Learning dynamic weights for an ensemble of deep models applied to medical imaging classification

Andre G. C. Pacheco
Graduate Program in Computer Science
Federal University of Espirito Santo
Vitoria, Brazil
agcpacheco@inf.ufes.br

Thomas Trappenberg
Faculty of Computer Science
Dalhousie University
Halifax, Canada
tt@dal.ca

Renato A. Krohling
Graduate Program in Computer Science
and Production Engineering Department
Federal University of Espirito Santo, UFES
Vitoria, Brazil
rkrohling@inf.ufes.br

***Abstract*—An ensemble of deep models is commonly used to provide more robust and accurate performance for medical image classification. A drawback of the most common ensemble aggregation operators is that they give the same importance to all models in the ensemble. As a consequence, they cannot identify weak models that may negatively influence the ensemble performance. In this work, we propose a new method based on the Dirichlet distribution and Mahalanobis distance to learn dynamic weights to an ensemble of deep learning models. Through this method, it is possible to reduce the influence of weak models for each new sample evaluated by the ensemble and perform online ensemble pruning. We evaluate this method for an ensemble of six well-known deep models applied to four medical imaging datasets. The experiments show that our method achieves the best balanced accuracy for 2 out of 4 datasets and increases the confidence of the ensemble predictions.**

***Index Terms*—Ensemble weighting, Ensemble pruning, Deep learning, Dirichlet distribution, Mahalanobis distance**

I. Introduction

Deep learning models have been achieving remarkable results for different medical imaging classification such as skin cancer detection [1], chest radiography diagnosis [2], lung cancer prediction [3], diabetic retinopathy grading [4], among others. Nonetheless, the performance of these models may vary due to multiple reasons such as weights' initialization, hyperparameters, data variance, and overfitting. A common strategy to deal with this issue is to trust in predictions from an ensemble of different deep models [5], [6], [7], which is known to be more robust and accurate than single models [8].

For deep learning, it is common to use a stack [9] of homogeneous or heterogeneous models as an ensemble. In the homogeneous approach, to achieve diversity, the model is trained several times using different training sets and/or parameters. On the other hand, in the heterogeneous method, diversity is achieved by training different models on the same training set [10], [11]. Since deep learning is greedy, i.e., it demands large amounts of training data, the heterogeneous approach is the most popular one since it does not partition the training set. Currently, this approach is applied by the top-ranked deep learning models in well-known computer vision challenges such as ImageNet [12], ISIC [13], and CheXpert [2].

The most used method to combine an ensemble of deep models, regardless it is a heterogeneous or homogeneous ensemble, is through aggregating the models' predictions. Aggregation operators are typically the majority voting [14], [15], [16], predictions average [5], [17], [18], [19], maximum prediction [20], and product of predictions [20]. Other known information fusion techniques such as Choquet integral [21], [22] and Dempster-Shafer theory [23] are avoided due to the computational burden, which is an important issue since deep learning itself is computationally expensive.

A drawback of the most common operators mentioned above is that they assign the same importance to all models in the ensemble. As a result, these operators cannot identify weak models that may negatively influence the ensemble performance. In order to tackle this issue, Krawczyk and Woźniak [24] proposed a method based on Gaussian function to estimate the weights of each model in the ensemble. Harangi [20] proposed to weight the deep models in an ensemble based on the models' accuracy in the validation set. Both approaches presented an improvement in the ensemble's performance since the influence of weak models is reduced. Nonetheless, they assign static weights to the models, i.e., the weights are the same for all samples evaluated by the ensemble. A promising way to deal with this issue is through Dynamic Selection (DS), a technique to select a single classifier from the ensemble on the fly, i.e., according to each new sample to be classified [15].

In this work, we propose a novel approach based on the Dirichlet distribution [25] and Mahalanobis distance [26] to learn dynamic weights to an ensemble of deep models. There are two main advantages of a dynamic weighting: 1) it is possible to reduce the influence of weak models for each new sample evaluated by the ensemble; 2) one can apply online ensemble pruning [27], [28]. The main contributions of this work are summarized as follows:

- We propose a new approach to learn dynamic weight to an ensemble of deep models. In brief, we estimate the Dirichlet distribution of the models' predictions and apply the Mahalanobis distance to determine the weight of each model.
- The weights learned by the proposed approach are used to reduce the impact of weak models' predictions on the aggregation operator. In addition, we introduce a mechanism to prune k models online.

- We apply six well-known deep learning models to four medical imaging classification. We present a discussion to demonstrate the advantages and limitations of the proposed approach.

The rest of this paper is organized as follows: in section 2, we introduce the Dirichlet distribution; in section 3, the proposed approach is described; in section 4, we carry out the experiments and present a discussion about the results obtained; and in section 5, we draw some conclusions.

II. Dirichlet distribution

The Dirichlet distribution is a generalization of the Beta distribution for multiple random variables. Let us consider a random variable $\mathbf{p} = \{p_1, \cdots, p_k\}$. The distribution is defined for all $\mathbf{p} \in \mathbb{S}$, where $\mathbb{S}$ is the standard simplex defined as $\{\mathbf{p} \in \mathbb{R}^k : p_i \geq 0, \sum p_i = 1\}$. In other words, the elements of $\mathbf{p}$ must be positive and sum up to 1. For this reason, each p_i may be interpreted as a probability itself and the Dirichlet distribution is known as a distribution over probability distributions. Thus, it is commonly applied to estimate proportional data [25].

The Dirichlet probability density function is defined by [25]:

$$p(\mathbf{p}) \sim \mathfrak{D}(\mathbf{p}; \boldsymbol{\alpha}) = \frac{\Gamma\left(\sum_{i=1}^{k} \alpha_i\right)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \prod_{i=1}^{k} p_i^{\alpha_i - 1} \tag{1}$$

where $\boldsymbol{\alpha} = \{\alpha_1, \cdots, \alpha_k\}$ is the distribution's parameters and $\alpha_k > 0$. It is important to note that $\boldsymbol{\alpha}$ and $\mathbf{p}$ are both k-dimensional. Γ is the Gamma function defined as $\Gamma(x) = \int_0^{+\infty} t^{x-1} e^{-t} dt$.

A. Expectation, variance and covariance

Considering a Dirichlet distribution $\mathfrak{D}(\mathbf{p}; \boldsymbol{\alpha})$, with $\mathbf{p} = \{p_1, \cdots, p_k\}$ and $\boldsymbol{\alpha} = \{\alpha_1, \cdots, \alpha_k\}$, the expectation, variance, and covariance of this distribution are defined in (2), (3), and (4), respectively [25].

$$E[p_i] = \frac{\alpha_i}{\sum_{j=1}^{k} \alpha_j} \tag{2}$$

$$Var[p_i] = \frac{\alpha_i \left(\sum_{i=1}^{k} \alpha_i - \alpha_i\right)}{\left(\sum_{i=1}^{k} \alpha_i\right)^2 \left(\sum_{i=1}^{k} \alpha_i + 1\right)} \tag{3}$$

$$Cov[p_i, p_j] = \frac{-\alpha_i \alpha_j}{\left(\sum_{i=1}^{k} \alpha_i\right)^2 \left(\sum_{i=1}^{k} \alpha_i + 1\right)}, \; i \neq j \tag{4}$$

with $i = 1, \cdots, k$.

B. Estimating a Dirichlet distribution

Let us consider a set of data $D = \{\mathbf{p}_1, \cdots, \mathbf{p}_N\}$, where each $\mathbf{p}_n = \{p_1, \cdots, p_k\}$, $\mathbf{p}_n \in \mathbb{S}$, $n = 1, \cdots, N$, and N is the number of samples in the set. The parameters $\boldsymbol{\alpha}$ of a Dirichlet distribution may be estimated from D using the Maximum-Likelihood Estimation (MLE) [29]. The Dirichlet log-likelihood function of the data is given by [30]:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\alpha}) &= \log p(D \mid \boldsymbol{\alpha}) = \log \prod_n p(\mathbf{p}_n \mid \boldsymbol{\alpha}) \\
&= \log \prod_n \frac{\Gamma\left(\sum_k \alpha_k\right)}{\prod_k \Gamma(\alpha_k)} \prod_k p_{nk}^{\alpha_k - 1} \\
&= N \left(\log \Gamma \left(\sum_k \alpha_k \right) - \sum_k \log \Gamma(\alpha_k) + \right. \\
&\qquad \left. + \sum_k (\alpha_k - 1) \log \bar{p}_k \right)
\end{aligned} \tag{5}$$

where $\log \bar{p}_k = \frac{1}{N} \sum_n \log p_{nk}$. Since the Dirichlet distribution belongs to the exponential family [31], the objective function $\mathcal{L}(\boldsymbol{\alpha})$ is convex in $\boldsymbol{\alpha}$. Thus, it is guaranteed that the function has a unique optimum [30].

A simple way to maximize $\mathcal{L}(\boldsymbol{\alpha})$ is using the gradient ascent algorithm. To this end, the gradient of $\mathcal{L}(\boldsymbol{\alpha})$ is computed as [30]:

$$\frac{\partial \mathcal{L}}{\partial \alpha_k} = N \left(\Psi \left(\sum_k \alpha_k \right) - \Psi(\alpha_k) + \log \bar{p}_k \right) \tag{6}$$

where $\Psi = \frac{d \log \Gamma(x)}{dx}$ is known as Digamma function. From (6), we can apply the gradient ascent algorithm, considering that $\boldsymbol{\alpha} > 0$, to estimate the Dirichlet distribution for the set of data D. Nonetheless, Minka [30] proposed a fixed-point iteration for maximizing the log-likelihood and, consequently estimate $\boldsymbol{\alpha}$, that is faster than gradient ascent method. The main idea is to determine an initial value for $\boldsymbol{\alpha}$ and to find a lower bound function which is tight at $\boldsymbol{\alpha}$. Then, this function is optimized in order to find a new value of $\boldsymbol{\alpha}$. The method is obtained from the following inequality [32]:

$$\Gamma(x) \geq \Gamma(\hat{x}) e^{(x - \hat{x})\Psi(\hat{x})} \tag{7}$$

Applying this inequality to $\Gamma(\sum_k \alpha_k)$ the following lower bound on the log-likelihood is obtained:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\alpha}) \geq N \Bigg[& \left(\sum_k \alpha_k \right) \Psi \left(\sum_k \hat{\alpha}_k \right) - \sum_k \log \Gamma(\alpha_k) + \\
& + \sum_k (\alpha_k - 1) \log \bar{p}_k + \text{Const.} \Bigg]
\end{aligned} \tag{8}$$

Next, one maximizes the previous equation by setting the gradient to zero and solving it for $\boldsymbol{\alpha}$, which leads to the following fixed-point iteration [30]:

$$\alpha_k^{new} = \Psi^{-1} \left[\Psi \left(\sum_k \alpha_k^{old} + \log \bar{p}_k \right) \right] \tag{9}$$

Therefore, to estimate $\boldsymbol{\alpha}$, we determine an initial guess for alpha and update this value according to (9) for a given number of iterations or considering a convergence tolerance. In Algorithm 1 is described the pseudo-code for the Dirichlet estimation via fixed-point iteration method.

Algorithm 1: *Dirichlet estimation using fixed-point iteration method*

```
1  Input:
2      $D = \{\mathbf{p}_1, \cdots, \mathbf{p}_N\}$;
3      $tol$ and $max\_iter$;
4  $D_{log} = \frac{1}{N}\sum_{n,k}\log p_{nk}$;
5  $\alpha_{old} = \frac{1}{N}\sum_{n,k} p_{nk}$;
6  while $j < max\_iter$ do:
7      Compute $\alpha_{new}$ according to (9)
8      Compute $\mathcal{L}(\alpha_{old})$ and $\mathcal{L}(\alpha_{new})$ according to (5);
9      if $\mathcal{L}(\alpha_{old}) - \mathcal{L}(\alpha_{new}) < tol$:
10         break;
11     $\alpha_{old} = \alpha_{new}$;
12     $j = j + 1$;
13 return $\alpha_{new}$;
```

III. Learning dynamic weights for an ensemble of deep models

In this section, we describe our method to determine the weight of each deep learning model in an ensemble. The main idea behind this method is to learn the probability distribution of the deep models' predictions for a set of unseen samples and compute a score that represents the relevance of each of them in an ensemble. We describe the method in two steps. First, we describe the probability distribution estimation of the validation set predictions. Second, we detail the weights computation.

A. Step 1: estimating the probability distribution

To describe this step, let us consider a classification problem with $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ inputs, $Y = \{\mathbf{y}_1, \cdots, \mathbf{y}_N\}$ outputs, and $\mathbf{c} = \{c_1, \cdots, c_L\}$ labels, where N and L are the number of samples and labels, respectively. The first step to train a deep model on this task is to split X and Y into training (X_{tr} and Y_{tr}) and validation (X_{val} and Y_{val}) partitions. The model is trained using $\{X_{tr}, Y_{tr}\}$ to fit a function $f(\mathbf{x})$ that outputs an array $\mathbf{y}$ containing the logits for the sample $\mathbf{x}$ to each label c. Usually, the logit is converted to the probability of y to assume a label c, i.e, $p(y = c \mid \mathbf{x})$. The most common way to assign this probability is through the *softmax* function:

$$p_l(y_l = c_l \mid \mathbf{x}) = \frac{e^{y_l}}{\sum_{z=1}^{L} e^{y_z}}, \;\; l = 1, \cdots, L. \qquad (10)$$

Therefore, when a new sample $\hat{\mathbf{x}}$ is presented to f, it outputs an array of probabilities $\mathbf{p}$ that represents the confidence of this sample for each label $c_l \in \mathbf{c}$. In such case, we can interpret each array $\mathbf{p}$ as a multivariate random variable and apply the fixed-point method, described in Algorithm 1, to estimate the Dirichlet distribution of a given partition of X. In particular, our goal is to estimate the Dirichlet distribution for the following sets of probabilities:

- P_l^h: the set of probabilities obtained by the model to the validation set considering the label $c_l \in \mathbf{c}$ when the label is correctly predicted by the network. We name it the hit set for the label l.
- P_l^m: the set of probabilities obtained by the model to the validation set considering the label $c_l \in \mathbf{c}$ when the label is incorrectly predicted by the network. We name it the miss set for the label l.

In order to sample in the model space, we applied the dropout sampling [33] for the validation partition. The dropout sampling consists of performing the model with the dropout rate active during the inference phase. Normally, the dropout is active only during the training phase as a tool to avoid over-fitting and increase generalization [33]. The goal of activating it also during inference is to vary the model and assess how well the model generalizes for unseen samples. In our case, the sets P_l^h and P_l^m are collected after d_e executions of the network using dropout sampling with dropout rate equal to d_r. The parameter d_e controls the number of times that a given sample $\hat{\mathbf{x}}$ is presented to the model; while the parameter d_r is the rate of nodes in the model that will be active during the execution. Therefore, if the model is well trained the output probabilities should not present a high variation for the same sample.

B. Step 2: computing the dynamic weights

Let us consider an ensemble with G deep models $E = \{f_1(\mathbf{x}), \cdots, f_G(\mathbf{x})\}$ trained using $\{X_{tr}, Y_{tr}\}$. For each model $f_g(\mathbf{x})$, we apply the step 1, i.e, we perform the dropout sampling for X_{val} and estimate the Dirichlet distributions $\mathfrak{D}_g(P_l^h; \boldsymbol{\alpha}_l^h)$ and $\mathfrak{D}_g(P_l^m; \boldsymbol{\alpha}_l^m)$, i.e., the distributions for each P_l^h and P_l^m sets. We name them as the hit and miss distributions for the model g and label l, respectively.

Now, let us consider $\hat{\mathbf{p}}_g$ the predictions obtained to a new sample $\hat{\mathbf{x}}$ by a model f_g. The main goal of this step is to measure how far $\hat{\mathbf{p}_g}$ is from the hit and miss distributions for each label l. In particular, we observed that a reliable prediction will be closer to a hit and far from a miss. An effective method to measure the distance between a random variable $\mathbf{z}$ from its distribution $P(\mathbf{z})$ is the Mahalanobis distance, defined as [26]:

$$Mah(\mathbf{z}, P(\mathbf{z})) = \sqrt{(\mathbf{z} - \mathbb{E}[P(\mathbf{z})])^T S^{-1} (\mathbf{z} - \mathbb{E}[P(\mathbf{z})])} \qquad (11)$$

where $\mathbb{E}[P(\mathbf{z})]$ is the expectation value of the distribution $P(\mathbf{z})$ and S^{-1} is the inverse of the covariance matrix of the distribution. It is effective because it takes into account not only the expectation but also the variance of the distribution. Therefore, we apply the Mahalanobis distance to measure the difference between $\hat{\mathbf{p}_g}$ and the distributions $\mathfrak{D}_g(P_l^h; \boldsymbol{\alpha}_l^h)$ and $\mathfrak{D}_g(P_l^m; \boldsymbol{\alpha}_l^m)$. The Dirichlet expectation is computing according to (2) and the covariance matrix using (3) and (4).

The standard way to select the label that the model f_g assigns to a sample $\hat{\mathbf{x}}$ is through argmax($\hat{\mathbf{p}}_g$). However, we interpret $\hat{\mathbf{p}}_g$ as a multivariate random variable. Thus, to compute the Mahalanobis distance between $\hat{\mathbf{p}}_g$ and the hit and miss distributions, we take into account the contribution

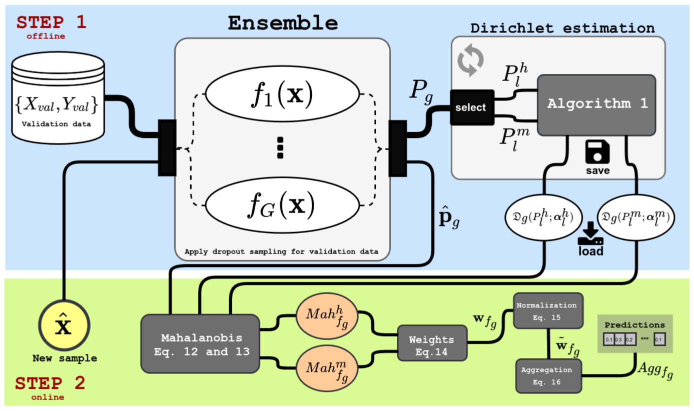


Fig. 1: A schematic diagram illustrating both steps of the proposed algorithm. In the first step, the validation data is used to get and save the hit and miss distributions. In the second step, the distributions are loaded and used to estimate the weights for each model within the ensemble according to the probabilities assigned to the new sample.

of each label l according to the probability assigned to it. This concept is described by (12) and (13).

$$Mah_{f_g}^h = \sum_{l=1}^{L} \hat{p}_{gl} Mah(\hat{\mathbf{p}}_g, \mathfrak{D}_g(P_l^h; \boldsymbol{\alpha}_l^h)), \; g = 1, \cdots, G \quad (12)$$

$$Mah_{f_g}^m = \sum_{l=1}^{L} \hat{p}_{gl} Mah(\hat{\mathbf{p}}_g, \mathfrak{D}_g(P_l^m; \boldsymbol{\alpha}_l^m)), \; g = 1, \cdots, G \quad (13)$$

Finally, we compute the relevance of the model f_g for a new sample $\hat{\mathbf{x}}$ as:

$$w_{f_g} = \frac{Mah_{f_g}^m}{Mah_{f_g}^h}, \; g = 1, \cdots, G \quad (14)$$

As we may note, this simple equation implements the idea previously mentioned. If $\hat{\mathbf{p}}_g$ is close to the hit distribution and far from the miss one, the weight increases. On the other hand, if $\hat{\mathbf{p}}_g$ is near to the miss and far from the hit, the weight decreases. To normalize the weights in the interval $[0, 1]$, we apply the following equation:

$$\tilde{w}_{f_g} = \frac{w_{f_g}}{\sum_{j=1}^{G} w_{f_j}}, \; g = 1, \cdots, G. \quad (15)$$

The last step of the proposed method is to compute the final aggregation. It is achieved through the following equation:

$$Agg_{f_g} = \sum_{j=1}^{L} \tilde{w}_{f_g} \hat{p}_j, \; g = 1, \cdots, G. \quad (16)$$

In Fig. 1 is illustrated a schematic diagram of the proposed algorithm. It is important to note that only the step 2 is computed online. As shown in the figure, the Dirichlet estimations for each model are saved during step 1. As such, the relevance of each model in the ensemble is computed online for a new unseen sample $\hat{\mathbf{x}}$, which means that the weights are dynamic and changes according to the presented sample. In this context, using this method we can identify weak models according to the sample. In addition, it is possible to prune the ensemble online by selecting the best g models for each new sample or setting a threshold for the weights.

IV. Experiments

In this section, we carry out some experiments to evaluate the performance of the proposed method. First, we describe the experiments' setup, including the datasets and deep learning models applied. Next, we show the results and compare them with standard approaches used in this area. Lastly, we present a discussion about the results.

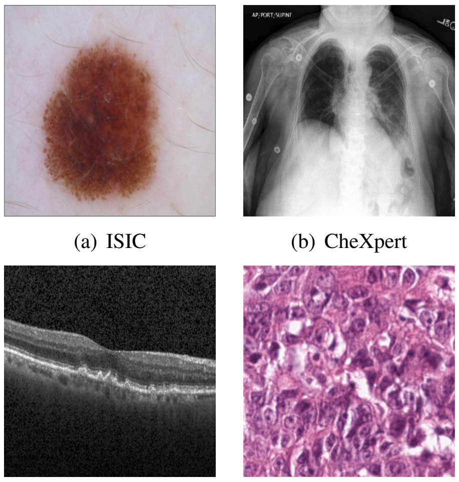
A. Experiments setup

In order to test our method, we create an heterogeneous ensemble composed of six well-known Convolution Neural Networks (CNNs): DenseNet-121 [34], GoogleNet [35], InceptionV4 [36], MobileNet-v2 [37], ResNet-50 [38], and VGG-16 [39]. The ensemble was trained on four medical datasets:

- **ISIC** 2019 [13]: this dataset contains 25,331 dermoscopy images with 8 different skin diseases.
- **CheXpert** [2]: a large chest radiograph dataset containing 224,316 images reporting the presence of different conditions in radiology. For this work, we select the pleural effusion condition considering the U-MultiClass, which has 3 classes: positive, negative, or uncertainty of the condition.
- **NCT**-CRC-HE-100K [40]: a set of 100,000 histology images of human colorectal cancer (CRC) and normal tissue.
- **OCT** [41]: a dataset containing 83,495 images of retinal optical coherence tomography (OCT) with 4 labels, 3 diseases and a normal retina.

In Fig. 2 is depicted an image sample of each dataset previously described.

All models in the ensemble were trained using their original architecture, except for the last layer that depends on the number of labels in the dataset. We performed the training



(a) ISIC (b) CheXpert

(c) OCT (d) NCT

Fig. 2: An example of an image from each medical dataset used in this work. We may observe that they have different features that affect the level of difficulty of each task.

phase for 120 epochs using Adam optimizer with a learning rate equal to 0.001, which is reduced by a rate of 0.1 if the model does not improve for 10 consecutive epochs. In addition, we used early stopping if the model does not improve for 15 consecutive epochs. As the datasets are imbalanced, i.e., the labels are not represented equally, we applied the weighted cross-entropy as the loss function in which the weights are determined according to the labels' frequency. All images were resized to 224×224 and we applied data augmentation using common image processing operations. For CheXpert and NCT we applied only horizontal and vertical flips [2], [40], and for ISIC we also include adjustments in brightness, contrast, saturation, and hue [18], [19].

For all experiments, we reserved 10% of each dataset for testing. The remaining 90% of data were used on the training phase through 5-fold cross-validation for assessing the effectiveness of the models. For each folder, we trained the models and estimate $\mathfrak{D}_g(P_l^m; \boldsymbol{\alpha}_l^m)$ and $\mathfrak{D}_g(P_l^m; \boldsymbol{\alpha}_l^m)$ as described in section III. To measure the performance, we computed the average and standard deviation of the following metrics: accuracy (ACC), balanced accuracy (BACC), aggregated area under the curve (AUC), and the cross-entropy (Loss). Finally, we compared the performance of our method with the most common aggregation operators: majority voting (MV), predictions average (PAvg), maximum prediction (Pmax), and product of predictions (PProd); and with the weighting average (WAvg) proposed by Harangi [20]. After the aggregation, we normalize all values to the interval $[0, 1]$. All procedures were implemented using Python and PyTorch and performed on Nvidia Tesla P-100. The code is available upon request.

B. Ensemble performance for medical imaging

In this section, we present the performance obtained by the deep models and the ensemble for the four medical imaging datasets previously described. In Table I is reported the performance for each deep model and each dataset. As we may note, there are different levels of performance among the datasets. While the deep models work quite well for NCT and OCT, they do not present the same performance for ISIC and CheXpert. In fact, as we can see through the loss metric, the models struggle to generalize to these latter datasets. It happens mainly because they present a high variability of features and strong similarities among diseases. For instance, in ISIC it is hard to distinguish melanoma and nevus [18], while in CheXpert it is common to confuse the uncertainty label between positive and negative [2].

Overall, due to the imbalance among the labels, the BACC is slightly lower than ACC for ISIC, NCT, and OCT. For CheXpert, it is approximately 10% lower. The reason for this difference in performance is that the models are not identifying the uncertainty label properly. In this context, we consider the BACC the priority metric in this experiment. Actually, this is the main metric for several medical task challenges such as ISIC [13].

Another important observation is that there is no model that presents the best performance for all datasets. Considering the average BACC, for ISCI the best performance is presented by InceptionV4, for CheXpert by Resnet-50, for NCT by MobileNet-V2, and for OCT by VGG-16. This is a result that supports the use of ensembles of models as a tool to improve the performance and robustness of the results.

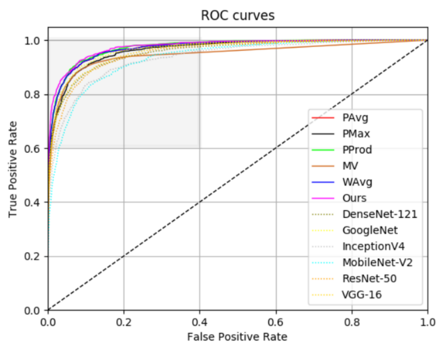
In Table II is presented the results achieved by the ensemble of deep models, considering the five aggregation methods and our proposed approach, for each medical imaging dataset. In general, the aggregation methods present better performance than the single models. We highlight the following observations for each dataset:

- ISIC: for this dataset, our method improved the average of the BACC in 5.4% comparing to the best single model and 1.2% comparing to the second-best aggregation method. In addition, it presents the lowest loss, which means it is predicting the correct label with more confidence.
- CheXpert: our method improved the average ACC in 3.9% comparing to the best single model. However, the BACC achieved by the ensemble is quite similar to the single models. Again, our approach presents the lowest loss among all aggregation methods.
- NCT: the results got by the ensemble are slightly better than the single models and all aggregation methods present similar performance for ACC and BACC. Nonetheless, for this dataset, the lowest average loss is achieved by PAvg.
- OCT: our method improved the average BACC in 1.2% comparing to the best single model and 1% comparing to the second-best aggregation method. As the models perform quite well for this dataset, the remaining metrics are similar.

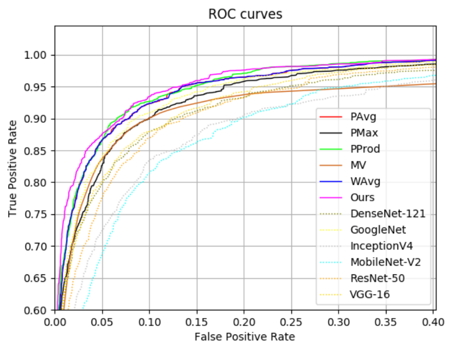
Overall, our method presents the lowest average loss for 3 out of 4 datasets and best average BACC for ISIC and OCT.

TABLE I: Performance achieved by each deep model for each medical imaging dataset.

Model	**ISIC**				**CheXpert**			
	ACC	**BACC**	**AUC**	**Loss**	**ACC**	**BACC**	**AUC**	**Loss**
DenseNet-121	0.720 ± 0.044	0.702 ± 0.055	0.932 ± 0.021	0.822 ± 0.155	0.676 ± 0.002	0.560 ± 0.006	0.781 ± 0.002	0.824 ± 0.015
GoogleNet	0.776 ± 0.035	0.758 ± 0.037	0.960 ± 0.010	0.622 ± 0.085	0.668 ± 0.002	0.561 ± 0.002	0.788 ± 0.001	0.830 ± 0.001
InceptionV4	0.778 ± 0.034	0.768 ± 0.040	0.959 ± 0.012	0.654 ± 0.103	0.656 ± 0.015	0.555 ± 0.003	0.768 ± 0.005	0.837 ± 0.015
MobileNet-V2	0.756 ± 0.047	0.738 ± 0.040	0.953 ± 0.010	0.688 ± 0.098	0.684 ± 0.008	0.549 ± 0.002	0.774 ± 0.001	0.817 ± 0.008
Resnet-50	0.768 ± 0.023	0.746 ± 0.026	0.956 ± 0.004	0.640 ± 0.049	0.685 ± 0.006	0.568 ± 0.009	0.794 ± 0.006	0.803 ± 0.010
VGG-16	0.745 ± 0.007	0.735 ± 0.033	0.944 ± 0.011	0.713 ± 0.050	0.675 ± 0.014	0.561 ± 0.020	0.789 ± 0.016	0.825 ± 0.010
Model	**NCT**				**OCT**			
	ACC	**BACC**	**AUC**	**Loss**	**ACC**	**BACC**	**AUC**	**Loss**
DenseNet-121	0.934 ± 0.014	0.909 ± 0.032	0.992 ± 0.001	0.403 ± 0.078	0.976 ± 0.002	0.956 ± 0.002	0.999 ± 0.000	0.064 ± 0.004
GoogleNet	0.924 ± 0.008	0.908 ± 0.003	0.989 ± 0.006	0.464 ± 0.191	0.968 ± 0.007	0.944 ± 0.013	0.997 ± 0.002	0.095 ± 0.026
InceptionV4	0.903 ± 0.005	0.883 ± 0.013	0.984 ± 0.006	0.502 ± 0.006	0.976 ± 0.001	0.957 ± 0.002	0.998 ± 0.000	0.073 ± 0.002
MobileNet-V2	0.935 ± 0.007	0.924 ± 0.001	0.993 ± 0.002	0.354 ± 0.023	0.957 ± 0.003	0.943 ± 0.006	0.995 ± 0.001	0.132 ± 0.017
Resnet-50	0.926 ± 0.004	0.920 ± 0.001	0.992 ± 0.001	0.442 ± 0.023	0.972 ± 0.002	0.952 ± 0.006	0.998 ± 0.000	0.081 ± 0.008
VGG-16	0.914 ± 0.014	0.911 ± 0.008	0.995 ± 0.002	0.392 ± 0.077	0.978 ± 0.002	0.960 ± 0.003	0.999 ± 0.001	0.065 ± 0.007



(a) Macro view



(b) Micro view

Fig. 3: The receiver operating characteristic (ROC) curves for the ISIC dataset considering the aggregation methods and the single models. In a) is depicted the full plot and in b) the zoom of the painted region of a).

Nonetheless, PAvg and WAvg present competitive performance considering all four sets of data. We also observe that the ensemble improves the AUC only for ISIC and CheXpert. Lastly, it is worth to note that the PProd and MV present the highest average loss among all methods. It happens because whenever there is a disagreement in the ensemble, these methods allow it to impact too much in the aggregation prediction.

To conclude this section, in Fig. 3 is shown the receiver operating characteristic (ROC) curves considering one folder for the ISIC dataset. As we may see, the aggregation methods present competitive performance, although our method's curve is slightly above the others. In general, the deep models' curve is below the aggregations' ones, except for the GoogleNet that is better than PMax and MV.

C. Dynamic pruning the ensemble

In this experiment, we online pruned the ensemble of deep models based on the weights assigned by our proposed method. For each new sample evaluated by the ensemble, we selected the best models through ranking the weights. We performed this experiment using only the ISIC dataset since the deep models present different results for this set. In Table III is presented the ensemble pruning that selects 1 to 5 models from the group. As the weights change according to the sample, the models selected may change each time the ensemble evaluated a new sample.

As we may note from Table III, the results present a similar performance when the ensemble is composed of at least three models, which suggests the weights assigned by the proposed approach are working properly. In addition, we observe that the loss decreases as the number of selected models increases. This result shows that even though it is possible to achieve similar performance, in terms of BACC, by pruning the ensemble, the number of models increases the prediction confidence. In Fig. 4 is shown the ROC curves considering each deep model and the ensemble pruning. We can see that the curves of the ensemble containing 1 and 2 models are slightly below the remaining ones, which is in accordance with the previous analysis.

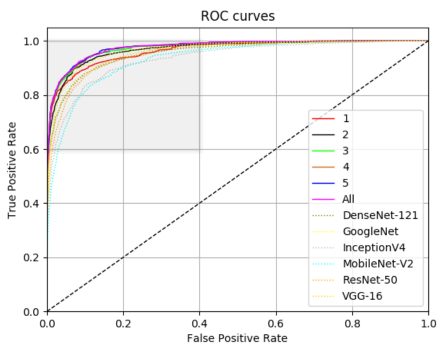
D. Discussion

The results presented in the previous section show that an ensemble of deep models worked properly to deal with medical imaging classification. Although the computational burden to train several deep models is high, the improvement presented is very desired, in particular for medical tasks.

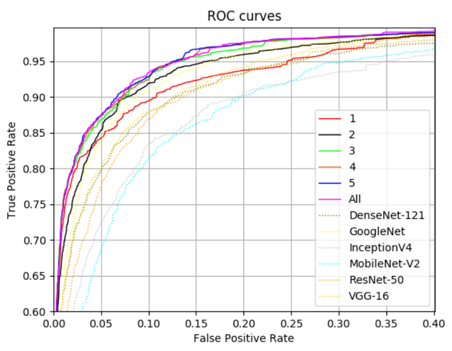
The proposed approach to aggregate the ensemble presented a competitive performance compared to the other methods. Beyond to achieve the best average BACC for two datasets,

TABLE II: Performance achieved by the ensemble of deep models for each medical imaging dataset considering the different aggregation methods.

Method	ISIC				CheXpert			
	ACC	BACC	AUC	Loss	ACC	BACC	AUC	Loss
PAvg	0.822 ± 0.001	0.810 ± 0.005	0.964 ± 0.002	0.557 ± 0.031	0.697 ± 0.003	0.571 ± 0.004	0.800 ± 0.003	0.800 ± 0.003
PMax	0.810 ± 0.010	0.805 ± 0.011	0.957 ± 0.003	0.698 ± 0.043	0.715 ± 0.007	0.566 ± 0.002	0.788 ± 0.003	0.819 ± 0.001
PProd	0.825 ± 0.011	0.805 ± 0.010	0.976 ± 0.002	1.399 ± 0.023	0.695 ± 0.003	0.574 ± 0.005	0.797 ± 0.004	1.469 ± 0.014
MV	0.805 ± 0.012	0.795 ± 0.013	0.952 ± 0.003	2.546 ± 0.218	0.709 ± 0.001	0.567 ± 0.011	0.760 ± 0.006	3.689 ± 0.124
WAvg	0.823 ± 0.010	0.809 ± 0.016	0.974 ± 0.002	0.543 ± 0.032	0.695 ± 0.003	0.572 ± 0.004	0.790 ± 0.006	0.801 ± 0.004
Ours	0.828 ± 0.012	0.822 ± 0.014	0.975 ± 0.002	0.503 ± 0.022	0.724 ± 0.002	0.568 ± 0.003	0.801 ± 0.005	0.728 ± 0.004
Method	**NCT**				**OCT**			
	ACC	**BACC**	**AUC**	**Loss**	**ACC**	**BACC**	**AUC**	**Loss**
PAvg	0.941 ± 0.002	0.928 ± 0.002	0.996 ± 0.001	0.189 ± 0.015	0.980 ± 0.002	0.962 ± 0.002	0.999 ± 0.002	0.064 ± 0.004
PMax	0.940 ± 0.003	0.931 ± 0.003	0.995 ± 0.002	0.201 ± 0.013	0.979 ± 0.002	0.960 ± 0.003	0.999 ± 0.001	0.095 ± 0.003
PProd	0.942 ± 0.002	0.932 ± 0.003	0.995 ± 0.000	1.167 ± 0.051	0.980 ± 0.002	0.960 ± 0.004	0.998 ± 0.001	0.190 ± 0.033
MV	0.943 ± 0.001	0.931 ± 0.003	0.984 ± 0.002	0.782 ± 0.092	0.980 ± 0.003	0.962 ± 0.005	0.995 ± 0.001	0.198 ± 0.011
WAvg	0.932 ± 0.002	0.922 ± 0.002	0.996 ± 0.001	0.196 ± 0.003	0.980 ± 0.002	0.961 ± 0.005	0.998 ± 0.000	0.063 ± 0.003
Ours	0.942 ± 0.001	0.932 ± 0.004	0.996 ± 0.002	0.251 ± 0.002	0.981 ± 0.002	0.972 ± 0.004	0.999 ± 0.000	0.058 ± 0.007



(a) Macro view



(b) Micro view

Fig. 4: The receiver operating characteristic (ROC) curves for the ISIC dataset considering the ensemble pruning approach. In a) is depicted the full plot and in b) the zoom of the painted region of a).

TABLE III: Ensemble pruning according to the weights assigned by our method.

Nº of models	ACC	BACC	AUC	Loss
1	0.799 ± 0.012	0.812 ± 0.012	0.957 ± 0.005	0.723 ± 0.041
2	0.804 ± 0.011	0.814 ± 0.011	0.967 ± 0.003	0.590 ± 0.031
3	0.816 ± 0.011	0.820 ± 0.012	0.971 ± 0.003	0.538 ± 0.041
4	0.820 ± 0.011	0.819 ± 0.014	0.973 ± 0.004	0.520 ± 0.030
5	0.823 ± 0.014	0.822 ± 0.014	0.975 ± 0.003	0.510 ± 0.040
All	0.828 ± 0.012	0.822 ± 0.014	0.975 ± 0.002	0.503 ± 0.022

the method increases the confidence of the predictions since it presented the lowest loss among all methods. The advantage of providing online weights for each new evaluated sample is important towards understanding the real contribution of each model within the ensemble. In addition, it can be applied to ensemble pruning, as shown in the experiments.

Although the proposed method is promising, it presents a limitation. As it depends on the validation set to estimate the Dirichlet distributions of the hit and miss sets, the model is sensitive to this set. For instance, if the distribution of this set is too far from the test set, i.e., the real world, the approach performance may decrease. However, deep learning training also depends on this assumption, i.e., the data distribution used in the training phase is close to the one in which the model will be applied. In any case, it is important to ensure a representative validation set.

V. Conclusion

In this paper, we proposed a new algorithm to learn dynamic weights for an ensemble of deep learning models. This algorithm uses the Dirichlet distribution and Mahalanobis distance to estimate the relevance of a deep model within the ensemble. In order to test the proposed method, we carried out experiments using four different medical imaging tasks. The method achieved the best averaged balanced accuracy for 2 tasks and the lowest loss for all of them. Although the method is promising, we discussed a limitation regarding the dependence of the validation set. To conclude, we believe it is possible to improve the ensemble's performance by aggregating the models' weights instead of only working with the predictions. In the future, we aim to extend the presented method to deal with this case.

Acknowledgment

A. G. C. Pacheco would like to thanks the financial support of the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001; T. Trappenberg acknowledges funding by NSERC; R. A. Krohling thanks the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) - grant n.309729/2018-1 - and the Fundação de Amparo a Pesquisa e Inovação do Espírito Santo (FAPES) - grant n. 575/2018.

References

[1] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, p. 115, 2017.

[2] J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya *et al.*, "Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison," *arXiv preprint arXiv:1901.07031*, 2019.

[3] D. Ardila, A. P. Kiraly, S. Bharadwaj, B. Choi, J. J. Reicher, L. Peng, D. Tse, M. Etemadi, W. Ye, G. Corrado *et al.*, "End-to-end lung cancer screening with three-dimensional deep learning on low-dose chest computed tomography," *Nature medicine*, vol. 25, no. 6, p. 954, 2019.

[4] J. Sahlsten, J. Jaskari, J. Kivinen, L. Turunen, E. Jaanio, K. Hietala, and K. Kaski, "Deep learning fundus image analysis for diabetic retinopathy and macular edema grading," *arXiv preprint arXiv:1904.08764*, 2019.

[5] N. C. Codella, Q.-B. Nguyen, S. Pankanti, D. A. Gutman, B. Helba, A. C. Halpern, and J. R. Smith, "Deep learning ensembles for melanoma recognition in dermoscopy images," *IBM Journal of Research and Development*, vol. 61, no. 4/5, pp. 5–1, 2017.

[6] E. Valle, M. Fornaciali, A. Menegola, J. Tavares, F. V. Bittencourt, L. T. Li, and S. Avila, "Data, depth, and design: Learning reliable models for skin lesion analysis," *Neurocomputing*, 2019, in press.

[7] S. Qummar, F. G. Khan, S. Shah, A. Khan, S. Shamshirband, Z. U. Rehman, I. A. Khan, and W. Jadoon, "A deep learning ensemble approach for diabetic retinopathy detection," *IEEE Access*, vol. 7, pp. 150 530–150 539, 2019.

[8] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger, "Snapshot ensembles: Train 1, get m for free," in *International Conference on Learning Representations (ICLR)*, 2017, pp. 1–14.

[9] D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

[10] A. S. Britto Jr, R. Sabourin, and L. E. Oliveira, "Dynamic selection of classifiers — a comprehensive review," *Pattern recognition*, vol. 47, no. 11, pp. 3665–3680, 2014.

[11] T. T. Nguyen, A. V. Luong, M. T. Dang, A. W.-C. Liew, and J. McCall, "Ensemble selection based on classifier prediction confidence," *Pattern Recognition*, vol. 100, p. 107104, 2020, in press.

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[13] I. S. I. C. (ISIC), "ISIC archive," 2019, https://www.isic-archive.com. [Last accessed: 02 Jan 2020].

[14] Y. Xiao, J. Wu, Z. Lin, and X. Zhao, "A deep learning-based multi-model ensemble method for cancer prediction," *Computer Methods and Programs in Biomedicine*, vol. 153, pp. 1–9, 2018.

[15] R. M. Cruz, R. Sabourin, and G. D. Cavalcanti, "Dynamic classifier selection: Recent advances and perspectives," *Information Fusion*, vol. 41, pp. 195–216, 2018.

[16] J. M.-T. Wu, M.-H. Tsai, Y. Z. Huang, S. H. Islam, M. M. Hassan, A. Alelaiwi, and G. Fortino, "Applying an ensemble convolutional neural network with savitzky–golay filter to construct a phonocardiogram prediction model," *Applied Soft Computing*, vol. 78, pp. 29–40, 2019.

[17] F. Perez, S. Avila, and E. Valle, "Solo or ensemble? choosing a CNN architecture for melanoma classification," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 1–9.

[18] N. Gessert, M. Nielsen, M. Shaikh, R. Werner, and A. Schlaefer, "Skin lesion classification using ensembles of multi-resolution efficientNets with meta data," *arXiv preprint arXiv:1910.03910*, 2019.

[19] A. G. Pacheco, A.-R. Ali, and T. Trappenberg, "Skin cancer detection based on deep learning and entropy to detect outlier samples," *arXiv preprint arXiv:1909.04525*, 2019.

[20] B. Harangi, "Skin lesion classification with ensembles of deep convolutional neural networks," *Journal of Biomedical Informatics*, vol. 86, pp. 25–32, 2018.

[21] D. Štefka and M. Holeňa, "Dynamic classifier aggregation using interaction-sensitive fuzzy measures," *Fuzzy Sets and Systems*, vol. 270, pp. 25–52, 2015.

[22] A. G. Pacheco and R. A. Krohling, "Aggregation of neural classifiers using choquet integral with respect to a fuzzy measure," *Neurocomputing*, vol. 292, pp. 151–164, 2018.

[23] B. Quost, M.-H. Masson, and T. Denœux, "Classifier fusion in the Dempster–Shafer framework using optimized t-norm based combination rules," *International Journal of Approximate Reasoning*, vol. 52, no. 3, pp. 353–374, 2011.

[24] B. Krawczyk and M. Woźniak, "Untrained weighted classifier combination with embedded ensemble pruning," *Neurocomputing*, vol. 196, pp. 14–22, 2016.

[25] K. W. Ng, G.-L. Tian, and M.-L. Tang, *Dirichlet and related distributions: Theory, methods and applications*. John Wiley & Sons, 2011, vol. 888.

[26] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart, "The Mahalanobis distance," *Chemometrics and intelligent laboratory systems*, vol. 50, no. 1, pp. 1–18, 2000.

[27] D. V. Oliveira, G. D. Cavalcanti, and R. Sabourin, "Online pruning of base classifiers for dynamic ensemble selection," *Pattern Recognition*, vol. 72, pp. 44–58, 2017.

[28] R. M. Cruz, D. V. Oliveira, G. D. Cavalcanti, and R. Sabourin, "Fire-des++: Enhanced online pruning of base classifiers for dynamic ensemble selection," *Pattern Recognition*, vol. 85, pp. 149–160, 2019.

[29] I. J. Myung, "Tutorial on maximum likelihood estimation," *Journal of Mathematical Psychology*, vol. 47, no. 1, pp. 90–100, 2003.

[30] T. Minka, "Estimating a Dirichlet distribution," 2012, technical report, MIT. https://tminka.github.io/papers/dirichlet/minka-dirichlet.pdf [Last accessed: 25 November 2019].

[31] G. Ronning, "Maximum likelihood estimation of Dirichlet distributions," *Journal of Statistical Computation and Simulation*, vol. 32, no. 4, pp. 215–221, 1989.

[32] S. S. Dragomir, R. P. Agarwal, and N. S. Barnett, "Inequalities for beta and gamma functions via some classical and new integral inequalities," *Journal of Inequalities and Applications*, vol. 2000, no. 2, p. 504054, 2000.

[33] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *International Conference on Machine Learning*, 2016, pp. 1050–1059.

[34] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[35] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[36] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017, pp. 4278–4284.

[37] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

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

[40] J. N. Kather, J. Krisam, P. Charoentong, T. Luedde, E. Herpel, C.-A. Weis, T. Gaiser, A. Marx, N. A. Valous, D. Ferber *et al.*, "Predicting survival from colorectal cancer histology slides using deep learning: A retrospective multicenter study," *PLOS Medicine*, vol. 16, no. 1, p. e1002730, 2019.

[41] D. S. Kermany, M. Goldbaum, W. Cai, C. C. Valentim, H. Liang, S. L. Baxter, A. McKeown, G. Yang, X. Wu, F. Yan *et al.*, "Identifying medical diagnoses and treatable diseases by image-based deep learning," *Cell*, vol. 172, no. 5, pp. 1122–1131, 2018.