Local Binary CNN for Diabetic Retinopathy Classification on Fundus Images

Peter Macsik, Jarmila Pavlovicova, Jozef Goga, Slavomir Kajan

Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava, Ilkovicova 3, 812 19 Bratislava, Slovakia
e-mail: peter.macsik@stuba.sk; jarmila.pavlovicova@stuba.sk; jozef.goga@stuba.sk; slavomir.kajan@stuba.sk

Abstract: Diabetic retinopathy (DR), is currently one of the major causes of preventable blindness, worldwide. With an early diagnosis and proper treatment of this eye disease, we can prevent the spread of diabetic retinopathy. In this paper, we propose a new alternative of local binary convolutional neural network (LBCNN) deterministic filter generation which can approximate the performance of the standard convolutional neural network (CNN) with less learnable parameters and also with less memory use, which can be helpful in systems with low-memory or low computational capacity, like smart-phones. We compare our scheme with standard CNN and LBCNN that uses stochastic filter generation strategy on retinal fundus image datasets in case of binary classification into healthy and damaged classes. These experiments are also evaluated according to the standard criteria used in medical applications, such as, overall accuracy, specificity, sensitivity and predictive values. On the small dataset (Aptos), one of our proposed LBCNN architectures outperformed all of the other deep learning models examined.

Keywords: CAD (Computer-aided diagnostics); Binary classification; Memory reduction; Learnable parameters

1 Introduction

Nowadays diabetes mellitus (DM) is a global disease [1] and the number of patients will probably increase in the future [2] [3]. On the other hand, diabetic retinopathy is the specific microvascular complication of DM and every third diabetic is affected by DR [1] and unfortunately is at risk of developing DR.

DR is an eye disease that can cause irreversible eye damages (i.e. blurred vision, black shapes, or dots in the vision area), in the worst cases even blindness, however, it could be preventable for in time diagnostic and proper treatment [4]. Diabetic retinopathy is classified according to symptoms and severity into two main groups, non-proliferative diabetic retinopathy (NPDR) and proliferative diabetic retinopathy (PDR). Subsequently, these stages are divided in more detail by individual symptoms according to the International Classification of Diabetic Retinopathy (ICDR) [1] scale. The international scale (ICDR) divides DR into five classes according to the severity of the disease. The definition of the ICDR scale is described in Table 1. Examples of fundus images from each ICDR class are shown in Fig. 1.

Table 1
Description of DR stages, ICDR scale [1]

Disease Severity Level	Findings Observable upon Dilated Ophthalmoscopy
No DR	No abnormalities
Mild NPDR	Microaneurysms only
Moderate NPDR	Microaneurysms and other signs, but less than severe NPDR
Severe NPDR	Moderate NPDR with any of the following: • Intraretinal hemorrhages (≥ 20 in each quadrant) • Definite venous beading (in 2 quadrants) • Intraretinal microvascular abnormalities (in 1 quadrant) • and no signs of proliferative retinopathy
Proliferative DR	Severe NPDR and 1 or more of the following: • Neovascularization • Vitreous/preretinal hemorrhage

Increased prevalence of DM leads to increased query for DR screening. Due to an increased number of patients with DM (respectively with DR) and an insufficient number of clinicians, there is more pressure on healthcare systems to find acceptable automatized DR screening methods with minimized costs.

In this paper, we propose the binary classification (especially classification into the healthy and damaged groups) with a memory-efficient CNN alternative which is called LBCNN and compare its performance with the standard CNN network. Memory-efficient CNN network alternatives are important for devices that are not equipped with sufficient memory. In our case, it could be for instance smart-phone which is one of the possible solutions how to avoid high costs and keep the comfort for the patient, but also enable regular DR screening. We tested our network on two fundus image databases (EyePACS [5] and Aptos [6]) of different sizes and showed computational efficiency of our solution in case of limited amount of training data.

The rest of this paper contains an overview of the related work in Section 2 and used datasets and image augmentations in Section 3. In Section 4 we introduce used methods for classification like standard CNN (ResNet18), LBCNN with stochastic filter generation mode and with proposed LBCNN with deterministic filter generation mode. In Section 5 there is the description of experiments and finally, we present our conclusions.

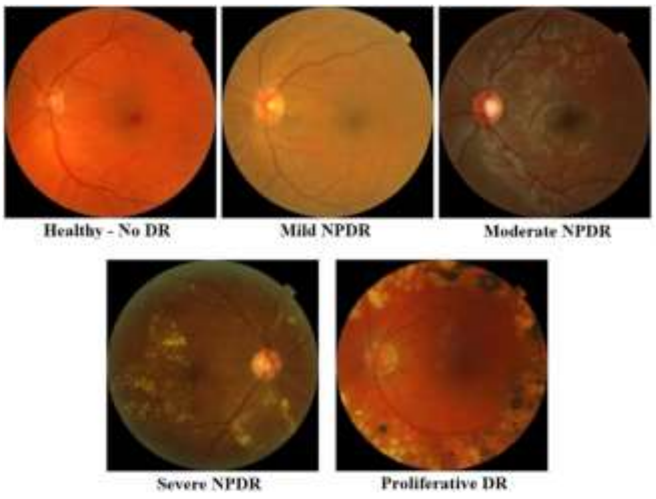


Figure 1
Examples of fundus images of different stages of DR

2 Related Works

The first attempts to automatically classify diabetic retinopathy were reported in the 1990s when Gardner and his colleagues described the usage of an artificial neural network, which was able to detect diabetic retinopathy with 88% sensitivity and 83% specificity compared to an ophthalmologist [7]. Certainly, since the first attempt, there were created many new applications for the classification of DR with different algorithms i.e. random forest classifier, support vector machine, or regression tree classifier reviewed in [8].

Nowadays, deep learning (DL) algorithms are the cost-effective solutions that could help to solve this problem. DL is a subarea in artificial intelligence (AI). On the other hand, CNN models belong to DL algorithms, that can be used among many others for image classification with repetitive analysis and compare the output with a standard (such as a human grader) and make self-correction in case of error. Several studies have confirmed successful results in the development of DL algorithms that have been able to identify DR without any need to have some specific properties of DR in advance. There are several ways to classify DR, e.g. by two (healthy, damaged), three (healthy, NPDR, PDR) or five (ICDR) classes [1].

For example, Islam et al. [9] developed two binary classification models. First, for detecting the presence of the disease (healthy vs damaged) and the second one for grading its severity (grades 0, 1 vs 2, 3, 4). In another study by Hagos et al. [10], the authors achieved 90.9% accuracy in binary classification with Inception-V3 [11] pre-trained and fine-tuned with a reduced dataset had 2500 fundus images.

Bodapati et al. [12] proposed a solution for binary classification (healthy versus diseased) and for the classification of the severity of DR (5 class - ICDR). They used different CNN architectures as feature extractor that were fused with deep neural network. Li et al. [13] proposed DR severity classification and an additional class (6 in total) to classify ungradable images as well. They trained many CNN architectures like VGG-16 [14], DenseNet-121 [15], GoogLeNet [16], ResNet-18 [17] where the best results in accuracy were achieved by ResNet-18 architecture.

On the other hand, one of the most challenging problems in designing robust DL methods, especially based on CNN models with deeper architectures, is the acquisition of huge volumes of labelled fundus images on pixel-level and with image-level annotations. The main issue is not the availability of huge datasets, but the annotation of these images, which is expensive and requires the services of expert ophthalmologists [18]. The solution could be a deep model, which is able to learn from limited data, and this is also an important area of research not only for the diagnosis of DR, but generally for medical image analysis, as well. To deal with this problem, we introduce a modified CNN model that has comparable performance with standard CNN but involves reduced number of learnable parameters.

Our research was inspired by the work of Juefei-Xu et al. [19]. They developed an efficient alternative to convolutional layers in standard CNN. This layer is called the local binary convolution (LBC) layer, which was motivated by local binary patterns (LBP) [20], a very efficient visual descriptor used for classification in computer vision. They called CNN with LBC layers LBCNN. In experiments, the LBCNN network was used for the classification task on ImageNet database [21]. The LBC layer comprises of fixed sparse pre-defined binary convolutional filters, which are fixed during the training process, a non-linear activation function and a set of learnable weights. The weights combine the activated filter responses to approximate the corresponding activated filter responses of a standard convolutional layer. The LBC layer affords significant savings, 9× to 169× in the number of learnable parameters compared to a standard convolutional layer (more details in Section 4.2). These parameter savings reduce memory and disk space requirements, which is beneficial for devices with lower computational power, e.g. smart phones [19]. Besides, smart phone DR screening is also a popular research field [22-24]. For example, Rajalakshmi et al. [24] assessed the role of AI based automated software for the detection of DR and sight-threatening DR fundus photography taken by a smartphone-based device and validated it against ophthalmologists grading.

For this reason, we applied the original LBCNN for DR classification and also we introduce an extension for the deterministic LBC layer with added Prewitt filters [25]. Thus, the original LBP filter base was extended for edge detection leading to improvement in feature extraction. The expected methodological scientific contributions of the paper were as follows:

- Experimental proof that original LBCNN with stochastic filters is usable also for binary DR classification and can achieve comparable results to standard CNN. For comparison, we use ResNet18 architecture as the best performing architecture in the study by Li et al. [13].
- Additional memory saving by application of a deterministic fixed filter base in LBCNN instead of stochastic filters while achieving comparable results to baseline LBCNN with stochastic filters. In this case it is not necessary to save the filter base for further reuse.

3 Datasets

We propose image classification of fundus images obtained by a fundus camera. Fundus images show the interior surface of the eye, opposite to the lens. In our work, we chose EyePACS [5] and Aptos [6] from freely available fundus eye databases. These two databases differ in size markedly. Difference in size allows us to demonstrate the benefits and the drawbacks of proposed methods compared to standard CNN.

3.1 EyePACS

EyePACS database [5] contains color fundus images which were divided by ophthalmologists into five classes (ICDR) according to the grade of DR retinal damage. EyePACS provided this database in 2015, for Kaggle [26] competitors. The aim of this competition was to design the best possible automated detection system of DR symptoms [1]. Original database contains 35126 training images with different image resolutions with following grade distribution: No DR 25810 (grade 0), Mild NPDR 2 443 (grade 1), Moderate NPDR 5 292 (grade 2), Severe NPDR 873 (grade 3), Proliferative DR 708 (grade 4). Due to few images in classes 3 and 4 which indicate unbalance between classes, we augmented (similarly like in [27]) this part of the dataset by adding images from the EyePACS testing dataset [5]. In case of class 3 it was 2087 images and in case of class 4 it was 1914 images. Since the dataset contains also left and right eyes, we used mirroring to double the number of images. After this augmentation, we observed 25790 images of healthy and 23472 images with DR symptoms.

3.2 Aptos

Asia Pacific Tele-Ophthalmology Society 2019 Blindness Detection Dataset [6] was divided as well as EyePACS dataset into five classes. Public Aptos database contains 3662 images with various resolutions (up to 3216×2136) with following DR grade distribution: No DR 1805 (grade 0), Mild NPDR 370 (grade 1), Moderate NPDR 999 (grade 2), Severe NPDR 193 (grade 3), Proliferative DR 295 (grade 4).

In the context of our binary classification, after merging retinal images with DR, we obtained 1805 images of healthy retina and 1857 of damaged retinal images with signs of DR.

3.3 Fundus Image Preprocessing

In both databases, it is possible to observe black borders around the eye fundus. However, from the point of view of CNN network training, these black borders do not contain any important information, so in order to reduce the size of input images and at the same time reduce computational complexity, it is appropriate to trim them. For this reason, before the training process we cropped black borders automatically with an adaptive method, similarly as in the study by Shao et al. [28]. After border cropping, images were resized to 300×300 pixels to reach uniform image resolution. An example of the cropped and resized image is shown in Fig. 2.

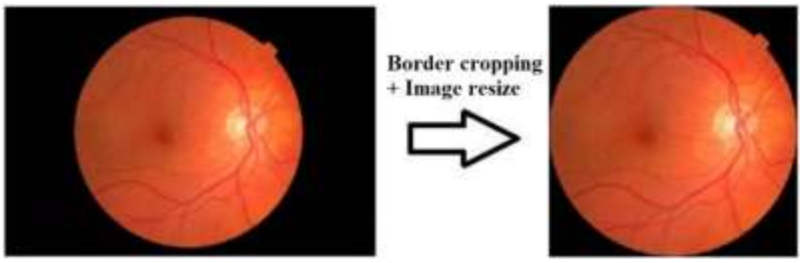


Figure 2
Fundus image preprocessing example

4 Methods

As a classification algorithm baseline model, we used well known CNN network and its new alternative LBCNN [19] in frame of architecture ResNet18 (Fig. 3). LBCNN was born from the idea of combination LBP descriptor and CNN architecture. The main advantage of LBCNN is the potential to achieve comparable results with CNN architecture with the benefits of less learnable parameters and lower memory requirements.

4.1 CNN

Convolutional neural networks are widely used deep learning models, which achieve high popularity for image classification tasks. They are mostly based on computational layers like convolutional or pooling layer and activation functions like ReLU or sigmoid. These networks have randomly initialized convolutional filters which are optimized during the training for feature extraction. One of the first

CNN was developed by Yann LeCun et al. which was called LeNet [29]. During the years many architectures of CNNs were published, i.e. AlexNet [30], ResNet [17], VGG [14], etc.

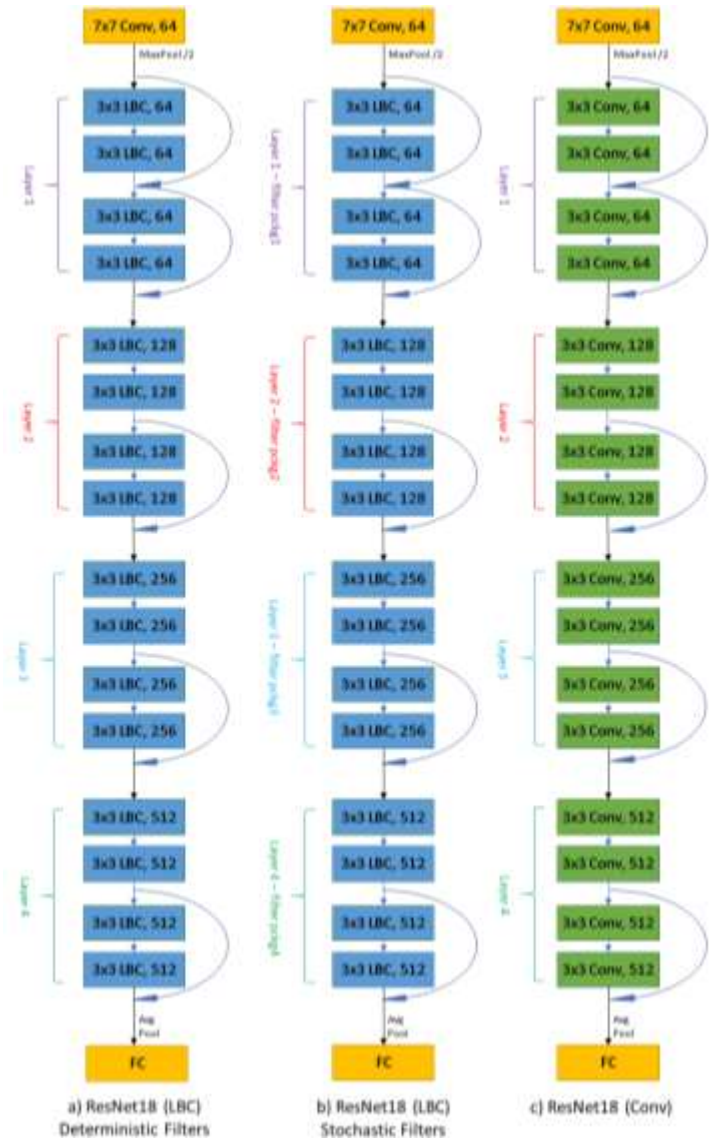


Figure 3
Visualization of used ResNet18 models: a) LBC layered with deterministic filters; b) LBC layered with stochastic filters; c) with standard convolutional layers

These architectures have different numbers and types of layers with different connections. In most cases, these architectures have a huge number of learnable parameters and certainly, this could lead to increased memory requirements. In this paper we compared one alternative of CNN with reduced number of learnable parameters. This network is called LBCNN [19]. As an experimental CNN architecture, for results comparison, we chose ResNet18 [17] implemented in PyTorch framework [31] (model c) in Fig. 3).

4.2 Local Binary CNN - LBCNN

LBCNN [19] is an alternative to the standard CNN which can approximate the performance of CNN with less learnable parameters. It was born from the idea of LBP convolution which has 8 special binary non-learnable filters, activation function, and binary weights for a linear combination. These factors were generalized to the m binary fixed filters (they are not learnable). The linear combination part of the layer was generalized from binary numbers to the real values. This linear combination with real values was applied as pointwise convolution (convolution with 1×1 sized filters) and this is the only part where this layer can learn [20]. Such layer is called the LBC layer and CNN with these LBC layers is called LBCNN. LBC layer function can be expressed by the following equation:

$$x_{l+1}^{t} = \sum_{i=1}^{m} \sigma\left(\sum_{s} b_i^s * x_l^s\right) \cdot V_{l,i}^{t} \tag{1}$$

where t and s represent the number of input and output channels, m is the number of fixed filters (b_i, $i \in [m]$), x_l is the input from l th layer and x_{l+1} is the output from layer, and consequently it is the input into layer $l+1$. $V_{l,i}$ are weights in pointwise convolution. The activation function is σ (we used ReLU). Operator * stands for standard 2D convolution and operator · denotes pointwise convolution. Visualization of a single LBC layer is shown in Fig. 4.

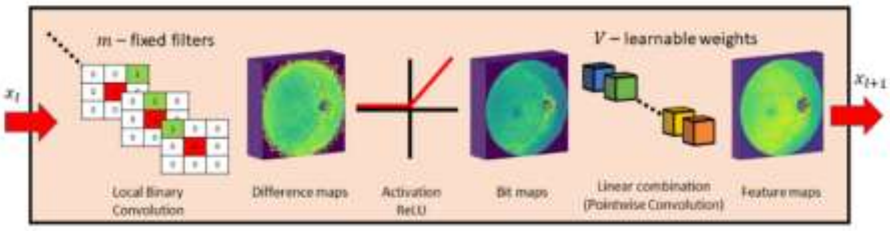


Figure 4
Single LBC layer (m - number of fixed filters, V - weights for linear combination)

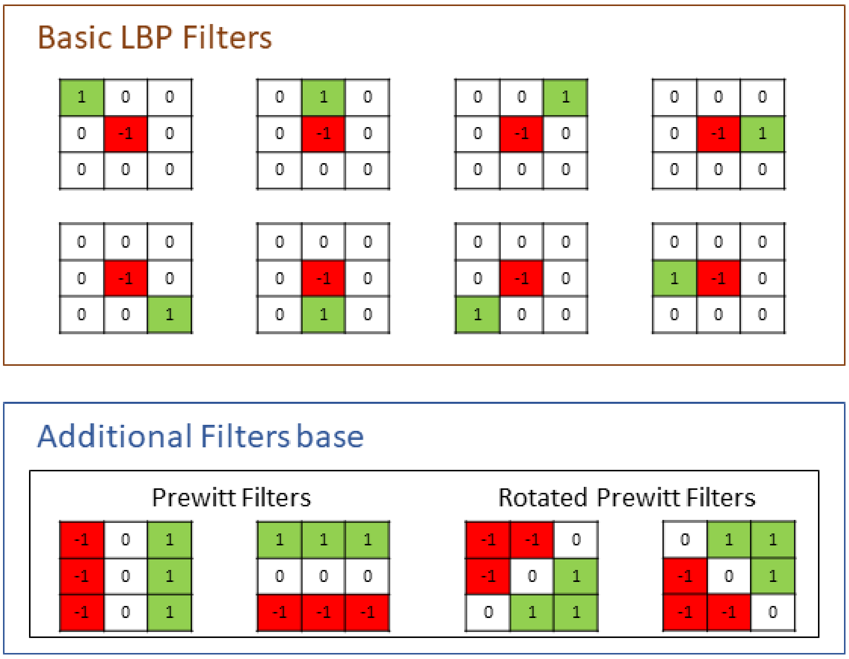
LBCNN saves parameters through binary non-learnable filters which can be generated in two ways, one is deterministic and the second one is stochastic. In this paper, we used a deterministic and also stochastic filter generation strategy.

4.2.1 Stochastic LBC Filters

We used stochastic fixed filter generation described in [19]. Memory savings were achieved here by the ability to share fixed filters across layers with the same dimensions. We generated a new package of fixed filters for every layer (model b) in Fig. 3) and we shared them between LBC layers. First, filter generation sparsity must be defined, which represents the ratio between zero values and non-zero values in the filter. If the value is non-zero it has value 1 or -1, according to the Bernoulli distribution. We used stochastic filters with a sparsity of 0.5, that was determined in original paper [19] as a good standard value.

4.2.2 Deterministic LBC Filters

Deterministic filter generation strategy can save extra memory compared to stochastic filter generation. In case of deterministic filters, it is not necessary to save fixed filters after training because we know how they look like. In case of stochastic generation, it is necessary to save all fixed filters for further model re-use due to random factor. In this paper, we used original LBP filters with some additional deterministic fixed filters in order to increase the filter base. Additional filters are shown in Fig. 5.



$$\begin{bmatrix}1&0&0\\0&-1&0\\0&0&0\end{bmatrix}\quad\begin{bmatrix}0&1&0\\0&-1&0\\0&0&0\end{bmatrix}\quad\begin{bmatrix}0&0&1\\0&-1&0\\0&0&0\end{bmatrix}\quad\begin{bmatrix}0&0&0\\0&-1&1\\0&0&0\end{bmatrix}$$

$$\begin{bmatrix}0&0&0\\0&-1&0\\0&0&1\end{bmatrix}\quad\begin{bmatrix}0&0&0\\0&-1&0\\0&1&0\end{bmatrix}\quad\begin{bmatrix}0&0&0\\0&-1&0\\1&0&0\end{bmatrix}\quad\begin{bmatrix}0&0&0\\1&-1&0\\0&0&0\end{bmatrix}$$

$$\begin{bmatrix}-1&0&1\\-1&0&1\\-1&0&1\end{bmatrix}\quad\begin{bmatrix}1&1&1\\0&0&0\\-1&-1&-1\end{bmatrix}\quad\begin{bmatrix}-1&-1&0\\-1&0&1\\0&1&1\end{bmatrix}\quad\begin{bmatrix}0&1&1\\-1&0&1\\-1&-1&0\end{bmatrix}$$

Figure 5

Base filters used in LBC layers. Basic 8 LBP filters extended by another 4 filters, Prewitt and rotated Prewitt filters

This additional base contains Prewitt filters [25] and rotated Prewitt filters for edge recognition which can be important near vessels and borders of the eye and also for better detection of circle-shaped disease signs i.e. microaneurysms. We used LBCNN with deterministic filters with 2 setups. In the first, we used 12 filters, as in Fig. 5. In the second, we doubled the number of fixed filters to 24, by keeping the original 12 and the expansion was done by swapping values -1 and 1 in every filter.

4.2.3 Number of Learnable Parameters

The major advantage of LBC layers application is the reduction of learnable parameters number by the preservation of similar learning ability. If we assume that convolutional filters do not have bias terms, comparison between learnable parameters in each LBC and Conv layer can be expressed with the following equation:

$$\frac{CNN\ params.}{LBCNN\ params.} = \frac{p \times h \times w \times q}{m \times q} \tag{2}$$

where m in our case was 12, 24, and 72 which is the number of fixed filters in LBC layers. Next parameters h and w stand for the height and width of fixed LBC filters (both are 3 in our case), respectively. Parameters p and q stand for the number of input and output channels. The accurate parameter difference for the models is shown in Table 2. This table contains model name and number of learnable parameters (Params).

Table 2
Learnable parameters. [number]f - number of filters, sto. - stochastic, det. - deterministic)

Model	**Params [million]**		
Standard CNN – ResNet18	11.178		
	12f	24f	72f
LBCNN – ResNet18 sto.	–	0.288	0.472
LBCNN – ResNet18 det.	0.242	0.288	–

4.2.4 Memory Size Difference

Deterministic filter generation strategy has the advantage in memory saving compared to the stochastic generation as we described above, as there is not necessary to store them. It means that alternatively, we could generate them programmatically in a predefined order. It is sufficient to save only learnable parts of the model, which are pointwise convolutional layers. However, in case of stochastic filter generation, it is important to save fixed filters, otherwise, further model re-use is impossible. To compare the memory requirement of these models in the PyTorch framework [15] we saved networks in the *.ckp file format, where we can save trained parameters and fixed filters for further re-use. This comparison is shown in Table 3. It contains the memory size requirements for standard CNN ResNet18 and each setup of LBCNN.

Note: The specific memory size may vary from different hardware and software factors.

Table 3
Memory size difference. ([number]f-number of filters, sto. - stochastic, det. - deterministic)

Model	**Size [KB]**		
Standard CNN – ResNet18	131106		
	12f	24f	72f
LBCNN – ResNet18 sto.	–	4325	8105
LBCNN – ResNet18 det.	2972	3512	–

5 Experiments and Results

In this chapter, we present our experiments of two approaches with proposed architectures. In the first case we did experiments on basic single model classification which is favorable in case of low memory and computational capacity. In the second approach, we experiment with an ensemble of more models that can offer improvements in classification accuracy with minimal increase in the number of parameters and memory requirements.

5.1 Single Model Classification

Firstly, we have made hyperparameter tuning with grid search method and empirically discovered the best training setup for selected models. We have tested different weight optimization methods (such as Adagrad, Adadelta, Adam, AdamW, and Nadam [32-35]) with different hyperparameters. We achieved the best results with Nadam optimizer, with learning rate 0.001 and with fixed number of epochs, 30 and 40 for EyePACS and Aptos, respectively. Other parameters were kept default as in PyTorch implementation of Nadam, which is on GitHub repository [36]. This hyperparameter tuning was made on CNN (ResNet18), and for objective comparison of models performance, we kept this setup for LBCNN models too. We made experiments on all the above-mentioned datasets, where we divided datasets into 80-20% ratio for the training and testing with random data selection in all cases. For every model, we made 30 repeated runs with the setups described above. After 30 runs we chose 10 best models based on the test accuracy, and evaluated the obtained results.

Results of our experiments for the smaller dataset (Aptos) are shown in Table 4, and for the bigger dataset (EyePACS) in Table 5. Tables contain evaluation metrics used in medicine like accuracy, sensitivity (sens), specificity (spec), negative predictive value (NPV), and positive predictive value (PPV). These metrics were calculated as an average of 10 best models. The above-described metrics for a single model were calculated from confusion matrices through the use of TP, FP, TN, FN values (F-False, T-True, P-Positive, N-Negative).

Specifically, in medical classification tasks, confusion matrix values can be described as follows:

- True Positive (TP): Damaged image correctly identified as damaged
- False Positive (FP): Healthy image incorrectly identified as damaged
- True Negative (TN): Healthy image correctly identified as healthy
- False Negative (FN): Damaged image incorrectly identified as healthy

These metrics are expressed by the following equations:

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{3}$$

$$spec = \frac{TN}{TN+FP} * 100 \tag{4}$$

$$sens = \frac{TP}{TP+FN} * 100 \tag{5}$$

$$NPV = \frac{TN}{TN+FN} * 100 \tag{6}$$

$$PPV = \frac{TP}{TP+FP} * 100 \tag{7}$$

To express the relation between specificity and sensitivity we also used an evaluation metric called AUC (Area under receiver operating characteristic curve) [37]. AUC is included in Tables 4 and 5 also as median accuracy and standard deviation (std) of the 10 best models. Tables also contains best and median accuracy. Figs. 6-7 show boxplot visualization of the 10 best models performance.

Table 4
Results of 10 best experiments on Aptos dataset (det. - deterministic, sto. - stochastic, f – filters, acc. - accuracy)

	Models				
	CNN ResNet18 [17]	**LBCNN (sto. 24f) [19]**	**LBCNN (sto. 72f) [19]**	**LBCNN (det. 12f) [ours]**	**LBCNN (det. 24f) [ours]**
mean acc. [%]	95.59	95.57	95.44	95.89	**96.58**
std	0.556	0.4616	**0.362**	0.3832	0.4022
median acc. [%]	95.29	95.36	95.36	95.91	**96.52**
best acc. [%]	96.73	96.32	96.04	96.45	**97.41**
auc	0.979	0.9803	0.979	0.9832	**0.9871**
spec [%]	95.94	96.01	95.74	95.96	**96.59**
sens [%]	93.77	93.67	93.22	94.03	**94.63**
NPV [%]	93.95	93.56	93.39	94.05	**94.35**
PPV [%]	95.71	95.83	95.75	96.08	**96.73**

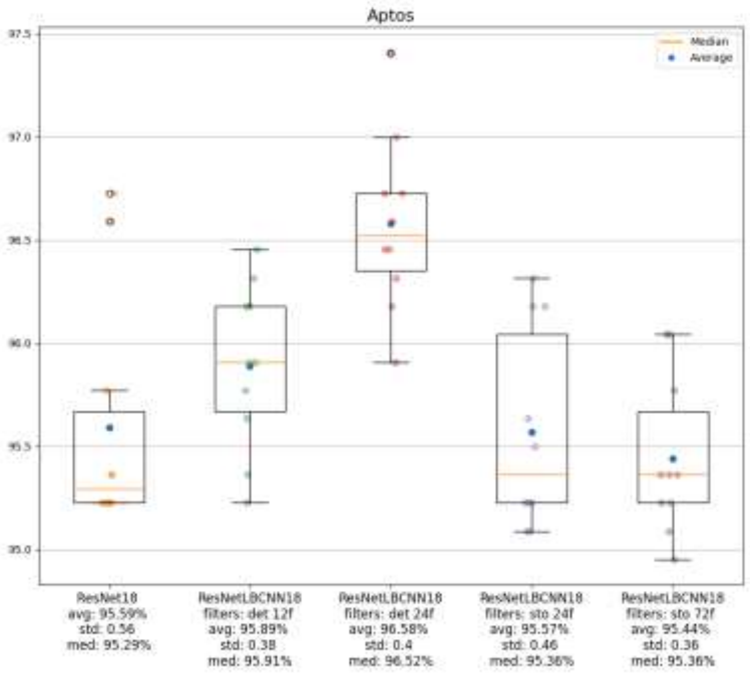


Figure 6

Boxplot visualization of reached results on Aptos for used models (best 10 experiments)

Table 5

Results of 10 best experiments on EyePACS dataset (det. - deterministic, sto. - stochastic, f - filters, acc. - accuracy)

	Models				
	CNN ResNet18 [17]	**LBCNN (sto. 24f) [19]**	**LBCNN (sto. 72f) [19]**	**LBCNN (det. 12f) [ours]**	**LBCNN (det. 24f) [ours]**
mean acc. [%]	**91.12**	90.22	90.4	88.73	89.71
std	**0.1828**	0.2484	0.2519	0.3696	0.2475
median acc. [%]	**91.11**	90.14	90.33	88.66	89.65
best acc. [%]	**91.44**	90.73	91.03	89.34	90.31
auc	**0.9725**	0.9698	0.9706	0.9605	0.9661
spec [%]	93.32	93.06	**93.41**	92.23	93.08
sens [%]	**87.82**	86.27	86.72	84.56	85.78
NPV [%]	**89.34**	88.14	88.5	86.68	87.65
PPV [%]	92.22	92.05	**92.28**	90.79	91.91

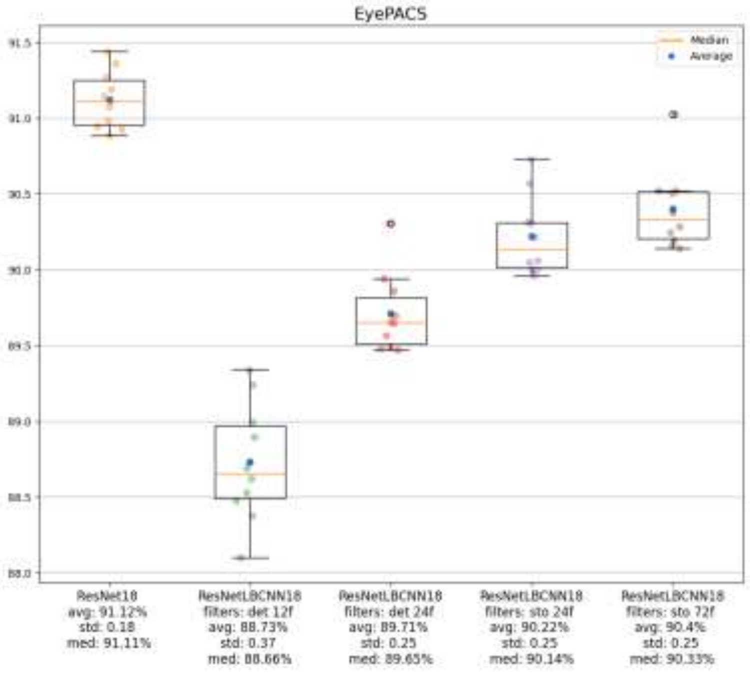


Figure 7

Boxplot visualization of reached results on EyePACS for used models (best 10 experiments)

5.2 Ensemble Classification

A different, interesting practical approach, could be the ensemble creation of standard CNN and LBCNNs, however, this ensemble model [38] will require a larger model size, over the single standard CNN, but on the other hand, we could achieve classification improvement, using minimal parameters and minimal model size increases. As a demonstration, we used ensemble of 3 models (LBCNN deterministic with 24 filters, LBCNN stochastic with 24 filters and standard ResNet18) (Fig. 8). We used the Model Averaging Ensemble, to combine particular predictions, which means, every single model has an equal impact (weight), on the final prediction.

We made these experiments on EyePACS database where ensemble of 3 models with equal weights in ensemble produced the following results: mean and median of the best 10 experiments were 92.00% and 91.97%, respectively. The best result we achieved was 92.39%. It means +0.88%, +0.86%, +0.95% improvement of accuracy on average, median, and on the best model with adding just 7 837 KB of memory or in learnable parameters, it means +0.488 million additional learnable parameters.

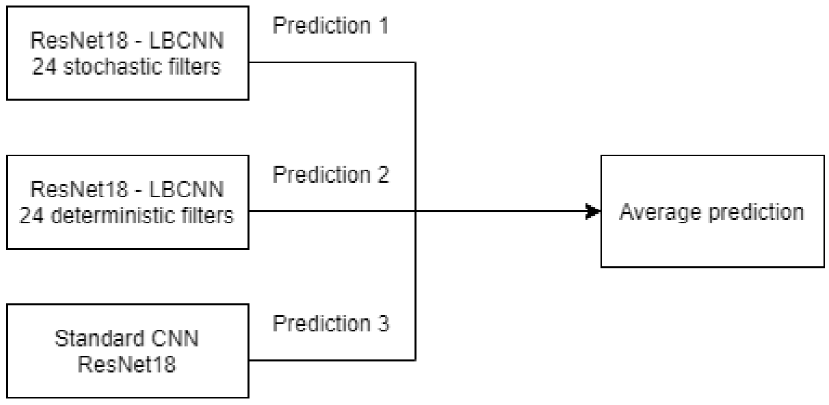


Figure 8
Visualization of Averaging Ensemble Model

Conclusions

In our experiments, we proposed the additional deterministic filters application, in LBCNN, to achieve a more accurate DR image classification, for healthy or damaged classes. Specifically, this means that we extended a filter base of 8 LBP filters by 4 Prewitt filters and then we doubled the number of filters by swapping non-zero values in each filter. Thus, we effectively created 24 fixed filters. This deterministic filter generation can decrease the parameters memory requirement, compared to stochastic filters, because there is no need for fixed filter storing for further trained model re-use. This approach can be useful for low-memory devices, such as, smart-phones. It can be a very cost-effective solution of regular DR screening. However, in general, the selection of the deterministic filters base can also be a weakness, for example, in the case that the selected filters are not optimized for the given classification task, the performance of this LBCNN can be even worse, compared to the standard CNN.

Based on our experiments on fundus image datasets, we can establish that LBCNN, with both strategies of filter generation (stochastic and deterministic), can approximate the performance of a standard CNN network for binary DR classification, moreover, it saves a significant amount of learnable parameters and decreases memory requirements. Specifically, in experiments on the smaller dataset (Aptos), performance of the LBCNN, with 24 deterministic filters, gave the best results, except for the standard deviation, where LBCNN with 72 stochastic filters achieved the best results, however, the difference was negligibly low. In case of the larger dataset (EyePACS), performance of the LBCNN with 24 deterministic filters, was slightly worse, because of an insufficient number of parameters, but it can be said that there is still a good trade-off between performance and the memory requirements. In this case, the standard CNN achieved the best results. This indicates that using LBCNN with deterministic filters is fully applicable for classification tasks, where only small datasets are available.

LBCNN with deterministically or stochastically generated filters are also usable separately and also as a part of an ensemble model, as a mechanism of improvement. Certainly, this option requires more memory compared to the single CNN, but if memory allows for it, it can be an alternative to improve classification accuracy with a minimal memory increase. This improvement was also demonstrated in our experiments, where we combined 3 models (LBCNN with 24 stochastic filters, LBCNN with 24 deterministic filters and a standard CNN-ResNet18). Conversely, using the dataset EyePACS, we achieved almost +1% improvement in accuracy, compared to the best classifier of the group.

As future work, we plan to experiment with a greater ensemble of models, with the purpose to find optimal weights for classification performance, with minimum memory requirement target.

References

[1] International Council of Opthalmology, ICO Guidelines for Diabetic Eye Care, (2017) 40

[2] D. R. Owens, J. Dolben, S. Young, R. E. J. Ryder, I. R. Jones, J. Vora, D. Jones, D. Morsman, T. M. Hayes, Screening for Diabetic Retinopathy, Diabetic Medicine. 8 (1991) S4–S10. https://doi.org/10.1111/j.1464-5491.1991.tb02148.x

[3] D. C. Klonoff, The Increasing Incidence of Diabetes in the 21st Century, Journal of Diabetes Science and Technology. 3 (2009) 1-2. https://doi.org/10.1177/193229680900300101

[4] P. Vashist, S. Singh, N. Gupta, R. Saxena, Role of Early Screening for Diabetic Retinopathy in Patients with Diabetes Mellitus: An Overview, Indian Journal of Community Medicine : Official Publication of Indian Association of Preventive & Social Medicine. 36 (2011) 247-252. https://doi.org/10.4103/0970-0218.91324

[5] EyePACS, Diabetic Retinopathy Detection - Kaggle, 2015. https://www.kaggle.com/c/diabetic-retinopathy-detection (accessed January 27, 2021)

[6] APTOS, APTOS 2019 blindness detection, 2019. https://www.kaggle.com/c/aptos2019-blindness-detection

[7] G. G. Gardner, D. Keating, T. H. Williamson, A. T. Elliott, Automatic detection of diabetic retinopathy using an artificial neural network: a screening tool., British Journal of Ophthalmology. 80 (1996) 940–944. https://doi.org/10.1136/bjo.80.11.940

[8] S. Vadloori, Y.-P. Huang, W.-C. Wu, Comparison of Various Data Mining Classification Techniques in the Diagnosis of Diabetic Retinopathy, Acta Polytechnica Hungarica. 16 (2019) 27-46

[9] S. M. S. Islam, M. M. Hasan, S. Abdullah, Deep Learning based Early Detection and Grading of Diabetic Retinopathy Using Retinal Fundus Images, ArXiv:1812.10595 [Cs] (2018) http://arxiv.org/abs/1812.10595 (accessed December 27, 2021)

[10] M. T. Hagos, S. Kant, Transfer Learning based Detection of Diabetic Retinopathy from Small Dataset, ArXiv:1905.07203 [Cs] (2019) http://arxiv.org/abs/1905.07203 (accessed December 8, 2021)

[11] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the Inception Architecture for Computer Vision, ArXiv:1512.00567 [Cs] (2015) http://arxiv.org/abs/1512.00567 (accessed February 12, 2022)

[12] J. D. Bodapati, V. Naralasetti, S. N. Shareef, S. Hakak, M. Bilal, P. K. R. Maddikunta, O. Jo, Blended Multi-Modal Deep ConvNet Features for Diabetic Retinopathy Severity Prediction, Electronics. 9 (2020) 914. https://doi.org/10.3390/electronics9060914

[13] T. Li, Y. Gao, K. Wang, S. Guo, H. Liu, H. Kang, Diagnostic assessment of deep learning algorithms for diabetic retinopathy screening, Information Sciences. 501 (2019) 511-522, https://doi.org/10.1016/j.ins.2019.06.011

[14] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, ArXiv:1409.1556 [Cs] (2015) http://arxiv.org/abs/1409.1556 (accessed February 4, 2021)

[15] G. Huang, Z. Liu, L. van der Maaten, K. Q. Weinberger, Densely Connected Convolutional Networks, ArXiv:1608.06993 [Cs] (2018) http://arxiv.org/abs/1608.06993 (accessed December 5, 2021)

[16] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going Deeper with Convolutions, ArXiv:1409.4842 [Cs] (2014) http://arxiv.org/abs/1409.4842 (accessed November 29, 2021)

[17] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016. https://doi.org/10.1109/CVPR.2016.90

[18] N. Asiri, M. Hussain, F. Al Adel, N. Alzaidi, Deep learning based computer-aided diagnosis systems for diabetic retinopathy: A survey, Artificial Intelligence in Medicine. 99 (2019) 101701. https://doi.org/10.1016/j.artmed.2019.07.009

[19] F. Juefei-Xu, V. N. Boddeti, M. Savvides, Local binary convolutional neural networks, in: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017. https://doi.org/10.1109/CVPR.2017.456

[20] F. Juefei-Xu, M. Savvides, Weight-Optimal Local Binary Patterns, in: L. Agapito, M.M. Bronstein, C. Rother (Eds.), Computer Vision - ECCV 2014 Workshops, Springer International Publishing, Cham, 2015: pp. 148-159. https://doi.org/10.1007/978-3-319-16181-5_11

[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, ImageNet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009: pp. 248-255. https://doi.org/10.1109/CVPR.2009.5206848

[22] M. W. M. Wintergerst, D. K. Mishra, L. Hartmann, P. Shah, V. K. Konana, P. Sagar, M. Berger, K. Murali, F. G. Holz, M. P. Shanmugam, R. P. Finger, Diabetic Retinopathy Screening Using Smartphone-Based Fundus Imaging in India, Ophthalmology. 127 (2020) 1529-1538. https://doi.org/10.1016/j.ophtha.2020.05.025

[23] R. Rajalakshmi, S. Arulmalar, M. Usha, V. Prathiba, K. S. Kareemuddin, R. M. Anjana, V. Mohan, Validation of Smartphone Based Retinal Photography for Diabetic Retinopathy Screening, PLOS ONE. 10 (2015) e0138285. https://doi.org/10.1371/journal.pone.0138285

[24] Automated diabetic retinopathy detection in smartphone-based fundus photography using artificial intelligence | Eye, (n.d.). https://www.nature.com/articles/s41433-018-0064-9 (accessed January 25, 2022)

[25] J. M. S. Prewit, Object enhancement and extraction, Picture processing and Psychopictorics, 1970

[26] Kaggle Competitions, (n.d.) https://www.kaggle.com/competitions (accessed February 12, 2022)

[27] S. Kajan, J. Goga, K. Lacko, J. Pavlovičová, Detection of Diabetic Retinopathy Using Pretrained Deep Neural Networks, in: Conference Cybernetics & Informatics, Velké Karlovice, Czech Republic, 2020

[28] F. Shao, Y. Yang, J. Qiuping, G. Jiang, Y.-S. Ho, Automated Quality Assessment of Fundus Images via Analysis of Illumination, Naturalness and Structure, IEEE Access. PP (2017) 1-1. https://doi.org/10.1109/ACCESS.2017.2776126

[29] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE. 86 (1998) 2278-2324. https://doi.org/10.1109/5.726791

[30] A. Krizhevsky, One weird trick for parallelizing convolutional neural networks, ArXiv:1404.5997 [Cs]. (2014) http://arxiv.org/abs/1404.5997 (accessed February 4, 2021)

[31] PyTorch, From research to production, (n.d.) https://pytorch.org/

[32] J. Duchi, E. Hazan, Y. Singer, Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, Journal of Machine Learning Research. 12 (2011) 2121-2159

[33] M. D. Zeiler, ADADELTA: An Adaptive Learning Rate Method, ArXiv:1212.5701 [Cs] (2012) http://arxiv.org/abs/1212.5701 (accessed February 9, 2021)

[34] I. Loshchilov, F. Hutter, Decoupled Weight Decay Regularization, ArXiv:1711.05101 [Cs, Math] (2019) http://arxiv.org/abs/1711.05101 (accessed February 9, 2021)

[35] T. Dozat, Incorporating nesterov momentum into adam (2016)

[36] NAdam — PyTorch 1.10.0 documentation, (n.d.) https://pytorch.org/docs/stable/generated/torch.optim.NAdam.html#torch.optim.NAdam (accessed November 16, 2021)

[37] T. Fawcett, An introduction to ROC analysis, Pattern Recognition Letters. 27 (2006) 861-874. https://doi.org/10.1016/j.patrec.2005.10.010

[38] O. Sagi, L. Rokach, Ensemble learning: A survey, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 8 (2018) e1249