TÜBİTAK

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2021) 29: 2824 – 2839
© TÜBİTAK
doi:10.3906/elk-2105-36

Deep hyperparameter transfer learning for diabetic retinopathy classification

Mahesh S PATIL[1,*], **Satyadhyan CHICKERUR**[2], **Yeshwanth Kumar VS**[1], **Vijayalakshmi A BAKALE**[1], **Shantala GIRADDI**[1], **Vivekanand C ROODAGI**[1], **Yashaswini N KULKARNI**[1]

[1]School of Computer Science and Engineering, KLE Technological University, Karnataka, India
[2]Center for High Performance Computing, KLE Technological University, Karnataka, India

Received: 05.05.2021 • **Accepted/Published Online:** 09.08.2021 • **Final Version:** 04.10.2021

Abstract: The detection of diabetic retinopathy (DR) in millions of diabetic patients across the globe is a challenging problem. Diagnosis of retinopathy is a lengthy and tedious process, requiring a medical professional to assess the individual fundus images of a patient's retina. This process can be automated by applying deep learning (DL) technology given a huge dataset. The problems associated with DL are the unavailability of a large dataset and their higher training time. The DL model's best performance is achieved using set of optimal hyperparameters (OHPs) obtained by performing costly iterations of hyperparameter optimization (HPO). These problems can be addressed by using transfer learning (TL) technique in both DL model training and HPO. TL in HP tuning is the focus of this work. The authors study the applicability of EyePACS DR dataset's OHPs to other DR datasets, forming the basis of the research question addressed in this work. The DR classification is performed using a ResNet model trained on the EyePACS (kaggle) and Indian diabetic retinopathy image dataset (IDRiD) datasets. Various HPs tuned in this work are data augmentation configuration, number of layers, optimizers, data samplers, learning rate, and momentum. The authors demonstrate that EyePACS dataset's OHPs are suitable for training with IDRiD dataset without needing to tune HPs for IDRiD dataset from scratch. The OHPs for a task and their reusability is poorly reported in the literature. Therefore, the EyePACS DR dataset's OHPs reported here can be used by other researchers. Moreover, the researchers working on other DR datasets can also apply the same OHPs since they are reusable and no iterations of HPO are required. The OHPs are provided for both EyePAC and IDRiD datasets after being tuned from scratch, which can be used as starting point for HPO by others.

Key words: Hyperparameter optimization, transfer learning, diabetic retinopathy, augmentation, ResNet, Bayesian optimization

1. Introduction

About 420 million people are diagnosed with diabetes around the world [1]. This disease has doubled over the past 30 years and is expected to increase in the future. DR is a disease that affects the retina of diabetic individuals, where blood vessels deteriorate over time which leads to blindness if left untreated. DR has several stages such as mild, moderate, severe, and proliferate. DR diagnosis is a tedious manual process. It requires a trained ophthalmologist to look at the fundus images of suspected patients to determine whether or not the patient has retinopathy. Detecting DR at early stages is usually difficult as significant symptoms are not shown until a stage where the damage is irreversible, this also increases the chances of the inaccurate diagnosis. The

[*] Correspondence: elektrik@tubitak.gov.tr

DR detection and classification can be automated using DL technology and ensure early diagnosis among the diabetic patients.

The DL learning models can predict a wide range of features given huge and accurate data. Especially the deep residual neural networks (DRNN) have solved vanishing gradients problem [1], enabling more number of layers to be stacked in DL model compared to AlexNet and VGGNet. However, this increases training time and as solution TL can be used, i.e. as a pretrained model. In a neural network, the initial layers extract low-level features while the deeper layers extract high-level features [2]. Low-level features remain the same across many tasks and can be reused [2–6]. Therefore, the authors of the proposed work use an ImageNet pretrained ResNet model and train only the last fully connected layers, keeping all the other layers frozen. The experiments are conducted on the NVidia DGX-1 server.

The HPs decide the structure of model and the training setting for a DL model. An optimal setting of HPs can lead to better performance of DL model in terms of training speed and accuracy. However, researchers need to tune and find the optimal set of HPs, which is a time- and resource-consuming task. To speed up the tuning of HPs, researchers have proposed numerous HP optimization algorithms [7, 8]. Many researchers have tuned and proposed OHPs for various datasets and suggested the important set of HPs for consideration [9–11]. Moreover, earlier studies indicate that knowledge from HPs tuning one task can be reused in another similar learning task [12], i.e. TL of HP, and not much research is carried out in this direction [12]. Thus, it all leads to the following research questions addressed in the proposed work.

Research question 1: Is it possible to reuse (TL of HP) OHP settings of one DR dataset to train on another DR dataset and achieve approximately the same performance as obtained when using HP tuned from scratch?

Research question 2: If OHP settings of one dataset are directly reusable for another, then what are such OHP settings for DR classification using a ResNet DL model with TL?

Considering the research questions above, the authors in the proposed work tune the HPs such as data augmentation configuration, number of layers, optimizers, data samplers, learning rate, and momentum using techniques such as grid search and Bayesian optimization. The authors also treat data augmentation (Section 3.3) as one of the HPs, as it can influence the performance of the DL model. To address the research questions, the authors tune and obtain OHPs on EyePACS DR dataset and demonstrate that it can be directly reused to train the DL model on IDRiD dataset [13]. The authors then demonstrate that similar performance is achieved when HPs for IDRiD dataset are tuned separately from the scratch. This indicates that the proposed OHP settings of EyePACS DR dataset are transferable and can be reused by other researchers working on DR classification with various datasets, saving their time and resources. The following are the contributions of the proposed work:

- The authors demonstrate that the knowledge of various HPs tuned for EyePACS DR dataset is transferable and can be directly reused to train a DL model on another DR dataset achieving similar performance to that of HPs being tuned from scratch.
- A set of OHPs for training DL model with EyePACS and IDRiD DR dataset are proposed; these OHPs can be reused by researchers working on the same DR datasets.

The rest of the paper is divided into several sections. Section 2 discusses existing related studies in this domain. Section 3 briefly explains the data preparation. Section 4 discusses the methodology of HP fine-tuning with TL. The results are discussed in Section 5, while Section 6 concludes the paper.

2. Related work

DR image classification has been a popular research area. Previously, researchers designed algorithms to extract features from images [14, 15] and then used classifiers like support vector machines (SVMs). The reported accuracy of such algorithms are between 87.5% and 99.4% [14]. However, such hand-engineered feature extraction algorithms overfit and fail to generalize well [16]. The introduction of DL solved this problem since DL can automatically learn to extract the features from a large dataset and provide better generalization [17, 18]. Therefore, the current research focus is on DL for DR image classification.

2.1. DL for DR classification and TL

Various DL models used in earlier studies are Alexnet, GoogleLeNet, ResNet-50, Inception, Xception, Dense121, and Dense169 [17, 19]. To date, many researchers have performed binary classification (healthy-unhealthy) of DR images with good accuracy in their results [17, 20–23]. However, only a few previous studies have reported multiclass classification [17] and the reported accuracy is relatively low. Table 1 gives various DL models used in DR classification along with results that are reported in the literature. The DL model training time are high (days to a week) and require a huge dataset. This problem can be solved using TL. TL is a technique of reusing the knowledge gained in one domain (source) into another (target domain). It is suitably applied where datasets and the computing resources are limited [2, 24, 25]. The authors in the proposed work apply Network-based TL, i.e. a neural network being trained in one source domain used as a pretrained network in a new task. The authors use ImageNet pretrained ResNet DL model which is stacked with fully connected (FC) layers. Furthermore, only the FC layers are trained on the new dataset.

Table 1. Results reported by previous work using DL for DR classification.

Research work	DL model	Classification type, results (accuracy)
Lam et al. [17]	AlexNet	Binary, 60%; 3-class, 57%
	GoogLeNet	4-class, 50%; binary, 73%; 3-class, 67%
Gulshan et al. [21]	CNN	Binary, 99%
Khaled et al. [19]	CNN	Binary, specificity, 96.1%
Islam et al. [26]	-unknown-	Binary, 98%
Shankar et al. [9]	Hyperparameter tuning inception-v4	4-class, 99%
Gargeya et al. [16]	custom DL model	Binary, 0.97 AUC
Shankar et al. [27]	Synergic deep learning	4-class, 98.58%
Qummar et al. [28]	ensemble of DL models	5-class, 80.8%
Hacisoftaoglu et al. [29]	ResNet	Binary, 98.6%

2.2. HPO techniques

The DL model training begins by choosing a set of the HPs such as learning rate, momentum rate, batch size, number and type of layers etc. These HPs influence the training speed and accuracy of the DL model. Choosing an appropriate set of HPs is a challenging and time-consuming task. Therefore, various HP tuning techniques are used, which are given below. [7, 30].

Equation 1 depicts HPO [9]

$$h = argmax_{h \in H} f(h) \tag{1}$$

In Equation 1, *f(h)* is the objective function to be fine-tuned and in the proposed work, *f(h)* is used for DR classification. *h** is a set of HP values from the domain *H*. The objective is to find the *h* that can maximize the *f(h)* in terms of either accuracy, speedup, or any other performance metric of DL training.

2.2.1. Grid search

Grid search and random search techniques are widely used due their ease of implementation and understanding [8, 31]. In grid search, a set of HPs are chosen with a range of values. Then, the DL model is trained with every combination of HPs to find the best one. The disadvantage is the increasing HP dimension and its exponential increase in combinations which would be computationally expensive to compute. The authors in this proposed work use similar approaches to that of grid search as discussed in Section 5.1 .

2.2.2. Random search

It is used when the number of HPs is higher and it is relatively less intensive than grid search [8]. This algorithm chooses a set of HP combinations for evaluation. Unlike the grid search, this algorithm chooses any value for a given HP randomly from the given range. The random search also tends to perform better than Bayesian optimization in some cases [8].

2.2.3. Sequential model-based global optimization (SMBO)

a) Gaussian process (GP): This technique has the context of Bayesian optimization [32, 33]. The function f from Equation 1 is a GP i.e. $p(\ f(h))$. Let S be some set of n samples where $S = (h_i, f(\ h_i\ ))$, i is the particular sample number. Now, when f is GP then S is also a GP i.e. $p(\ h|f(h)\ )$ [7]. Now using Bayesian optimization, the optimized set HPs can be predicted and DL can be trained (costly step) with predicted h^* from S. i.e. modelling $p(\ f(h)|h)$. Every time training is done, the pair of h_i and obtained $f(h_i\ )$ can be added to sample set S which can be used for making further predictions. More details can be found in [7, 33].

b) Tree-structured Parzen estimator (TPE) approach: While the GP tries to model $p(\ f(h)|h)$, the TPE technique models $p(\ h|f(h))$ instead. The $p(\ h|f(h))$ is modelled as nonparametric densities with distributions such as uniform, log/log quantized uniform, and categorical variables. More details can be found in [7]. The authors use this HPO technique to tune learning rate and momentum (Section 5.1).

The HPO has received low priority in the reported literature and many have not reported OHP settings of their experiments thus making it difficult to replicate the same experiments. Therefore, researchers are forced to tune HP from scratch, which is both time- and resource-consuming leading to poor performances [12]. The abovementioned HPO techniques need to iterate several tens of times before finding the OHP setting. In each iteration, the learning model is trained with chosen HP settings. However, when the dataset size is huge, such HPO techniques can be very costly; therefore, it is essential to obtain OHP settings in a few iterations. All these points lead to the following research gap which can be addressed by HP TL [34].

Research gap-1: There is necessity to speed up HP tuning through TL and publish OHP settings for a task that can be reused.

The above research gap can be addressed by transferring the HP-tuning knowledge from one task to another [12, 35] and in the proposed work, the authors provide EyePACS's OHP settings which can be directly applied as HP to train a DL model on another dataset.

2.3. TL in HP tuning

One approach of TL in HP tuning involves reusing the previous task's HP tuning knowledge to tune HPs for a new task. Each learning task will have an important set of HPs that can impact its performance and such set of HPs remain the same across the dataset [11]. The second approach involves tuning the HPs simultaneously for multitasks using Bayesian based optimizations. TL is applied by utilizing the extent of correlation between the tasks to speedup the HP optimization process. For example, adaptive Bayesian linear regression (ABLR) model [36] uses a neural network (NN) and Bayesian linear regression surrogate function to learn HPO from multitasks. The NN takes HPs' related information from multitasks as shared feature map. Then the NN output is used by separate surrogate functions for each task to predict next HP settings. The authors in [34] use HP tuning data of previous tasks to tune HPs for newer multiple tasks. In every iteration, one among the multiple datasets is used to tune and update the HP per dataset. In updating the surrogate model, a response surface value per dataset is used which depicts the deviation from previous iterations' HP data. This work is further improved by limiting the search region of HPs in [37], i.e. two search region constraints, box and ellipsoid, are proposed for HPO. A problem associated with these HP TL approaches is overfitting issues. The multitask Bayesian optimization with adaptive complexity, ABRAC, (similar to ABLR) solves this overfitting issue by introducing nested dropout as regularization in the NN used for HPO.

All the abovementioned multitask HPO methods can provide speedup only when HPs need to be optimized for multiple tasks which is not widely required. Generally, HPs are tuned individually for a particular dataset (like EyePACS dataset) and not many HPO techniques utilizing TL are reported in literature [9, 10, 38, 39] for the same. A DL model used in a task (like ResNet) can give different accuracy when trained on different datasets (like EyePACS and IDRiD). This accuracy depends on whether the dataset is balanced or skewed. However, the OHPs for such datasets will be in the nearby region and are thus transferable [34]. Therefore, the authors in the proposed work attempt to study the adaptability of EyePACS's OHP settings to other dataset leading to following research gap.

Research gap-2: In the DR classification, the adaptability of EyePACS dataset's OHP to another DR dataset needs to be demonstrated so that it can be directly reused to train the DL model on another DR dataset.

Addressing the above research gap, the authors in the proposed work apply EyePACS OHP setting to train IDRiD dataset and show that similar performance is achieved when compared to HP being tuned from scratch for IDRiD dataset.

3. Data preparation

3.1. Dataset

The proposed work uses EyePACS dataset (kaggle competition dataset) to demonstrate TL with OHP by its direct reuse on another dataset. The training and validation set contains 35,126 and 53,576 high-resolution fundus images of patients' eyes as depicted in Table 2 with number of samples for each class label. Whether the image is patient's left/right eye information is also given. The images are collected from different sources being captured with different cameras. The images have significant variations such as visual appearance, inverted, and noises like out of focus, external artefacts, overexposure and underexposure. The validation dataset provided by Kaggle is used as is and it is larger than the training set. This provides more accurate and generalized results of the DL model. Sample images are given in Figure 1.

Table 2. EyePACS dataset class distribution.

Class	Name	Original training dataset		Original validation dataset		Augmented training dataset	
		No.of images	Percentage	No. of images	Percentage	No. of images	Percentage
0	No DR	25810	73.48%	39533	73.48%	25810	37.40%
1	Mild DR	2445	6.96%	3762	7.02%	12215	17.70%
2	Moderate DR	5292	15.07%	7861	14.67%	10584	15.33%
3	Severe DR	873	2.48%	1214	2.26%	10476	15.18%
4	Prolific DR	708	2.01%	1206	2.25%	9912	14.36%

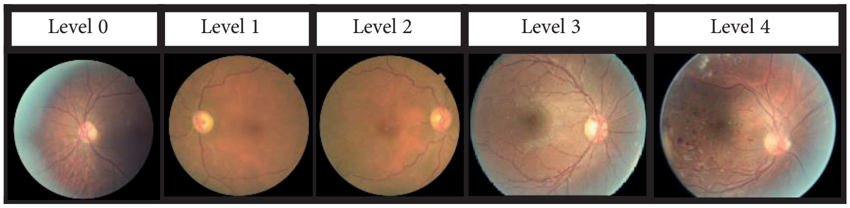

Figure 1. Fundus images with various levels of DR.

3.2. Image preprocessing

The image processing steps reported [40] by Kaggle DR competition winner BT Graham are used in the proposed work. Firstly, the images are clipped to 90% of their original size to remove the boundary effects produced by microscopes. Next, the local average color of each image is mapped to 50% gray which removes the visual variations produced by various cameras and finally scaled to resolutions of 512 x 512 pixels. These steps are applied to both training and validation dataset as depicted in Algorithm 1. The authors also tried with colored images, but it did not improve the performance of the model.

Algorithm 1: Algorithm for preprocessing the image

```
Load the dataset
foreach  $image \in dataset$  do
    Clip the outer 10% of the image
    Subtract local mean color of the image
    Rescale the image to 512x512 pixels
end
Result: all the images in the dataset are preprocessed
```

3.3. Data augmentation

Table 2 shows that the dataset is highly imbalanced, which is not good for a deep learning model. If an imbalanced dataset is provided to a neural network for training, it may cause overfitting and also might increase the chances of inaccurate predictions. Therefore, it is essential to balance the dataset. To do this, the authors augmented the training dataset based on the class of images as suggested by the DR Kaggle competition winner BT Graham in his competition report [40]. In the augmentation, the images are rotated to a random angle between 0 and 360 degrees keeping the aspect ratio intact. The images are further randomly flipped vertically, horizontally or both. Finally, the images are also skewed by +2%. After augmenting the original training dataset, the number of images in the training dataset is increased to 68,997. Various augmentation configurations are tested (Section 5.1), and the augmented configuration given in Table 2 yielded the best results. A sample image with augmentation is shown in Figure 2.

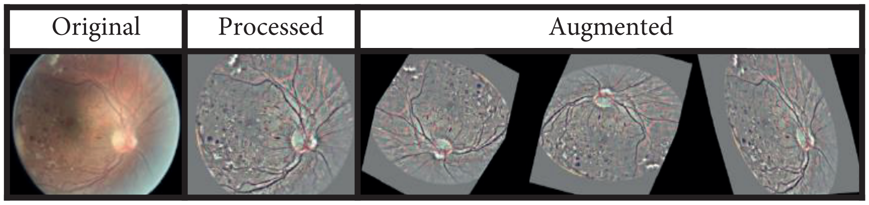

Figure 2. Augmentation example.

4. Methodology

4.1. DL model

A ResNet-50 model pretrained on ImageNet is used in this proposed work for all the experiments. The variants of the ResNet model consist of 18 to 158 layers, and ResNet-50 is chosen considering its performance and the time required to train. The ResNet-50 consists of 5 blocks stacked, and each block consists of 3 convolution layers. The pretrained model consist of 5 such blocks to which authors stack 3 dense layers with parametric ReLU (PReLU) activation for the first two layers. A dropout layer is also used to prevent overfitting on the dataset, as shown in Figure 3. The final layer is connected to the softmax layer. The optimizers used cross-categorical entropy loss. The authors use learning rate scheduler and Kaiming-Uniform method [41] as dense layer weight initializer. The batches of 32 images selected by random sampler is used in both training and validation. The authors train only the FC layers in all the experiments.

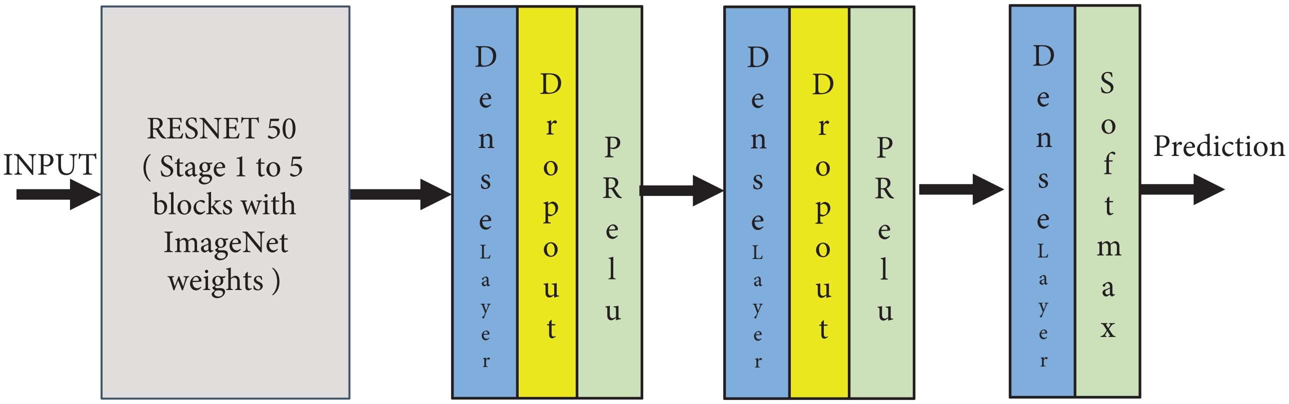

Figure 3. ResNet-50 with dense layers used in the proposed work.

4.2. Experiments

Overview of all the experiments conducted in the proposed work is presented in Figure 4. To address the research question stated in Section 1, the authors have conducted the following experiments.

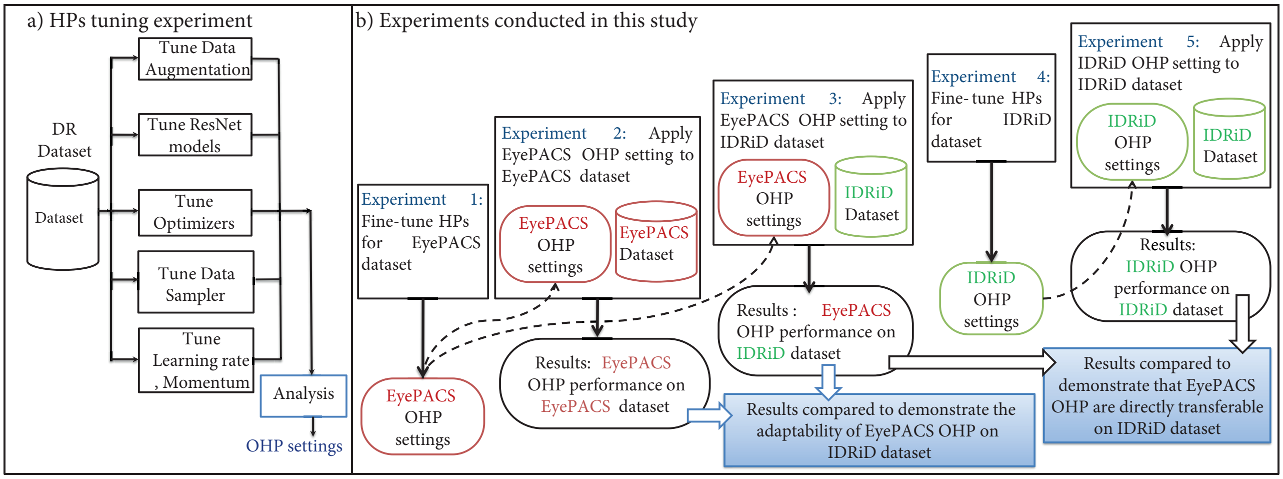
a) HPs tuning experiment
DR Dataset
Dataset
Tune Data Augmentation
Tune ResNet models
Tune Optimizers
Tune Data Sampler
Tune Learning rate , Momentum
Analysis
OHP settings
b) Experiments conducted in this study
Experiment 1: Fine-tune HPs for EyePACS dataset
EyePACS OHP settings
Experiment 2: Apply EyePACS OHP setting to EyePACS dataset
EyePACS OHP settings
EyePACS Dataset
Results: EyePACS OHP performance on EyePACS dataset
Experiment 3: Apply EyePACS OHP setting to IDRiD dataset
EyePACS OHP settings
IDRiD Dataset
Results : EyePACS OHP performance on IDRiD dataset
Results compared to demonstrate the adaptability of EyePACS OHP on IDRiD dataset
Experiment 4: Fine- tune HPs for IDRiD dataset
IDRiD OHP settings
Experiment 5: Apply IDRiD OHP setting to IDRiD dataset
IDRiD OHP settings
IDRiD Dataset
Results: IDRiD OHP performance on IDRiD dataset
Results compared to demonstrate that EyePACS OHP are directly transferable on IDRiD dataset

Figure 4. Overview of experiments conducted in the proposed work.

4.2.1. Experiment 1- Fine tune HPs for EyePACS dataset

The authors first fine-tune HPs and obtain OHPs for EyePACS dataset since it is not reported in the literature. At the end of this experiment, the OHPs for EyePACS dataset which are required in the next experiments is obtained. The authors considered the following HP for tuning.

i. Data augmentation configurations: The original dataset, as depicted in Table 2, is a highly imbalanced dataset. Therefore, the authors have augmented the data as explained in Section 3.3 to avoid the overfitting problem. The following are augmentation configurations experimented by the authors.

- Configuration 1: one set with no augmentation,
- Configuration 2: one set with approximately 10,000 images for each class,
- Configuration 3: one set as described in Table 2,
- Configuration 4: one set with approximately 25,000 images of each class.

The DL model is trained separately using the abovementioned configurations of dataset and best configuration is selected based on performance.

ii. ResNet models: The ResNet architecture has various configurations with different numbers of layers. Thus, the DL models ResNet-34, ResNet-50, and ResNet-101 are trained separately, and the best model is selected based on performance.

iii. Optimizers: The authors use RMSProp, SGD, SGD with Nesterov Momentum and Adam for comparison. The DL model is trained separately with these optimizers to choose the best one.

iv. Datasampler: The bias of the model is dependent on the type of data sampling; therefore, a proper sampling technique must be equipped while training a model. Choosing batches of data from the whole dataset is done by a data sampler. In this experiment, the performance of weighted sampler (WS) and random sampler (RS) with a ResNet-50 model are compared.

v. Learning rate and momentum: Learning rate and momentum is tuned using the Bayesian optimization library [7] as depicted in Algorithm 2.

Algorithm 2: Algorithm for the experiment on learning rate and momentum rate optimization using Bayesian optimization

```
Load the DR config3 dataset
Set bayesian optimization exploration range for Learning rate from 0.001 to 0.01
Set bayesian optimization exploration range for Momentum rate from 0.5 to 0.9
Prepare the ResNet-50 model for training with ResNet SGDNM optimizer
$bayesIterations \leftarrow 10$
foreach $bItr \in bayesIterations$ do
    $epochs \leftarrow 50$
    Predict and set next learning rate and momentum rate using tree-structured Parzen estimator
     approach (Section 2.2 )
    foreach $i \in epochs$ do
        train and update the model
        validate the model
        store the results for further analysis
    end
end
Determine the best learning rate and momentum rate by analyzing the stored results
Result: Best learning rate and momentum rate for ResNet-50 model is selected
```

4.2.2. Experiment 2, applying EyePACS OHP setting to EyePACS dataset

Once the EyePACS OHPs are obtained, it is applied to train the DL model on a dataset to record the OHP settings performance on the dataset. In this experiment, the EyePACS OHPs are applied to EyePACS dataset and the observed performance is used for comparison. In the Section 5.3, the authors demonstrate that this set of OHPs adapt to IDRiD dataset and are also transferable to other DR datasets through direct reuse of the OHPs.

4.2.3. Experiment 3, applying EyePACS OHP setting to IDRiD dataset

In this experiment, the authors test the adaptability of EyePACS OHPs on another DR dataset. A small DR dataset IDRiD is used to train the DL model by applying EyePACS OHPs directly (i.e. no HP tuning is done in this experiment). The observed performance is then compared with that in experiment 2 to demonstrate the adaptability of EyePACS OHP settings to IDRiD dataset.

4.2.4. Experiment 4, fine-tuning HPs for IDRiD dataset

It is necessary to cross-check the EyePACS-OHP+IDRiD-dataset performance with IDRiD datasets own OHPs. This will indicate the extent of adaptability of EyePACS OHPs to other datasets, thereby indicating the direct reusability of EyePACS OHPs on other dataset. To do so, in this experiment, the authors tune HPs for IDRiD dataset from scratch, which is similar to experiment 1. At the end of this experiment, IDRiD datasets' own OHP setting is obtained.

4.2.5. Experiment 5, fine-tuning HPs for IDRiD dataset

In this last experiment, the DL model is trained with IDRiD dataset applying IDRiD's own OHPs. This result is compared with those of EyePACS-OHP+IDRiD-dataset (experiment 4) to demonstrate that similar performance is achieved in both cases. This result comparison shall indicate that the EyePACS OHPs are transferable to other datasets and achieve similar performance to that of other datasets' own OHPs. Therefore, HPs do not need to be tuned for other DR datasets and EyePACS OHPs can be directly reused on other DR datasets as well.

4.3. Software and hardware

The software packages used in the proposed work are Pytorch [42], Numpy, Pandas, OpenCV, PIL, and Glob. The authors have used the pretrained ResNet model provided by Pytorch. For image preprocessing operations, the authors have used OpenCV as it provides additional functionality. All of these are installed and executed on an Ubuntu-based Nvidia DGX 1 server with 32GB Tesla V100 GPUs.

5. Results and discussion

This section discusses and analyzes the results of all the experiments that are explained in Section 4.2.

5.1. HP tuning for EyePACS datasets: Experiment 1

The following are results of HPs tuned for EyePACS dataset.

i. Data augmentation configurations: As explained in Section 4.2, the results of 4 dataset augmentation configurations are analyzed. The result of this subexperiment is as shown in Figure 5. Configuration 1 resulted in overfitting while configuration 2 and 4 had relative low bias and low variance. Configuration 3 had relative low bias and low variance and performed the best with a validation accuracy of 77.17% and a training accuracy of 83.38% under multiclass classification.

ii. ResNet models: The results of three ResNet models training are as shown in Figure 5. ResNet-34 and ResNet-101 have similar performances in the training phase, whereas ResNet-50 performed worse than the other two. In the validation phase, the performances of ResNet-34 and ResNet-50 are similar, while ResNet-101 performed worse. The overall performance of ResNet-50 is the best compared to other two with a validation accuracy of 77.16% and a training accuracy of 80.11%. The difference in the overall performances of all three models are minor.

iii. Optimizers: The results of four optimizers are shown in Figure 5. As stated in the paper [43], SGD is better in generalizing compared to adaptive optimizers (Adam and RMSProp). Though adaptive optimizers provide a greater training accuracy than SGD in some cases, validation accuracy of SGD is higher. In the proposed work, SGD performed better than other adaptive optimizers in both training and validation phases. Adding Nesterov Momentum to SGD (SGDNM) showed a slight improvement of training accuracy, whereas the improvement in validation accuracy is negligible. The validation accuracy of SGDNM is 77.01% and its training accuracy is 76.7%. RMSProp had the lowest training and validation accuracies out of all the other optimizers. Its graph is too erratic and therefore not included in the results graphs.

iv. Datasampler: The result of the subexperiment is as shown in Figure 5. Though WS is specifically designed for imbalanced datasets, its overall performance was poor compared to RS. RS and WS have similar performances in the training phase. However, during validation, RS performed significantly better than WS with a validation accuracy of 74.37% and a training accuracy of 90.28%.

v. Learning rate and momentum: The range of exploration for learning rate is from $1e^{-}4$ to $1e^{-}2$ and range for exploration for momentum is from 0.5 to 0.9. Ten iterations and 10 steps of random exploration are performed before finding the maximum. With the optimized values, the model converges to a validation accuracy of 76% within the first 10 epochs, it might happen anytime before 10 epochs as the data sampling is random. The values provided by the algorithm after optimization are as follows:

$$Learningrate = 0.006913899455031393 \text{ and } Momentumrate = 0.8214711456880412$$

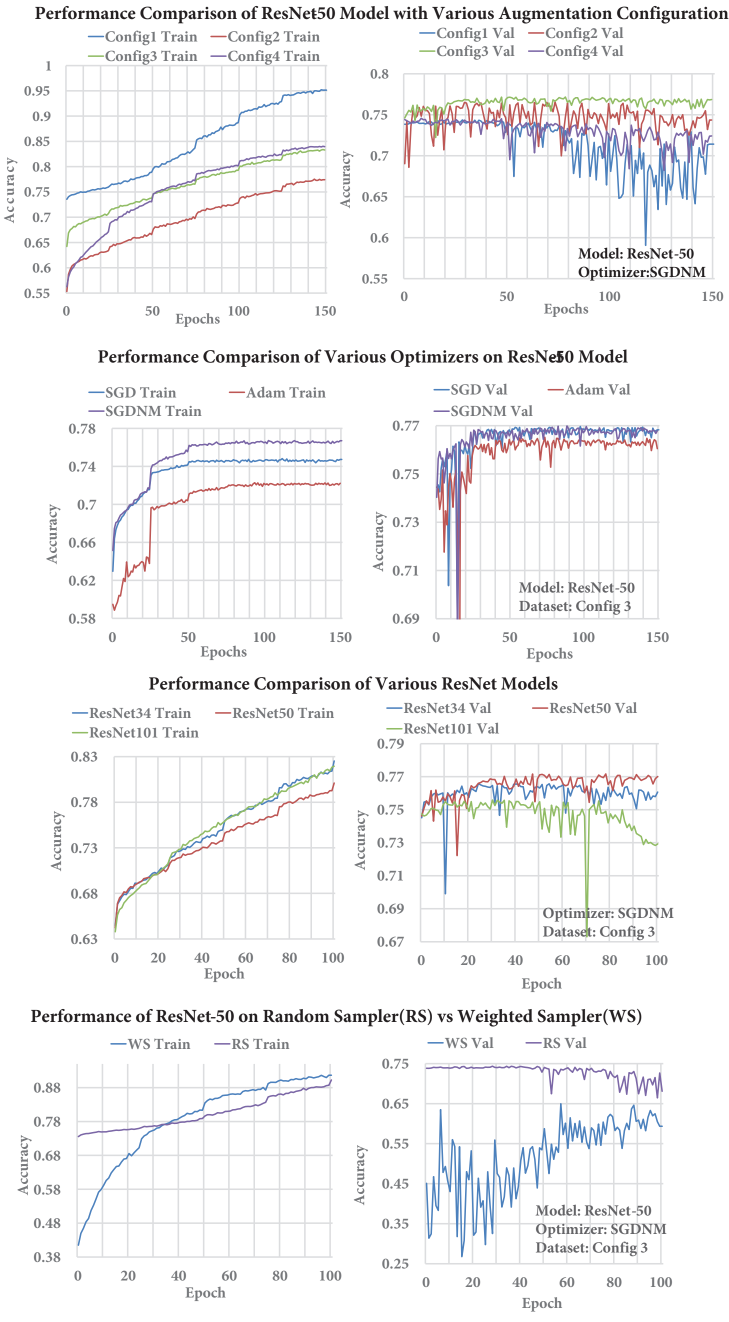

Figure 5. Results of EyePACS datasets HP tuning with respects to data augmentation, ResNet models, optimizers, and data samplers.

The following are the EyePACS dataset's OHPs, obtained at the end of this experiment:

i. Data augmentation: configuration 3,

ii. Learning model: ResNet-50,

iii. Optimizer: SGDNM,

iv. Data sampler: random sampler,

v. Learning rate and momentum: 0.006913 and 0.82147.

Applying the EyePACS OHPs to EyePACS dataset (Experiment 2): The obtained EyePACS OHP settings are applied to train the EyePACS dataset and the observed performance is used as base result for comparison. In the binary classification mode, training and validation accuracies of 90.65% and 80.27% are obtained respectively. In the multiclass classification, training and validation accuracies of 80.11% and 77.17% are obtained respectively. This result, shown in column 4 of Table 3 is used for comparison with the results obtained in Experiment 3.

5.2. HPs tuning for IDRiD datasets: Experiment 4

The authors have also fine-tuned the HPs for IDRiD dataset from scratch. The following are the best HP settings obtained:

i. Augmentation configuration: configuration 4, one set with approximately 25,000 images of each class

ii. Layers: ResNet-50

iii. Optimizers: stochastic gradient descent (SGD)

iv. Learning rate: 0.00503

v. Momentum: not applicable since SGD optimizer without Nesterov momentum is found to be the best.

5.3. Transferring EyePACS dataset's OHP to IDRiD dataset training

The comparison of results explained below address the research question 1 mentioned in Section 1.

Table 3. Performance comparison of OHPs on EyePACS and IDRiD datasets.

Classification	Dataset type	Performance parameters	EyePACS'S OHP on EyePACS dataset (Experiment 2)	EyePACS's OHP on IDRiD dataset (Experiment 3)	IDRiD OHP on IDRiD dataset (Experiment 5)
Binary classification	Train	Highest Acc	90.65%	94.32%	95.78%
		Epochs	100	93	91
	Validation	Highest Acc	80.27%	80.58%	81.55%
		Epochs	94	13	7
Multiclass classification	Train	Highest Acc	80.11%	85.88%	90.64%
		Epochs	100	100	100
	Validation	Highest Acc	77.17%	63.05%	62.14%
		Epochs	78	73	60

5.3.1. Performance of EyePACS datasets OHP

The column 4 (Experiment 2) and 5 (Experiment 3) of Table 3 depicts the performance of EyePACS's OHPs being applied to train the EyePACS and IDRiD datasets. In binary and multiclass classification, the IDRiD dataset achieves its highest train and validation accuracy in less number of epochs compared to EyePAC dataset. Since IDRiD was a smaller dataset, the model overfits soon; therefore, the observed train accuracy is higher when compared to EyePAC dataset. The validation accuracies in binary classification are similar, but it is comparatively low in IDRiD mulitclass classification, which can be attributed to the small dataset size. However, it is the best validation accuracy for IDRiD dataset when compared to performance of IDRiD's own OHPs (Experiment 5). All these observations indicate that EyePAC OHPs adapt well to IDRiD dataset and can be directly reused for other DR datasets as well.

5.3.2. Performance of transferred OHPs vs dataset's own OHPs

The authors finally compare the performance of the DL model trained using IDRiD dataset applying EyePACS's OHP (i.e. transferring OHPs) and model trained on IDRiD dataset's own OHP (Section 5.2). The results for the same are depicted in columns 5 (Experiment 3) and 6 (Experiment 5) of Table 3. In the binary classification, both EyePACS and IDRiD OHPs produce similar performances in training and validation phases with IDRiD OHP performing slightly better. Moreover, IDRid OHPs achieves the highest accuracy in less epochs. In multiclass classification, the IDRiD OHPs give better training accuracy. However, the validation accuracy is slightly higher with EyePACS OHPs, but it takes more epochs to obtain it. Overall ,the IDRiD's own OHPs perform slightly better than EyePACS OHPs but it is achieved at the cost of fine-tuning the HPs from scratch. Since EyePAC OHPs yield a similar performance, it would be wise to directly reuse EyePACS OHPs on another DR dataset. Thus, such TL of OHPs avoid fine-tuning the HPs on another DR dataset, saving both time and resources.

5.3.3. Comparing the proposed work with state of the art (SOTA)

To the best of our knowledge, the proposed idea of direct reuse of OHPs has not been reported. However, algorithms for multitask HPO with TL are reported, and the results are compared with the same as shown in Table 4. One point to be noted is that, in all the SOTA algorithms, up to 5–10 iterations of HPO have to be performed, which can be very costly operations for DR classification with DL. However, with the proposed idea of DR OHPs' reusability, iterations of HPO are not required for a new DR dataset.

Overall the proposed work demonstrates that the EyePACS's OHP settings are transferable and can be directly applied to other dataset without the need of further tuning. However, the research can still use the proposed OHP settings as starting point to optimize the HPs for other DR datasets. Considering the idea of the proposed work, similar works can be carried out for other learning tasks in the future.

6. Conclusion

This paper presents the research work of transferring HPs knowledge from one DR dataset to another DR dataset in training the DL model. This work also provides the OHPs to help other researchers to start their DL model training on EyePACS dataset and IDRiD dataset. Sustainability of using OHPs of EyePACS dataset on the IDRiD dataset is indicated through the experiments, this explains that the same OHPs can be reused for other DR datasets as hyperparameter transfer learning(HPTL). These OHPs can also be used as starting

Table 4. Comparison of the proposed work with SOTA.

Research work	HPO approach	Results	Iterations in HPO
Yogatama et al. [34]	Multitask HPO, Gaussian process model	In the Ranking metrics, achieves lower rank in most iterations with SOTA algorithms at the time	-Not applicable-
Perrone et al. [36]	Multitask HPO, Bayesian linear regression	2x-3x speed compared to SOTA at the time	5 iteration
Perrone et al (2019). [37]	Multitask HPO, search region constraints imposed on HP	HPO done in 2-8 iteration compared to 15-30 for SOTA algorithms at the time	2-8 iterations
Horvath et al. [44]	Multi-Task HPO, Bayesian linear regression with drop-outs in NN	In the Ranking metrics, achieves rank 1 in all the iterations compared with SOTA algorithms at the time	-Not applicable-
Proposed direct reuse of EyePAC OHP	To tune HPs for a large dataset for task and reuse the obtained OHP directly on other dataset.	EyePAC OHPs applies well to IDRiD dataset giving similar performance compared to IDRiD own OHP.	No iteration required since OHP are reusable directly.

point and tune them further for a DR dataset. In the future scope of work, authors are trying to use HPTL with Bayesian optimization algorithm for DR classification.

Acknowledgment

The authors are grateful for the support given by the management of the KLE Technological University, Hubballi, for carrying out this research work.

References

[1] Kaiming H, Zhang X,Ren S, Sun J. Identity mappings in deep residual networks. In: European Conference on Computer Vision. Springer, Cham; 2016. pp. 630-645.

[2] Xiaogang L, Tiantian P, Biao X, Weixiang L, Liang P et al. Convolutional neural networks based transfer learning for diabetic retinopathy fundus image classification. In: IEEE 2017 10th International Congress on Image and Signal Processing, Biomedical Engineering and Informatics (CISP-BMEI); 2017. pp. 1-11.

[3] Sharif Razavian A, Hossein A, Josephine S, Stefan C. CNN features off-the-shelf: an astounding baseline for recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops; 2014. pp. 806-813.

[4] Oquab M, Leon B, Ivan L,Josef S. Learning and transferring mid-level image representations using convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2014. pp. 1717-1724.

[5] Tamaki T, Junki Y, Misato K, Bisser R, Kaneda K et al. Computer-aided colorectal tumor classification in NBI endoscopy using local features. Medical Image Analysis 2013; 17(1): 78-100.

[6] Donahue J, Jia Y, Vinyals O, Hoffman J, Zhang N et al. Decaf: A deep convolutional activation feature for generic visual recognition. In: International Conference on Machine Learning; PMLR; 2014. pp. 647-655.

[7] Bergstra J, Bardenet R, Bengio Y, Kégl B. Algorithms for hyperparameter optimization. Advances in Neural Information Processing Systems 2011; 24.

[8] Bergstra J, Bengio Y. Random search for hyper-parameter optimization. Journal of Machine Learning Research 2012; 13 (2).

[9] Shankar K, Zhang Y, Liu Y, Wu L, Chen CH. Hyperparameter tuning deep learning for diabetic retinopathy fundus image classification. IEEE Access 2020; 8: 118164-118173.

[10] Mohammadian S, Karsaz A, Roshan YM. Comparative study of fine-tuning of pre-trained convolutional neural networks for diabetic retinopathy screening. In: IEEE 2017 24th National and 2nd International Iranian Conference on Biomedical Engineering (ICBME); 2017. pp. 1-6.

[11] Van Rijn JN, Hutter F. Hyperparameter importance across datasets. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining; 2018. pp. 2367-2376.

[12] Swersky K, Snoek J, Adams RP. Multi-task Bayesian optimization. Advances in Neural Information Processing Systems 2013; 26: 2004-2012.

[13] Porwal P, Pachade S, Kamble R, Kokare M, Deshmukh G et al. Indian diabetic retinopathy image dataset (IDRiD): a database for diabetic retinopathy screening research. Data 2018; 3 (3): 25.

[14] Verma K, Deep P, Ramakrishnan AG. Detection and classification of diabetic retinopathy using retinal images. In: 2011 Annual IEEE India Conference; 2011. pp. 1-6).

[15] Akram MU, Khalid S, Khan SA. Identification and classification of microaneurysms for early detection of diabetic retinopathy. Pattern Recognition 2013; 46 (1): 107-16.

[16] Gargeya R, Leng T. Automated identification of diabetic retinopathy using deep learning. Ophthalmology. 2017; 124 (7): 962-969.

[17] Lam C, Yi D, Guo M, Lindsey T. Automated detection of diabetic retinopathy using deep learning. AMIA Summits on Translational Science Proceedings 2018; 2018: 147.

[18] Bidari I, Chickerur S, Ranmale H, Talawar S, Ramadurg H et al. Hyperspectral imagery classification using deep learning. In: IEEE 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4); 2020. pp. 672-676.

[19] Khaled O, El-Sahhar M, El-Dine MA, Talaat Y, Hassan YM et al. Cascaded architecture for classifying the preliminary stages of diabetic retinopathy. In: Proceedings of the 2020 9th International Conference on Software and Information Engineering (ICSIE); 2020. pp. 108-112.

[20] Khan RU, Zhang X, Kumar R, Tariq HA. Analysis of resnet model for malicious code detection. In: IEEE 2017 14th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP); 2017. pp. 239-242.

[21] Gulshan V, Peng L, Coram M, Stumpe MC, Wu D et al. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. Jama 2016; 316 (22): 2402-2410.

[22] Zhang D, Bu W, Wu X. Diabetic retinopathy classification using deeply supervised ResNet. In: 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI); 2017. pp. 1-6.

[23] Gardner GG, Keating D, Williamson TH, Elliott AT. Automatic detection of diabetic retinopathy using an artificial neural network: a screening tool. British Journal of Ophthalmology 1996; 80 (11): 940-944.

[24] Weiss K, Khoshgoftaar TM, Wang D. A survey of transfer learning. Journal of Big Data 2016; 3 (1): 1-40.

[25] Zhuang F, Qi Z, Duan K, Xi D, Zhu Y et al. A comprehensive survey on transfer learning. Proceedings of the IEEE. 2020; 109 (1): 43-76.

[26] Islam SM, Hasan MM, Abdullah S. Deep learning based early detection and grading of diabetic retinopathy using retinal fundus images. arXiv preprint arXiv:1812.10595. 2018.

[27] Shankar K, Sait AR, Gupta D, Lakshmanaprabu SK, Khanna A et al. Automated detection and classification of fundus diabetic retinopathy images using synergic deep learning model. Pattern Recognition Letters 2020; 133: 210-216.

[28] Qummar S, Khan FG, Shah S, Khan A, Shamshirband S et al. A deep learning ensemble approach for diabetic retinopathy detection. IEEE Access 2019; 7: 150530-150539.

[29] Hacisoftaoglu RE, Karakaya M, Sallam AB. Deep learning frameworks for diabetic retinopathy detection with smartphone-based retinal imaging systems. Pattern Recognition Letters 2020; 135: 409-417.

[30] Xia Y, Liu C, Li Y, Liu N. A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. Expert Systems with Applications 2017; 78: 225-241.

[31] Gonzalez-Cuautle D, Corral-Salinas UY, Sanchez-Perez G, Perez-Meana H, Toscano-Medina K et al. An efficient botnet detection methodology using hyper-parameter optimization trough grid-search techniques. In: IEEE 2019 7th International Workshop on Biometrics and Forensics (IWBF); 2019. pp. 1-6.

[32] Dewancker I, McCourt M, Clark S. Bayesian optimization for machine learning: A practical guidebook. arXiv preprint arXiv:1612.04858. 2016.

[33] Kulkarni U, Meena SM, Gurlahosur SV, Mudengudi U. Classification of cultural heritage sites using transfer learning. In: 2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM); 2019. pp. 391-397.

[34] Yogatama D, Mann G. Efficient transfer learning method for automatic hyperparameter tuning. In: PMLR Artificial Intelligence and Statistics; 2014. pp. 1077-1085.

[35] Law HC, Zhao P, Chan L, Huang J, Sejdinovic D. Hyperparameter learning via distributional transfer. arXiv preprint arXiv:1810.06305 2018.

[36] Perrone V, Jenatton R, Seeger M, Archambeau C. Scalable hyperparameter transfer learning. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems; 2018. pp. 6846-6856.

[37] Perrone V, Shen H, Seeger M, Archambeau C, Jenatton R. Learning search spaces for bayesian optimization: Another view of hyperparameter transfer learning. arXiv preprint arXiv:1909.12552 2019.

[38] Zela A, Klein A, Falkner S, Hutter F. Towards automated deep learning: Efficient joint neural architecture and hyperparameter search. arXiv preprint arXiv:1807.06906 2018.

[39] Ozaki Y, Yano M, Onishi M. Effective hyperparameter optimization using Nelder-Mead method in deep learning. IPSJ Transactions on Computer Vision and Applications 2017; 9 (1): 1-12.

[40] Graham B. Kaggle diabetic retinopathy detection competition report. University of Warwick, 2015.

[41] He K, Zhang X, Ren S, Sun J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision; 2015. pp. 1026-1034.

[42] Paszke A, Gross S, Massa F, Lerer A, Bradbury J et al. Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems 2019; 32: 8026-8037.

[43] Wilson AC, Roelofs R, Stern M, Srebro N, Recht B. The marginal value of adaptive gradient methods in machine learning. arXiv preprint arXiv:1705.08292 2017.

[44] Horváth S, Klein A, Richtárik P, Archambeau C. Hyperparameter Transfer Learning with Adaptive Complexity. In: PMLR International Conference on Artificial Intelligence and Statistics; 2021. pp. 1378-1386.