at 0.96 and 0.907, compared to the second best result at 0.959 and 0.9 of OLE respectively. On the Left ventricle class, the proposed method achieved the same result as the second best at 0.946. For the ES case, the proposed ensemble achieved roughly the same result as OLE on Left ventricle and Myocardium class at 0.93 and 0.955 respectively. However, on Left atrium class, the proposed method achieved a score of 0.934, which is better than the second best (OLE) at 0.929. The segmentation algorithms with VGG16 backbone performed very poorly on all cases, achieving only from 0.2 (LinkNet-VGG16 on Myocardium) to 0.307 (UNet-VGG16 on Left ventricle).

With the Hausdorff distance, the proposed ensemble beats the segmentation models in all classes for the ES case. It achieved 4.4 on the Left ventricle class while the second best among the segmentation models (LinkNet-ResNet34) achieved only 4.7 and OLE achieved 4.6. The same observation is on the Myocardium and Left atrium class. However, for the ED case, the proposed ensemble performed worse than LinkNet-VGG16, such as in the Myocardium class where the proposed method achieved a score of 5 while the LinkNet-VGG16 segmentation algorithms achieved 3.8, which is better by a score of 1.2. This can be explained from the observation in [45] in which it is possible for the Hausdorff distance to miscalculate when the curvature has a high degree of winding and low similarity.

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

TABLE II
RESULT FOR CAMUS DATASET, DICE MEASURE

	End Diastolic			End Systolic		
	Left ventricle	Myocardium	Left atrium	Left ventricle	Myocardium	Left atrium
Author's best	0.936	0.956	0.889	0.913	0.946	0.918
UNet-VGG16	0.307	0.3	0.244	0.295	0.305	0.244
UNet-ResNet34	**0.946**	0.958	0.9	0.925	0.952	0.927
LinkNet-VGG16	0.203	0.2	0.197	0.106	0.113	0.119
LinkNet-ResNet34	0.942	0.958	0.897	0.928	0.954	0.922
FPN-VGG16	0.354	0.356	0.279	0.317	0.317	0.241
FPN-ResNet34	0.945	0.958	0.899	0.927	0.953	0.926
OLE	**0.946**	0.959	0.9	0.929	**0.955**	0.929
Proposed ensemble	**0.946**	**0.96**	**0.907**	**0.93**	**0.955**	**0.934**

TABLE III
RESULT FOR CAMUS DATASET, HAUSDORFF MEASURE

	End Diastolic			End Systolic		
	Left ventricle	Myocardium	Left atrium	Left ventricle	Myocardium	Left atrium
Author best	5.3	5.2	5.7	5.3	5.7	5.3
UNet-VGG16	15.8	25.3	16.9	6.6	14.3	8.6
UNet-ResNet34	5.1	5.2	5	4.9	5.3	5.1
LinkNet-VGG16	**3.1**	**3.8**	**4.5**	8.1	8.9	9.2
LinkNet-ResNet34	5	5.2	5.5	4.7	5.1	5.5
FPN-VGG16	3.8	4.9	7.1	4.8	6.8	8
FPN-ResNet34	4.8	5.3	5.5	4.8	5.4	5.2
OLE	4.7	5.1	5.3	4.6	5	4.9
Proposed ensemble	4.7	5	4.8	**4.4**	**4.8**	**4.7**

Figure 4 shows an example in which the proposed ensemble improved on the result of the segmentation models. While the predictions by the methods using VGG16 backbone (first row) contain a number of deformations compared to the test image, the predictions on the second row using ResNet34 backbone give better results. It can be seen that LinkNet-ResNet34 and FPN-ResNet34 failed to predict a large region in the bottom right of the Left atrium (second row, second and third column). On the other hand, while the prediction by UNet-ResNet34 is better than that of LinkNet-ResNet34 and FPN-ResNet34, it nevertheless contains a sharp inward region which was not correctly segmented. The proposed ensemble has improved upon the predictions by the constituent segmentation models as its prediction overall segment the bottom right part correctly.

V. CONCLUSION

In this paper, we presented a two-layer ensemble of deep learning models for segmentation of medical images. The key idea is to use the probability prediction by the constituent models in the first layer as augmented data for the second layer. The output probability prediction by the the second layer is combined by using a weight-based scheme which is not only a effective combiner but also computational efficient. The weights are found by solving a linear regression problem associated with each class label. Our results on two benchmark datasets show that the proposed ensemble method is able to combine the strengths and mitigate the drawbacks of the constituent segmentation methods, resulting in an overall improvement.

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] J. Shotton, M. Johnson *et al.*, "Semantic text on forests for image categorization and segmentation," in *Proceedings of CVPR*, 2008, pp. 1–8.

[2] B. Fulkerson, A. Vedaldi *et al.*, "Class segmentation and object localization with superpixel neighborhoods," in *Proceedings of ICCV*, 2009, pp. 670–677.

[3] J. Carreira, R. Caseiro *et al.*, "Semantic Segmentation with Second-Order Pooling," in *ECCV*, 2012, pp. 430–443.

[4] P. Krähenbühl and V. Koltun, "Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials," in *Advances in Neural Information Processing Systems*, 2011, pp. 109–117.

[5] Xuming He, R. S. Zemel *et al.*, "Multiscale conditional random fields for image labeling," in *Proceedings of CVPR*, vol. 2, 2004, pp. II–II.

[6] A. Krizhevsky, I. Sutskever *et al.*, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2012.

[7] J. Long, E. Shelhamer *et al.*, "Fully convolutional networks for semantic segmentation," in *Proceedings of CVPR*, 2015, pp. 3431–3440.

[8] V. Badrinarayanan, A. Kendall *et al.*, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[9] O. Ronneberger, P. Fischer *et al.*, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015.

[10] W. Zhang, R. Li *et al.*, "Deep convolutional neural networks for multi-modality isointense infant brain image segmentation," *NeuroImage*, vol. 108, pp. 214 – 224, 2015.

[11] D. Shen, G. Wu *et al.*, "Deep Learning in Medical Image Analysis," *Annual review of biomedical engineering*, vol. 19, pp. 221–248, Jun. 2017.

[12] Q. V. Le, J. Ngiam *et al.*, "On optimization methods for deep learning," in *Proceedings of ICML*. Omnipress, 2011, p. 265–272.

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, Y. Bengio and Y. LeCun, Eds., 2015.

[14] A. C. Wilson, R. Roelofs *et al.*, "The marginal value of adaptive gradient methods in machine learning," in *Proceedings of NIPS*, 2017, p. 4151–4161.

[15] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2015.

[16] K. He, X. Zhang *et al.*, "Deep residual learning for image recognition," in *Proceedings of CVPR*, 2016, pp. 770–778.

[17] J. Deng, W. Dong *et al.*, "ImageNet: A Large-Scale Hierarchical Image Database," in *Proceedings of CVPR*, 2009.

[18] A. Garcia-Garcia, S. Orts-Escolano *et al.*, "A survey on deep learning techniques for image and video semantic segmentation," *Applied Soft Computing*, vol. 70, pp. 41 – 65, 2018.

[19] L. Chen, G. Papandreou *et al.*, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[20] P. Krähenbühl and V. Koltun, "Parameter learning and convergent inference for dense random fields," in *Proceedings of ICML*, vol. 28, 2013.

[21] Y. Xie, Z. Zhang *et al.*, "Spatial Clockwork Recurrent Neural Network for Muscle Perimysium Segmentation," *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, vol. 9901, pp. 185–193, Oct. 2016.

[22] J. Kleesiek, G. Urban *et al.*, "Deep MRI brain extraction: A 3D convolutional neural network for skull stripping," *NeuroImage*, vol. 129, pp. 460–469, Apr. 2016.

[23] A. Alansary, K. Kamnitsas *et al.*, "Fast Fully Automatic Segmentation of the Human Placenta from Motion Corrupted MRI," in *MICCAI*, Oct. 2016.

[24] M. Fernández-Delgado, E. Cernadas *et al.*, "Do we need hundreds of classifiers to solve real world classification problems?" *Journal of Machine Learning Research*, vol. 15, no. 90, pp. 3133–3181, 2014.

[25] T. T. Nguyen, M. T. Dang *et al.*, "A weighted multiple classifier framework based on random projection," *Information Sciences*, vol. 490, pp. 36 – 58, 2019.

[26] T. T. Nguyen, M. P. Nguyen *et al.*, "Combining heterogeneous classifiers via granular prototypes," *Applied Soft Computing*, vol. 73, pp. 795 – 815, 2018.

[27] T. T. Nguyen, A. V. Luong *et al.*, "Ensemble selection based on classifier prediction confidence," *Pattern Recognition*, vol. 100, p. 107104, 2020.

[28] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[29] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, p. 785–794.

[30] J. J. Rodriguez, L. I. Kuncheva *et al.*, "Rotation forest: A new classifier ensemble method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1619–1630, 2006.

[31] Z.-H. Zhou and J. Feng, "Deep Forest: Towards An Alternative to Deep Neural Networks," *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3553–3559, 2017.

[32] L. V. Utkin, M. S. Kovalev *et al.*, "A deep forest classifier with weights of class probability distribution subsets," *Knowledge-Based Systems*, vol. 173, pp. 15 – 27, 2019.

[33] T. T. Nguyen, N. Van Pham *et al.*, "Multi-layer heterogeneous ensemble with classifier and feature selection," in *Proceedings of GECCO*, 2020, p. 725–733.

[34] Z. Qi, B. Wang *et al.*, "When ensemble learning meets deep learning: a new deep support vector machine for classification," *Knowledge-Based Systems*, vol. 107, pp. 54 – 60, 2016.

[35] C. J. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, Jun. 1998.

[36] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119 – 139, 1997.

[37] G. Litjens, T. Kooi *et al.*, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60–88, Dec. 2017.

[38] C. Lawson and R. Hanson, "Solving least squares problems," in *Classics in applied mathematics*, 1995.

[39] P. Stark, "Bounded-variable least-squares: an algorithm and applications," in *Computational Statistics*, 2008.

[40] R. Bro and S. de Jong, "A fast non-negativity-constrained least squares algorithm," *Journal of Chemometrics*, vol. 11, 1997.

[41] L. Zhang and W. Zhou, "Sparse ensembles using weighted combination methods based on linear programming," *Pattern Recognit.*, vol. 44, pp. 97–106, 2011.

[42] A. Chaurasia and E. Culurciello, "Linknet: Exploiting encoder representations for efficient semantic segmentation," in *2017 IEEE Visual Communications and Image Processing (VCIP)*, 2017, pp. 1–4.

[43] T. Lin, P. Dollár *et al.*, "Feature pyramid networks for object detection," in *Proceedings of CVPR*, 2017, pp. 936–944.

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

[45] H. S. Kim, S. B. Park *et al.*, "Bidirectional local distance measure for comparing segmentations," *Medical Physics*, vol. 39, no. 11, pp. 6779–6790, Nov. 2012.

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

[47] D. Jha, P. H. Smedsrud *et al.*, "The Kvasir-SEG Dataset." [Online]. Available: https://datasets.simula.no/kvasir-seg/

[48] S. Leclerc, E. Smistad *et al.*, "Deep Learning for Segmentation Using an Open Large-Scale Dataset in 2D Echocardiography," *IEEE transactions on medical imaging*, vol. 38, no. 9, pp. 2198–2210, 2019.