Biomedical Signal Processing and Control 76 (2022) 103685

Contents lists available at ScienceDirect

Biomedical Signal Processing and Control

journal homepage: www.elsevier.com/locate/bspc

Enhancing CNNs through the use of hand-crafted features in automated fundus image classification

Gergo Bogacsovics [a,*], Janos Toth [a], Andras Hajdu [a], Balazs Harangi [a]

[a] Department of Data Science and Visualization, Faculty of Informatics, University of Debrecen, 4002 Debrecen PO Box 400, Hungary

ARTICLE INFO

Keywords:
Deep learning
Diabetic macular edema
Diabetic retinopathy
Ensemble learning
Hand-crafted features
Screening systems

ABSTRACT

Eye diseases such as diabetic retinopathy and diabetic macular edema pose a major threat in today's world as they affect a significant portion of the global population. Therefore, it is of utmost importance to develop robust solutions that can accurately detect these diseases, especially in their early stages. However, current methods, based on hand-crafted features devised by experts, are not sufficiently accurate. Several solutions have been proposed that use deep learning techniques to improve the performance of such systems. However, they ignore the highly valuable hand-crafted features, that could contribute to more accurate prediction, which underlines the significance of our research. In this paper, we revisit the problem of combining these hand-crafted features with the features extracted by neural networks with the objective of delivering more accurate predictions. We systematically study several state-of-the-art neural networks and methods and propose a number of ways to integrate them into our framework. We show that we arrived at the conclusion that it is possible to achieve significantly better results and outperform networks that do not consider hand-crafted features using the proposed methods.

1. Introduction

Nowadays, diabetic retinopathy (DR) is the most common cause of blindness in developed countries. It is an eye disease caused by longstanding diabetes. Moreover, about 1 in 15 people with diabetes will develop diabetic macular edema (DME). DME occurs when blood vessels of the retina leak fluid into the macula which causes blurry vision. In 2019, an estimated 1.5 million deaths were directly caused by diabetes and the World Health Organization estimates that 422 million people worldwide have the disease [1]. Progression to vision impairment can be slowed down if DR/DME is detected at an early stage.

Diabetic retinopathy (DR) is a diabetes complication, with possible consequences ranging from mild visual impairment to blindness. Currently, detecting the signs of DR/DME is a time-consuming and manual process that requires a trained clinician to examine and evaluate digital color fundus photographs of the retina. During the annual screening, a large number of digital images are taken and evaluated by an ophthalmologist. Only in the UK, the screening programs result in around two million retinal images for evaluation each year [2]. Repeated screening of DR is therefore not only costly but also exhausting, so there is a significant need to automate this process.

Expectations for trustworthy automated screening systems are high in the case of DR and DME. Recently, deep learning has been increasingly used in the field of medical image analysis, and many such methods have been proposed that use convolutional neural networks (CNNs) to detect microaneurysms (MAs), hemorrhages (HEs), hard exudates (EXs), or soft exudates (SEs). In [3], Shan et al. proposed a solution in which the entire fundus image was divided into patches that were considered as input without any further preprocessing steps. Then, the applied Stacked Sparse Autoencoder automatically extracted the distinguishing features to classify these patches. Tan et al. [4] showed how they used a single CNN to automatically segment and discriminate lesions on fundus images. They also divided the input image into 51×51 pixel patches and used their convolutional neural network to classify them as background, EXs, HEs, or MAs.

In addition to the publications mentioned above, there are many other papers in which the relevant features are extracted using conventional digital image processing tools. Walter et al. [5] used morphological processes and kernel density estimation to segment MAs and evaluated the input images based on the number of detected lesions.

* Corresponding author.
E-mail addresses: bogacsovics.gergo@inf.unideb.hu (G. Bogacsovics), toth.janos@inf.unideb.hu (J. Toth), hajdu.andras@inf.unideb.hu (A. Hajdu), harangi.balazs@inf.unideb.hu (B. Harangi).

https://doi.org/10.1016/j.bspc.2022.103685
Received 26 October 2021; Received in revised form 9 March 2022; Accepted 9 April 2022
Available online 18 April 2022

Quellec et al. [6] adapted optimal wavelet transform and template matching to perform automatic segmentation of MAs in retinal images. In [7], Agurto et al. proposed the use of multiscale amplitude modulation-frequency modulation (AM-FM) as a feature extraction method to discriminate between normal and pathological retinal images.

In our previous work [8], we proposed a solution in which we combine the powerful, self-extracted, CNN-based features with traditional, hand-crafted ones into a single framework to enhance classification performance. Our idea derived from [9], where the authors claimed that a CNN can automatically extract many important local, textual features from images by convolving with a sliding window and forming a filter. However, besides the local features, the global image descriptors also have been playing an important role in many image processing tasks. While the local features can be called textural features, the global features usually mean contour features and structural features. In the case of DR and DME, the presence of diseases is characterized by detecting one or more retinal lesions like MAs, HEs, EXs, and SEs. These signs can be described well by their contour feature and we can successfully apply them to improve the final accuracy of a CNN-based screening system.

In this paper, we extend our previous work by investigating the applicability of some additional state-of-the-art neural networks. We show that our experimental results support our former statement, namely, that we can improve the final classification results of a CNN-based solution by using hand-crafted features besides deep learning ones. Moreover, we investigate the advantages of the proposed methodology and also its limitations by a comprehensive comparison, where we test different ways to concatenate the automatically extracted textural features with the hand-crafted ones.

2. Materials and methods

Deep learning techniques have been widely used for solving a variety of problems in the area of medical diagnosis [10–12]. The reason why these solutions are so popular is the fact that modern, state-of-the-art deep CNNs have the ability to automatically and efficiently extract features that are required for classification. This means that experts do not need to manually extract these features, thus saving time and effort, while still being able to extract features that are required for a precise diagnosis. Besides this positive property of CNNs we also need to consider that although these neural networks could take spatial relations into account by pooling local features into a global representation but they are known to perform sub-optimally when learning long-range patterns [13]. Baker et al. in [14] tested exhaustively their hypothesis that CNNs are sensitive to an object's local contour features but have no access to global shape information. There is also evidence that there are certain features that CNNs simply cannot learn, such as global features, like Scale Invariant Feature Transform (SIFT) and Histogram of Oriented Gradient (HOG) as mentioned in [9]. This suggests that it may be worth to somehow fuse the hand-crafted features, which can take into account these properties, and the features extracted by neural networks, which tend to perform better than the traditional solutions. Ultimately, this fusion could retain the best of both worlds, thus leading to better generalization and more accurate diagnosis capabilities.

In this section, we present methods to combine hand-crafted features with ones extracted by neural networks to deliver more accurate predictions. First, we introduce the datasets used for our experiments. We also outline how hand-crafted feature extraction works and what features are extracted in the process. Then, we show that these features can be combined with those extracted by a variety of CNNs in several different ways. Finally, we give a detailed overview of how these algorithms performed on our test set and compare the results of the neural networks with and without the hand-crafted features.

2.1. Datasets

In this section, we give a brief description of the publicly available datasets that were used to train and evaluate the methods described in this work. The Kaggle DR and Messidor datasets and the training part of the Indian Diabetic Retinopathy Image Dataset (IDRiD) were used for training, while all evaluations were performed using the test part of the IDRiD dataset.

2.1.1. Indian diabetic retinopathy image dataset

The IDRiD dataset [15] consists of 516 color fundus images divided into a training part of 413 images and a test part of 103 ones, approximately maintaining the mixture of disease stratification. The images have a resolution of 4288 × 2848 pixels and a 50° field of view. Each image is categorized according to the severity of DR (5 classes) and the risk of DME (3 classes). Regarding DR, the training (test) set contains 134 (33) images labeled as no DR (DR0), 20 (5) as mild DR (DR1), 136 (32) as moderate DR (DR2), 74 (19) as severe DR (DR3), and 49 (13) as proliferative DR (DR4). Concerning DME, the training (test) set includes 177 (44) images categorized as no risk of macular edema (DME0), 41 (10) as moderate risk of macular edema (DME1), and 195 (48) as high risk of macular edema (DME2) based on presence of hard exudates near the macular center.

2.1.2. Kaggle DR dataset

The Kaggle DR dataset is the train portion of the dataset provided by EyePACS for a DR grading challenge [16] held by Kaggle. It contains 35 126 color fundus images with resolutions ranging from 400 × 315 to 5184 × 3456 pixels and different fields of view. For each image, a DR grading score is provided: 25 810 of these images are categorized as DR0, 2443 as DR1, 5292 as DR2, 873 as DR3, and 708 as DR4. In addition, 7806 of the images in this dataset were labeled by an experienced local ophthalmologist for the risk of DME: 5949 of the images are categorized as DME0, 1033 as DME1, and 824 as DME2. It is important to note that several images in this dataset are affected by imaging artifacts, blurring, under- or overexposure.

2.1.3. Messidor dataset

The Messidor dataset [17] comprises 1200 color fundus images with three different resolutions (1440 × 960, 2240 × 1488, and 2304 × 1536 pixels) and a 45° field of view. Both DR and DME grading scores are available for the images of this dataset; however, DR scoring differs slightly from the method used for the other two datasets. Using the DR severity classes of IDRiD and Kaggle DR, 546 of the images in this dataset are categorized as DR0, 153 as DR1, 247 as DR2, and 254 as DR4. Regarding DME, 974 images are categorized as DME0, 75 as DME1, and 151 as DME2.

2.2. Extracting the hand-crafted features

The image-level and lesion-specific approaches used to extract the traditional features considered in our methods are described in this section.

2.2.1. Image-level feature extraction

For image-level feature extraction, we employed an amplitude and frequency modulation-based approach [7], which extracts various features from a retinal image by decomposing its green channel into AM-FM components at different scales [18]. A collection of twenty-five band-pass channel filters, coupled with four frequency scales, is used to generate the different scales for feature extraction. After the image features are extracted, the information is clustered into 30 groups using k-means clustering. As a result, a 30-element feature vector is generated that reflects the intensity, shape, and texture of the structures of the image.

2.2.2. *Lesion-specific feature extraction*

We employed two detector ensembles consisting of a set of ⟨preprocessing method, candidate extractor⟩ pairs (⟨PP, CE⟩ for short) organized into a voting system to extract lesion-specific features associated with microaneurysms (MAs) and exudates (EXs). By applying a PP to the input retinal image and a CE to the PP output, a ⟨PP, CE⟩ pair is formed. A ⟨PP, CE⟩ pair extracts a set of candidate lesions from the input image and functions as a single detector method in this way.**Number of MAs** Because MAs are the earliest manifestations of DR and indicators of its progression, their number is an important factor in DR classification. MAs are capillary swellings that appear as tiny red dots; however, their similarity to vascular fragments makes them difficult to detect.

The combined output of the MA detector ensemble is obtained in the following way: If the Euclidean distances of the candidates in the result sets of ⟨PP, CE⟩ pairs are smaller than a given value, they are merged. The ratio of the number of ⟨PP, CE⟩ pairs suggesting this candidate to the total number of pairs in the ensemble is used to assign a confidence value to each common candidate in the ensemble.

We used the results of Antal and Hajdu [19] to form the ⟨PP, CE⟩ pairs of the ensemble. Five PPs (contrast-limited adaptive histogram equalization (CLAHE) [20], illumination equalization (IE) [21], vessel removal with inpainting (VR) [22], Walter-Klein contrast enhancement (WK) [23], and ”no preprocessing” (NP) for formal reasons) and three CEs (the method of Lazar et al. [24], Walter et al. [25], and Zhang et al. [26]) were selected. Table 1 (a) lists the ⟨PP, CE⟩ pairs in our MA detector ensemble.

After the MA candidate set of the ensemble is obtained, it is thresholded at six confidence levels and the number of candidates at each level is counted to obtain six features.**EX-specific features** EXs have properties useful in determining the severity of nonproliferative DR and the risk of DME. EXs are lipid residues of serous leakage from injured capillaries that appear as bright spots of various shapes in retinal images.

The combined output of the EX detector ensemble is obtained as follows: Each ⟨PP, CE⟩ pair produces a binary mask containing EX candidates. The probability that a pixel belongs to an EX is determined by the ratio of the number of pairs that marked the pixel as EX to the total number of pairs, which is then used to construct a probability map using the output of the different pairs.

The results of Nagy et al. [27] were used to create the ensemble described above. Four PPs (gray-world normalization (GN) [28], illumination equalization (IE) [21], morphological contrast enhancement (MC) [29], and vessel removal with inpainting (VR) [22]) and three CEs (the method of Sopharak et al. [30], Walter et al. [Walter2002], and Welfer et al. [31]) were selected. Table 1 (b) lists the eight ⟨PP, CE⟩ pairs that were used in the ensemble.

As the last step, the result set of the EX ensemble is thresholded at eight different confidence levels, resulting in a total of 32 features: the ratio of all EX pixels to ROI pixels, the number of EXs (8-connected components), the ratio of the largest EX (8-connected component) to ROI, and the average EX size to ROI.

Table 1
The ⟨PP, CE⟩ pairs of the (a) MA, and (b) EX detector ensembles.

(a)

	PP	CE
1	NP	Lazar et al.
2	CLAHE	Lazar et al.
3	IE	Lazar et al.
4	WK	Lazar et al.
5	NP	Walter et al.
6	CLAHE	Walter et al.
7	NP	Zhang et al.
8	VR	Zhang et al.
9	WK	Zhang et al.

(b)

	PP	CE
1	GN	Sopharak et al.
2	MC	Sopharak et al.
3	VR	Sopharak et al.
4	IE	Walter et al.
5	MC	Walter et al.
6	VR	Walter et al.
7	IE	Welfer et al.
8	MC	Welfer et al.

2.3. *Combining the hand-crafted features with deep learning techniques*

In our previous work [8], we showed that by merging the hand-crafted features with the features extracted by a CNN we could attain state-of-the-art accuracy for this problem. To achieve this, we extended the last fully connected (FC) layer (4096 neurons) with additional neurons (68 in total) and then reduced the weights to the given class probabilities. More precisely, for any given image $x^{(i)}$, we calculated the feature vectors extracted by AlexNet and the traditional extractors, denoting them as $y_1^{(i)}$ and $y_2^{(i)}$ respectively. The hand-crafted features were normalized to bring $y_2^{(i)}$ to the same scale as $y_1^{(i)}$. Then, we concatenated these vectors to get $y_c^{(i)} = < y_1^{(i)}, y_2^{(i)} >$ and passed them through an additional fully connected layer. Namely, we used a weight matrix A of size 4164 x 5 (or 4164 x 3 for DME) to calculate $\overline{y}^{(i)} = A^\top y_c^{(i)}$. Then, we used the softmax function to obtain the predictive class probabilities $\widehat{y}^{(i)}$.

2.3.1. *Extending the original method by using other architectures*

Although we managed to get really good results with this solution, it can also be seen how optimizing for more descriptive $y_1^{(i)}$, in other words, choosing the architecture, can affect performance. This is why in this paper, we extend this idea to several commonly used networks other than AlexNet [32] and objectively compare the results of those networks and their variants that use the hand-crafted features to demonstrate that our solution greatly improves the classification accuracy. All the architectures received the input images in the shape that the original networks operated on. This meant that each RGB input image $x^{(i)}$ was resized to the size of 224x224x3 before being given to the given network. During the research, we chose AlexNet as our baseline network and tried to improve its accuracy. To this end, we used several state-of-the-art networks, such as MobileNetv3 [33] and Resnet-50 [34] and compared their results to that of the baseline – both with and without the hand-crafted features. In the case of the first one, for any given input image $x^{(i)}$ the hand-crafted features were calculated and then concatenated with the extracted CNN features of the given network, while for the latter one only the input image was given to the algorithm. For this version V1 of the algorithm, we kept the original structure [8] but swapped the feature extracting part with the given network (AlexNet, MobileNetv3, and Resnet-50, respectively), as can be seen in Fig. 1.

While by substituting AlexNet with the other variants, we could greatly improve classification accuracy as can be seen later in Section 3, but this method had some shortcomings. Namely, we can clearly see that although we take into account both the hand-crafted and CNN features, we only pass them through one layer by calculating $y_c^{(i)} = < y_1^{(i)}, y_2^{(i)} >$. This makes combining $y_1^{(i)}$ and $y_2^{(i)}$ hard, since we need to do that with only a single hidden layer (A). For this reason, in this paper we also explore new architectural solutions that may more effectively combine the information extracted by these hand-crafted features with that of the features extracted by a neural network.

2.3.2. *Learning the features in parallel*

One reasonable solution is to divide the computational graph of the

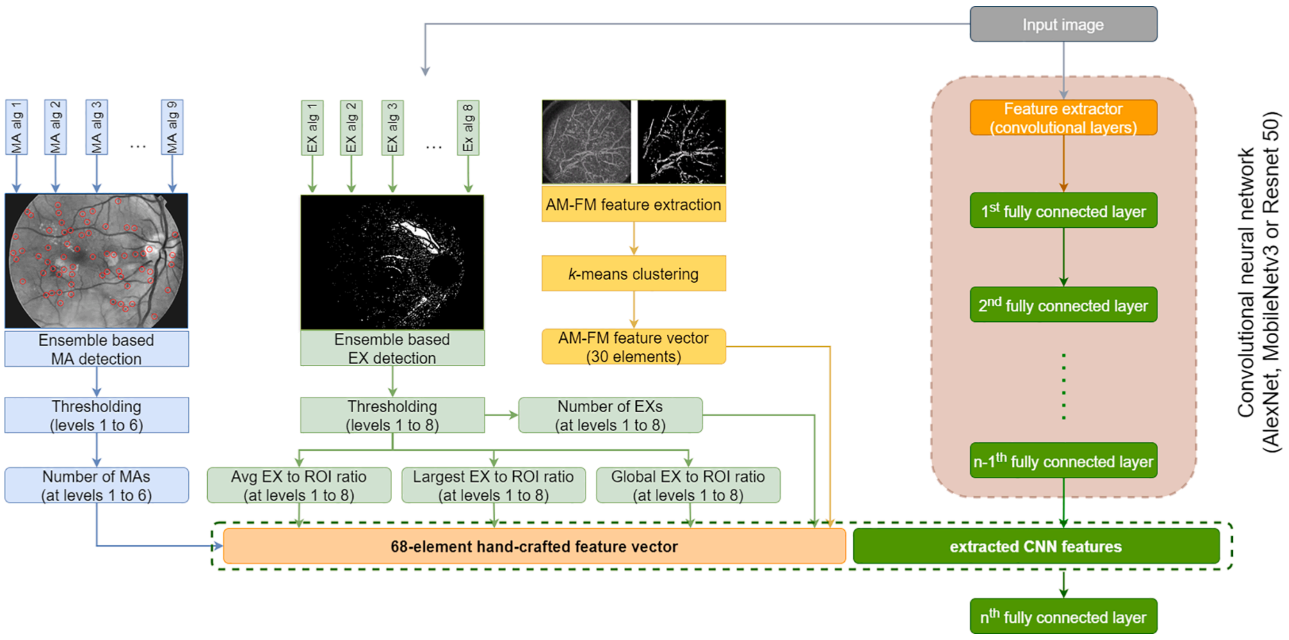
Input image
MA alg 1
MA alg 2
MA alg 3
…
MA alg 9
EX alg 1
EX alg 2
EX alg 3
…
Ex alg 8
Ensemble based MA detection
Ensemble based EX detection
AM-FM feature extraction
k-means clustering
AM-FM feature vector (30 elements)
Thresholding (levels 1 to 6)
Number of MAs (at levels 1 to 6)
Thresholding (levels 1 to 8)
Number of EXs (at levels 1 to 8)
Avg EX to ROI ratio (at levels 1 to 8)
Largest EX to ROI ratio (at levels 1 to 8)
Global EX to ROI ratio (at levels 1 to 8)
68-element hand-crafted feature vector
extracted CNN features
Feature extractor (convolutional layers)
1st fully connected layer
2nd fully connected layer
n-1th fully connected layer
nth fully connected layer
Convolutional neural network (AlexNet, MobileNetv3 or Resnet 50)

Fig. 1. Fusing the hand-crafted features with those extracted by a CNN in the last layer of the given neural network.

network into two sub-parts as can be seen in Fig. 2: one that is responsible for processing hand-crafted features and one that processes the input image. These parts would calculate their predictions separately and we could merge the results to get our final predictions. Namely, instead of calculating the feature vectors $y_1^{(i)}$ and $y_2^{(i)}$, we could calculate the approximate predictions $\overline{y}_1^{(i)}$ and $\overline{y}_2^{(i)}$ for each part and then average the outputs. This would in fact act like a "mini" ensemble network that

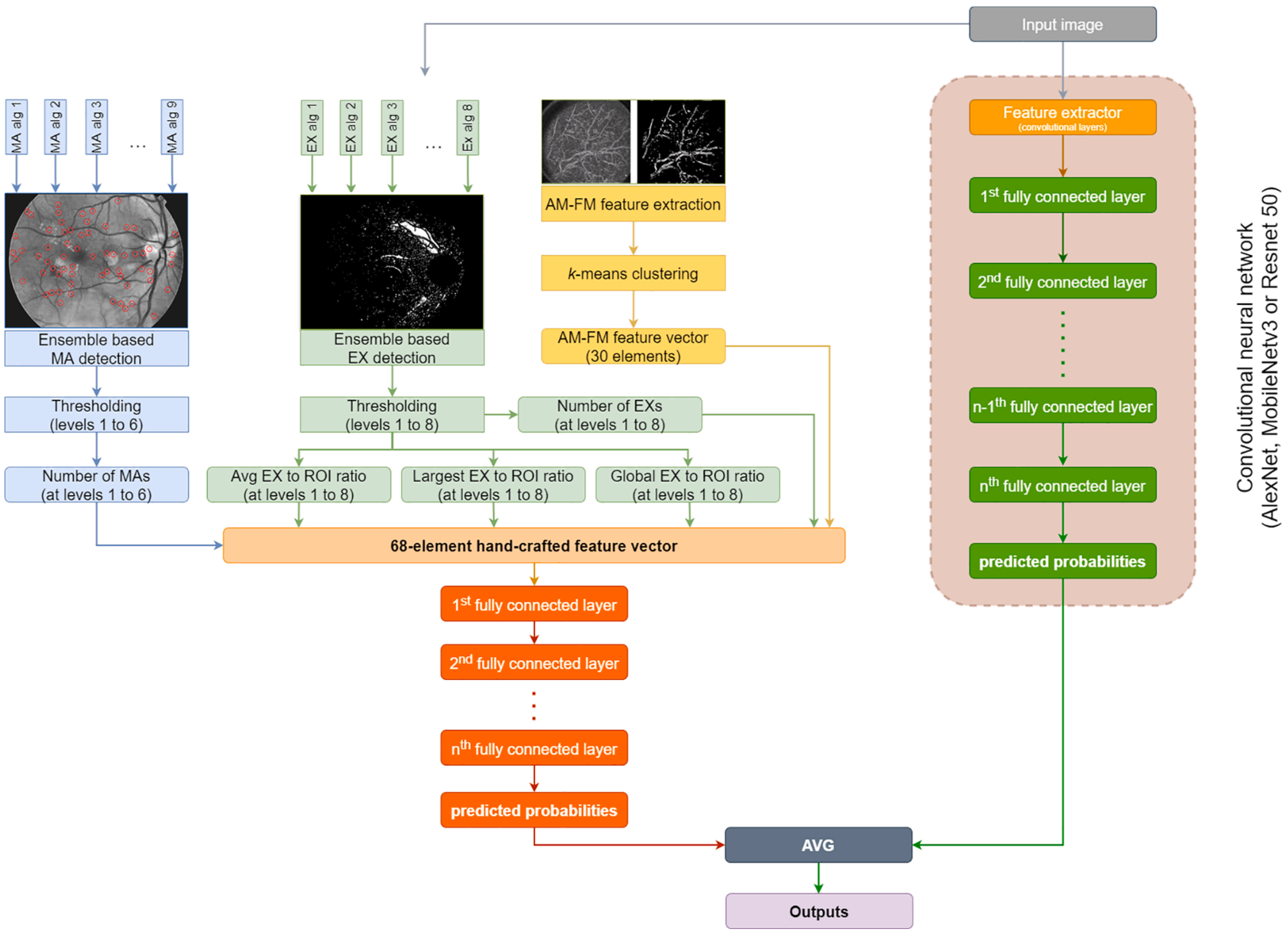
Input image
MA alg 1
MA alg 2
MA alg 3
…
MA alg 9
EX alg 1
EX alg 2
EX alg 3
…
Ex alg 8
Ensemble based MA detection
Ensemble based EX detection
AM-FM feature extraction
k-means clustering
AM-FM feature vector (30 elements)
Thresholding (levels 1 to 6)
Number of MAs (at levels 1 to 6)
Thresholding (levels 1 to 8)
Number of EXs (at levels 1 to 8)
Avg EX to ROI ratio (at levels 1 to 8)
Largest EX to ROI ratio (at levels 1 to 8)
Global EX to ROI ratio (at levels 1 to 8)
68-element hand-crafted feature vector
1st fully connected layer
2nd fully connected layer
nth fully connected layer
predicted probabilities
Feature extractor (convolutional layers)
1st fully connected layer
2nd fully connected layer
n-1th fully connected layer
nth fully connected layer
predicted probabilities
Convolutional neural network (AlexNet, MobileNetv3 or Resnet 50)
AVG
Outputs

Fig. 2. Fusing the hand-crafted features with those extracted by a CNN by learning in two separate paths.

uses majority voting, but where each path of the computational graph is trained at the same time. This is possible since the backpropagation of the predicted error is executed with one step from the calculated predictions:

$$\overline{y}^{(i)} = \frac{\overline{y}_1^{(i)} + \overline{y}_2^{(i)}}{2}. \tag{1}$$

This approach has many strong points. First of all, the features are processed separately and undergo more steps, making it possible to derive more high-level information. Moreover, we could benefit from the advantages of using an ensemble solution, which often yields better and more stable results. Finally, the network could also learn when it should emphasize the hand-crafted features $y_1^{(i)}$ and when the CNN features $y_2^{(i)}$, depending on the input image $x^{(i)}$ by adjusting its weights accordingly. This can be especially important if sometimes the first one and sometimes the latter one is more accurate, and when thus having the ability to weight these predictions depending on the input image $x^{(i)}$ could result in more stable predictions.

2.3.3. *A deeper combination of the hand-crafted and CNN features*

The method described in 2.3.2 guarantees that both the features extracted by the CNN and the hand-crafted ones undergo a longer processing step due to the increased number of fully connected layers. However, in the long run, it may not always be beneficial to entirely separate the processing of the hand-crafted and CNN features as it may make the learning process much harder due to the increased number of parameters and by having the network concentrate on essentially two different things: making sense of the hand-crafted features and making sense of the input image. This could greatly disrupt the learning process, ultimately leading to poor results in some cases. Instead, we could move the concatenation step $y_c^{(i)} =< y_1^{(i)}, y_2^{(i)} >$ up in the algorithm described in 2.3.1 to merge the features $y_1^{(i)}$ that have been extracted by purely the convolutional layers with the hand-crafted ones, denoted as $y_2^{(i)}$, and then use more fully connected layers to process the joint features $y_c^{(i)}$. This process can be observed in Fig. 3.

With this approach, we could get the benefits of both worlds: the network would have more capabilities to process the feature vectors, thanks to the increased number of dense layers, and both the features extracted by the convolutional layers and the hand-crafted ones would be processed at the same time, layer by layer. This gives the network the ability to learn and recognize more complex patterns that would require both of them, ultimately leading to more robust and accurate predictions.

3. Results and discussion

In this section, we explain what methods we used to divide the dataset into training and validation sets and how the different hyper-parameters were optimized for each architecture. We also detail how the experimental results were obtained, what metrics we used, how we evaluated the different algorithms, and draw some experimental conclusions.

3.1. *Cross-validation*

As noted in 2.1, we used a total of 36 739 images (413 from IDRiD, 35 126 from the Kaggle and 1200 from the Messidor dataset) for training and the test part of the IDRiD dataset for testing. To get a better understanding of the results, we divided the dataset into smaller cross-validation sets. For each set and every problem type (DME or DR), we divided the original data into training and validation parts in the ratio of 4:1 (80% training, 20% validation), while keeping the relative frequencies of the different classes the same. With the latter, we tried to

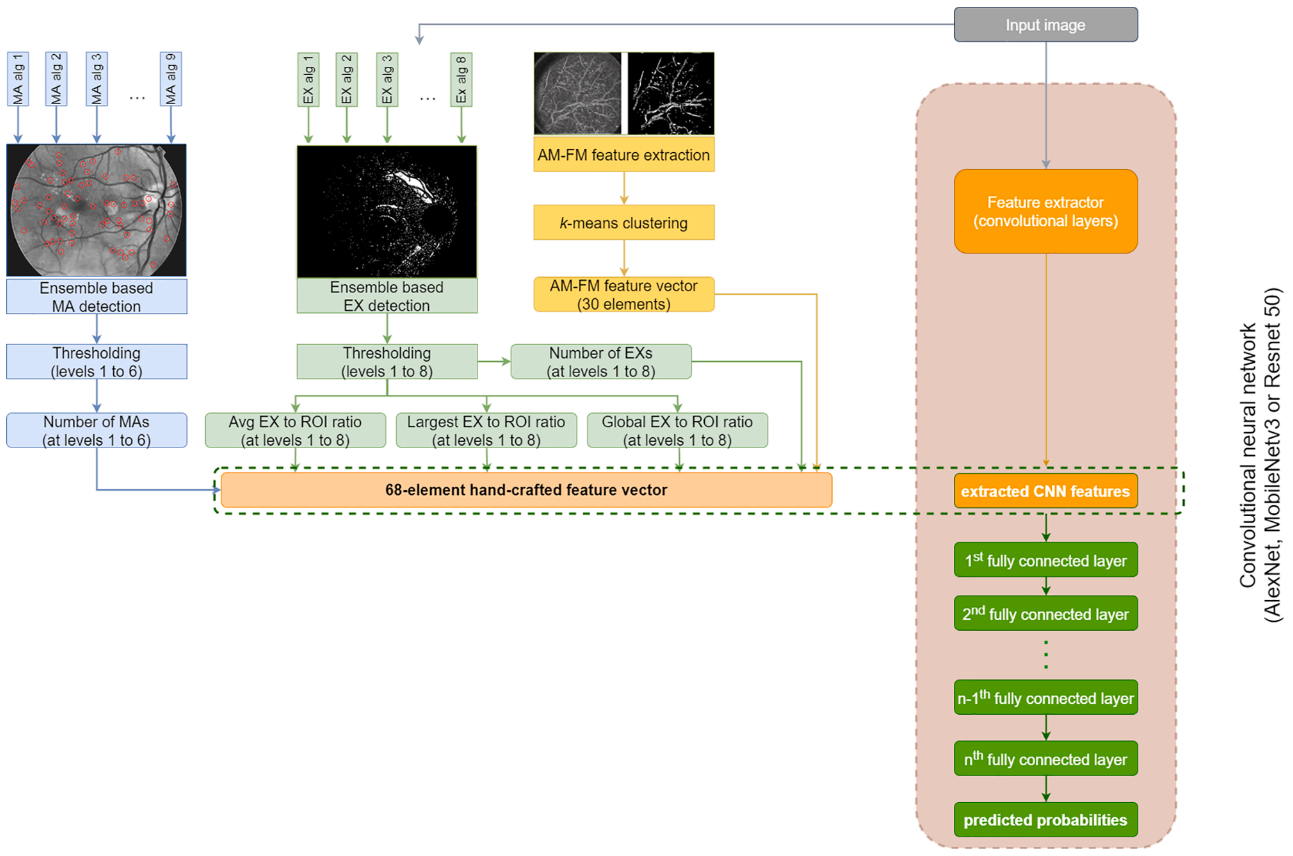
Input image
MA alg 1
MA alg 2
MA alg 3
...
MA alg 9
EX alg 1
EX alg 2
EX alg 3
...
Ex alg 8
Ensemble based MA detection
Ensemble based EX detection
AM-FM feature extraction
k-means clustering
AM-FM feature vector (30 elements)
Thresholding (levels 1 to 6)
Thresholding (levels 1 to 8)
Number of EXs (at levels 1 to 8)
Number of MAs (at levels 1 to 6)
Avg EX to ROI ratio (at levels 1 to 8)
Largest EX to ROI ratio (at levels 1 to 8)
Global EX to ROI ratio (at levels 1 to 8)
68-element hand-crafted feature vector
Feature extractor (convolutional layers)
extracted CNN features
1st fully connected layer
2nd fully connected layer
n-1th fully connected layer
nth fully connected layer
predicted probabilities
Convolutional neural network (AlexNet, MobileNetv3 or Resnet 50)

Fig. 3. Fusing the hand-crafted features with those extracted by a CNN by moving the concatenation step up in the architecture.

mitigate the distribution shift between the training and validation set. Furthermore, there was no overlap between the cross-validation sets, meaning that the validation sets were disjunct. This meant that if an image appeared in the validation set of one cross-validation set, then it could no longer appear in the validation set of any subsequent cross-validation set. As for the classes themselves, we made sure that all annotations followed the same grading procedure. The manual annotations of the images were made by ophthalmologists according to the International Clinical Diabetic Retinopathy (ICDR) disease severity scale [35] for all three of the original image datasets (Messidor [17], Kaggle [16], and IDRiD [15] datasets) that we used. The ICDR scale is one of the most commonly used clinical scales and consists of a 5-point grade for DR: no, mild, moderate, severe, and proliferative.

3.2. Hyperparameter search

We thoroughly searched the optimal hyperparameters for each algorithm. This search included looking for the optimal batch size, optimizer, learning rate, and the number of epochs. The hyperparameter search was naturally carried out purely on the training set. In the case of optimizers, we experimented with Stochastic Gradient Descent (SGD) [36] (both with and without momentum) and Adam [37] and looked for the one that made the loss decrease in the smoothest way possible, with minimal oscillations and continuous decreases over the epochs. For the batch size, we looked for the value that made the learning process the most stable and led to the smallest oscillation in the training loss. In the case of the learning rate, we used learning rate scheduling and examined the change of the loss. Then, we picked the learning rate lr that had the lowest observed loss value and had a sufficiently large environment $[lr-\epsilon, lr+\epsilon]$ in which the loss values did not oscillate and did not increase rapidly. Finally, we chose the number of epochs so that training was terminated when the validation loss started to continuously increase.

We found the batch size of 32 and the Adam optimizer to be the best for every algorithm, while the optimal learning rate and the number of epochs varied from network to network. Table 2 and 3 below summarize the best hyperparameter settings for each network and problem type (DR and DME). These were the ones that we used for evaluating the different algorithms in Section 3.5.

3.3. Evaluation

All of the algorithms were evaluated on the test part of the IDRiD dataset, as described in 2.1. Each image $x^{(i)}$ had its corresponding DR and DME labels, as well as the hand-crafted features associated with the given image. As the test set that we had access to was heavily imbalanced and skewed towards certain classes, we took several metrics into

Table 2
The list of optimal hyperparameters for the DR problem.

Network	*Type*	*Optimizer*	Batch size	Learning rate	*Epochs*
AlexNet	original	Adam	32	0.0001	100
MobileNetv3	original	Adam	32	0.001	150
ResNet-50	original	Adam	32	0.001	100
AlexNet	V1	Adam	32	0.0001	100
MobileNetv3	V1	Adam	32	0.001	150
ResNet-50	V1	Adam	32	0.0001	100
AlexNet	V2	Adam	32	0.001	150
MobileNetv3	V2	Adam	32	0.001	200
ResNet-50	V2	Adam	32	0.001	150
AlexNet	V3	Adam	32	0.0001	150
MobileNetv3	V3	Adam	32	0.0003	200
ResNet-50	V3	Adam	32	0.0001	100

Table 3
The list of optimal hyperparameters for the DME problem.

Network	*Type*	*Optimizer*	Batch size	Learning rate	*Epochs*
AlexNet	original	Adam	32	0.0001	150
MobileNetv3	original	Adam	32	0.0001	200
ResNet-50	original	Adam	32	0.001	200
AlexNet	V1	Adam	32	0.0003	150
MobileNetv3	V1	Adam	32	0.0001	200
ResNet-50	V1	Adam	32	0.001	200
AlexNet	V2	Adam	32	0.001	150
MobileNetv3	V2	Adam	32	0.001	200
ResNet-50	V2	Adam	32	0.001	150
AlexNet	V3	Adam	32	0.0003	200
MobileNetv3	V3	Adam	32	0.0003	200
ResNet-50	V3	Adam	32	0.001	200

account during the evaluation process to make it fairer. Namely, since a standard accuracy value would give biased results in our case, we used a weighted average of the accuracies calculated at the class level, where each weight represented how many times a given class occurred in the test set. Furthermore, we calculated a number of metrics regarding the performance of the algorithms and their ability to differentiate healthy and non–healthy images, such as Positive Predictive Value (PPV), Sensitivity (SE), Specificity (SP), F_1-score, and weighted accuracy (ACC_w). To calculate these, we considered the following quantities for every prediction: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). In our case TP meant that both the prediction and the ground truth said the given image $x^{(i)}$ belonged to a non–healthy specimen, while TN meant the same but with healthy specimens. The number of FPs indicated how many times the algorithm predicted the non–healthy label, while the ground truth was healthy, and the number of FNs showed the exact opposite. The exact formulas of the used metrics are given as:

$$PPV = \frac{TP}{TP+FP} \tag{2}$$

$$SE = \frac{TP}{TP+FN}, \quad SP = \frac{TN}{TN+FP} \tag{3}$$

$$F_1 - \text{score} = \frac{2TP}{2TP+FP+FN} \tag{4}$$

$$ACC = \frac{TP+TN}{TP+FP+FN+TN} \tag{5}$$

$$ACC_w = \sum_{c \in classes} w_c * ACC(c), \tag{6}$$

where w_c is the ratio of the samples in the class c to the total number of samples.

3.4. Project setup

The experiments were carried out on a computer with an Nvidia RTX 3080 video card and a total of 128 GB RAM. The codebase was written purely in Python and we used the PyTorch framework for writing the experiment code.

3.5. Experimental results

As noted in 3.3, we used a variety of metrics to reliably evaluate the performance of the different algorithms, which is why we have included multiple tables for each problem type (DR and DME). Furthermore, as we noted in 3.1, we divided the dataset into cross-validation sets. Consequently, all the tables in this section show the results of the different algorithms measured after being trained on both of these cross-validation sets. The results shown are calculated at 95% confidence levels and are sorted by network and algorithm type. For the original networks, we refer to as *original*, while the algorithms introduced in 2.3.1, 2.3.2 and 2.3.3 are noted as V1, V2, and V3, respectively. We would also like to note that since the ResNet-50 architecture only had a total of one fully connected layer, the V3 version of the network is fully equivalent to the V1 version. Therefore, the tables presented contain the same results for these two versions of the ResNet-50 architecture.

We have also trained several baselines that only used the hand-crafted features $y_2^{(i)}$ but did not use the input images to measure the predictive capabilities of the hand-crafted features alone. We have considered three different approaches for this task. In the first approach, we optimized directly for the best set of weights A to calculate $\overline{y}^{(i)} = A^\top y_2^{(i)}$, then, similarly to Section 2.3.1, we used the softmax function to calculate the predicted class probabilities $\widehat{y}^{(i)}$. We will refer to this approach as Softmax in the subsequent tables. In the second and third approaches, we used an SVM and a neural network (MLP) respectively to calculate $\widehat{y}^{(i)}$. For the SVM, we used the Radial Basis Function (RBF) kernel, while for the MLP, we used 2 hidden layers with 64 and 32 units respectively. Furthermore, we have also considered another set of architectures from [38] as additional baselines to compare our results with. We will refer to this last approach as LightCNN in all the subsequent tables while also specifying the type of architecture used as discussed in the original paper [38] by abbreviating random forest with RF and decision tree with DT.

3.5.1. Diabetic retinopathy

First, we measured the ability of the networks to differentiate between healthy and non–healthy images as a starting point. While the networks that used only the hand-crafted features performed surprisingly well, they achieved substantially worse results in terms of *SE* and F_1-score as compared to the original networks that received only the images as inputs. In other words, – according to the formulas (2) and (3) – the number of *FN*s was substantially greater than the number of *FP*s for the networks that only used the hand-crafted features. This increase in the *FN*s resulted in the model misclassifying numerous ill specimens as non–healthy. The architectures proposed in [38] achieved really good results similar to that of the original ResNet-50 model but were still outperformed by the networks that used the hand-crafted features. While all of the original networks were able to more or less determine these classes without the hand-crafted features, as can be seen in Table 4, the usage of hand-crafted features improved the performance greatly in all cases. The improvements were the most substantial for networks derived from the original AlexNet architecture, while the best performing algorithms were the V2 versions of the two newer networks.

Since all of the networks were able to differentiate between healthy and non–healthy specimens, we measured their performance with regard to their accuracies per class. These class-level accuracies can be seen in Table 5. The networks that used only the hand-crafted features performed surprisingly well yet again for some classes (DR1, DR3 and DR4) but performed really poorly for other classes (DR2). The reason behind their remarkably good results for DR4 was that they did not predict DR4 for any given sample; meaning that they only predicted 0 values for this class. Since the total number of DR4 specimens was really low compared to other classes, this resulted in only a handful of DR4 cases, leading to a great number of 0 values and only a few 1 values in the ground truth, which combined with the fact that the networks predicted 0 values for each sample resulted in a high accuracy value for this class. The architectures proposed in [38] delivered much better results, surpassing several networks that only used the input images but they were still outperformed by the networks that used both the hand-crafted features and image inputs. For these networks, while the same ones (the V2 version of MobileNetv3 and Resnet-50) seemed to perform the best for DR0 and DR1 as in Table 4, for the other classes, there were better performing alternatives. 3 out of 5 of these used hand-crafted features with DR4 being an exception due to its low cardinality, and even in the case of DR3 and DR4, the difference between the best and second-best network (which used the hand-crafted features) was negligible (0.005 and 0.005). While on the other hand, in the case of the other classes, the difference between networks without and with the hand-crafted features was quite substantial, in favor of the latter. This demonstrates again that using the hand-crafted features is beneficial for any of these networks and can drastically increase the accuracy.

To get a general overview of how the different networks performed on the test set, we also measured their weighted accuracies. For this purpose, we calculated the accuracies on the class level and then weighted the results depending on the number of samples per class in the test set. The results can be seen in Table 6. It can be seen that although the networks that only used the hand-crafted features performed well for some classes as we noted for Table 5, the overall accuracy of these networks was substantially lower than that of the networks that used the input images as well. The architectures proposed in [38] performed considerably better but were yet again outperformed by some of the networks that used both the hand-crafted features and the input images.

Table 4
A summary of the metrics measured on the DR dataset.

Network	*Type*	*PPV*	*SE*	*SP*	F_1-score
LightCNN [38]	SVM	0.905 ± 0.044	0.804 ± 0.128	0.824 ± 0.115	0.849 ± 0.052
LightCNN [38]	RF	0.858 ± 0.017	0.826 ± 0.142	0.721 ± 0.086	0.840 ± 0.066
LightCNN [38]	MLP	0.909 ± 0.038	0.790 ± 0.071	0.838 ± 0.087	0.845 ± 0.025
LightCNN [38]	DT	0.801 ± 0.003	0.848 ± 0.043	0.574 ± 0.029	0.824 ± 0.019
Hand-crafted	Softmax	0.780 ± 0.012	0.616 ± 0.043	0.647 ± 0.001	0.688 ± 0.031
Hand-crafted	SVM	0.793 ± 0.042	0.638 ± 0.001	0.662 ± 0.086	0.707 ± 0.017
Hand-crafted	MLP	0.888 ± 0.017	0.746 ± 0.014	0.809 ± 0.029	0.811 ± 0.015
AlexNet	original	0.746 ± 0.053	0.783 ± 0.142	0.412 ± 0.173	0.774 ± 0.016
MobileNetv3	original	0.820 ± 0.017	0.855 ± 0.028	0.544 ± 0.086	0.828 ± 0.012
ResNet-50	original	0.908 ± 0.108	0.797 ± 0.085	0.824 ± 0.231	0.846 ± 0.001
AlexNet	V1	0.741 ± 0.005	0.833 ± 0.014	0.662 ± 0.087	0.800 ± 0.021
MobileNetv3	V1	0.893 ± 0.054	0.841 ± 0.000	0.794 ± 0.115	0.866 ± 0.025
ResNet-50	V1	0.932 ± 0.029	0.783 ± 0.028	0.882 ± 0.058	0.850 ± 0.005
AlexNet	V2	0.880 ± 0.030	0.783 ± 0.028	0.794 ± 0.058	0.809 ± 0.001
MobileNetv3	V2	0.876 ± 0.056	**0.877** ± **0.099**	0.750 ± 0.140	**0.873** ± **0.019**
ResNet-50	V2	**0.938** ± **0.013**	0.797 ± 0.028	**0.897** ± **0.029**	0.840 ± 0.021
AlexNet	V3	0.835 ± 0.013	0.848 ± 0.071	0.691 ± 0.029	0.824 ± 0.024
MobileNetv3	V3	0.924 ± 0.047	0.783 ± 0.000	0.868 ± 0.087	0.847 ± 0.020
ResNet-50	V3	0.932 ± 0.029	0.783 ± 0.028	0.882 ± 0.058	0.850 ± 0.005

Table 5
A summary of the class-level accuracies measured on the DR dataset.

Network	*Type*	DR0	DR1	DR2	DR3	DR4
LightCNN [38]	SVM	0.811 ± 0.048	0.951 ± 0.001	0.651 ± 0.057	0.835 ± 0.019	0.859 ± 0.029
LightCNN [38]	RF	0.791 ± 0.067	0.918 ± 0.048	0.675 ± 0.086	0.835 ± 0.019	0.869 ± 0.010
LightCNN [38]	MLP	0.806 ± 0.019	0.947 ± 0.010	0.694 ± 0.029	0.835 ± 0.057	0.874 ± 0.019
LightCNN [38]	DT	0.757 ± 0.019	0.850 ± 0.048	0.617 ± 0.029	0.777 ± 0.019	0.835 ± 0.001
Hand-crafted	Softmax	0.626 ± 0.029	0.951 ± 0.001	0.452 ± 0.010	0.816 ± 0.001	**0.874 ± 0.001**
Hand-crafted	SVM	0.646 ± 0.029	0.951 ± 0.001	0.495 ± 0.095	0.840 ± 0.010	**0.874 ± 0.001**
Hand-crafted	MLP	0.767 ± 0.019	0.951 ± 0.001	0.612 ± 0.019	0.835 ± 0.001	0.864 ± 0.001
AlexNet	original	0.695 ± 0.028	0.947 ± 0.009	0.534 ± 0.020	0.753 ± 0.105	0.762 ± 0.086
MobileNetv3	original	0.763 ± 0.028	0.937 ± 0.028	0.631 ± 0.057	0.840 ± 0.028	0.830 ± 0.010
ResNet-50	original	0.806 ± 0.020	0.942 ± 0.019	0.709 ± 0.095	**0.884 ± 0.019**	0.855 ± 0.019
AlexNet	V1	0.777 ± 0.038	0.951 ± 0.000	0.549 ± 0.085	0.840 ± 0.028	0.850 ± 0.009
MobileNetv3	V1	0.796 ± 0.020	0.947 ± 0.009	0.641 ± 0.019	0.850 ± 0.028	0.869 ± 0.010
ResNet-50	V1	0.806 ± 0.020	0.947 ± 0.009	0.728 ± 0.114	0.864 ± 0.020	0.859 ± 0.010
AlexNet	V2	0.738 ± 0.020	0.947 ± 0.009	0.622 ± 0.019	0.869 ± 0.028	0.753 ± 0.008
MobileNetv3	V2	**0.830 ± 0.010**	**0.956 ± 0.010**	0.612 ± 0.095	0.821 ± 0.028	0.845 ± 0.019
ResNet-50	V2	0.806 ± 0.020	**0.956 ± 0.010**	0.675 ± 0.048	0.879 ± 0.028	0.788 ± 0.002
AlexNet	V3	0.758 ± 0.019	0.932 ± 0.037	0.622 ± 0.038	0.826 ± 0.019	0.731 ± 0.008
MobileNetv3	V3	0.811 ± 0.028	0.937 ± 0.028	**0.743 ± 0.028**	0.869 ± 0.010	0.831 ± 0.028
ResNet-50	V3	0.806 ± 0.020	0.947 ± 0.009	0.728 ± 0.114	0.864 ± 0.020	0.859 ± 0.010

We can once again see that by using the hand-crafted features, the overall accuracy has greatly improved for all of the networks. The best performing networks were once again the ones based on the MobileNetv3 and Resnet-50 architectures, but even the ones using AlexNet benefitted greatly from the usage of the hand-crafted features.

Table 6
The weighted accuracies measured on the DR dataset.

Network	*Type*	ACC_w
LightCNN [38]	SVM	0.778 ± 0.002
LightCNN [38]	RF	0.779 ± 0.007
LightCNN [38]	MLP	0.792 ± 0.016
LightCNN [38]	DT	0.731 ± 0.014
Hand-crafted	Softmax	0.654 ± 0.012
Hand-crafted	SVM	0.678 ± 0.041
Hand-crafted	MLP	0.752 ± 0.001
AlexNet	original	0.687 ± 0.011
MobileNetv3	original	0.771 ± 0.022
ResNet-50	original	0.807 ± 0.030
AlexNet	V1	0.723 ± 0.025
MobileNetv3	V1	0.799 ± 0.007
ResNet-50	V1	0.805 ± 0.048
AlexNet	V2	0.725 ± 0.004
MobileNetv3	V2	0.763 ± 0.021
ResNet-50	V2	0.788 ± 0.002
AlexNet	V3	0.731 ± 0.008
MobileNetv3	V3	0.803 ± 0.034
ResNet-50	V3	0.805 ± 0.048

3.5.2. *Diabetic macular edema*

For DME, we followed the exact same steps as outlined previously in 3.5.1. First, we measured how reliably the various algorithms could differentiate the healthy specimens from the non–healthy ones using the PPV, SE, SP, and F_1-score metrics. As it can be seen in Table 7, all the networks were able to achieve this task. The networks that only used the hand-crafted features had the exact same problem that we discussed in Section 3.5.1: their SE scores were substantially lower than that of the other networks, except for the network that used the two-layer neural network (MLP). As for the architectures proposed in [38], they achieved really good results, surpassing most (but not all) of the original networks but were still outperformed by the networks that used both the hand-crafted features and the input images. As for AlexNet, MobileNetv3 and ResNet-50, comparing the solutions that did not use the hand-crafted features with those that did, it is clearly visible that the results of the latter were substantially better. The network with the most notable and drastic improvements was MobileNetv3, which turned out to be the best performing solution regarding these metrics.

Next, we measured how accurately the networks could classify the specimens into different classes. In the case of DME, there were a total of 3 classes: DME0, DME1, and DME2. Table 8 shows the measured class-level accuracies. Among the networks that only used the hand-crafted features, the Softmax and SVM versions performed substantially worse than the other networks in the case of DME0 and DME2. The version that used the MLP as its backbone performed remarkably well, almost surpassing the original AlexNet network in all aspects but was outperformed by both the other original networks and the ones that used the hand-crafted features as well as the input images. The LightCNN variants performed quite uniformly, meaning that the accuracy values were almost the same for each architecture but were now outperformed by not only the networks that used the hand-crafted features and the input images but also by the original networks as well. It can be also seen that the networks using the hand-crafted features performed substantially better than the networks that did not use these features. Only the MobileNetv3 architecture was a slight exception since it achieved good results for the DME2 class. However, the results of this network without the hand-crafted features were significantly lower in the case of the other classes. The best performing network was the ResNet-50 network

Table 7
A summary of the metrics measured on the DME dataset.

Network	*Type*	*PPV*	*SE*	*SP*	F_1-score
LightCNN [38]	SVM	0.913 ± 0.001	0.724 ± 0.001	0.911 ± 0.001	0.808 ± 0.001
LightCNN [38]	RF	0.897 ± 0.002	0.750 ± 0.017	0.889 ± 0.001	0.817 ± 0.011
LightCNN [38]	MLP	0.922 ± 0.020	0.707 ± 0.001	0.922 ± 0.022	0.800 ± 0.008
LightCNN [38]	DT	0.898 ± 0.04	0.759 ± 0.034	0.889 ± 0.044	0.822 ± 0.037
Hand-crafted	Softmax	0.786 ± 0.027	0.285 ± 0.017	0.900 ± 0.022	0.418 ± 0.015
Hand-crafted	SVM	0.781 ± 0.142	0.595 ± 0.017	0.778 ± 0.174	0.674 ± 0.064
Hand-crafted	MLP	0.859 ± 0.003	0.733 ± 0.017	0.844 ± 0.001	0.791 ± 0.011
AlexNet	original	0.815 ± 0.023	0.681 ± 0.085	0.789 ± 0.022	0.757 ± 0.048
MobileNetv3	original	0.848 ± 0.007	0.785 ± 0.017	0.811 ± 0.022	0.831 ± 0.019
ResNet-50	original	0.938 ± 0.001	0.785 ± 0.017	**0.956** ± **0.044**	0.855 ± 0.011
AlexNet	V1	0.851 ± 0.078	0.724 ± 0.034	0.833 ± 0.109	0.782 ± 0.013
MobileNetv3	V1	0.900 ± 0.039	0.810 ± 0.034	0.889 ± 0.044	0.833 ± 0.036
ResNet-50	V1	0.940 ± 0.002	**0.836** ± **0.017**	0.922 ± 0.022	0.878 ± 0.006
AlexNet	V2	0.897 ± 0.068	0.724 ± 0.068	0.889 ± 0.087	0.806 ± 0.003
MobileNetv3	V2	0.859 ± 0.109	0.750 ± 0.017	0.845 ± 0.131	0.787 ± 0.069
ResNet-50	V2	0.832 ± 0.010	0.724 ± 0.034	0.811 ± 0.022	0.774 ± 0.015
AlexNet	V3	0.797 ± 0.014	0.802 ± 0.017	0.745 ± 0.022	0.786 ± 0.007
MobileNetv3	V3	**0.951** ± **0.019**	**0.836** ± **0.017**	0.945 ± 0.022	**0.890** ± **0.018**
ResNet-50	V3	0.940 ± 0.002	**0.836** ± **0.017**	0.922 ± 0.022	0.878 ± 0.006

and its V1 and V3 versions, which achieved top 1 scores for 2 out of the 3 classes.

Finally, we measured the weighted accuracies (ACC_w) of the networks on the test set. As it can be seen in Table 9, we can once again confirm that the usage of the hand-crafted features led to significant improvements. The networks that only used the hand-crafted features performed significantly worse yet again with the exception of the MLP variant which performed almost comparably to the AlexNet original network. The results of the LightCNN architectures was once again really uniform but they were outperformed by both the majority of the original networks and the ones that used both the hand-crafted features and the input images. The best performing networks were the MobileNetv3 (V3) and ResNet-50 (V1 and V3) architectures, which corresponds to the previous tables (Table 7 and Table 8) and other metrics as well.

4. Conclusions

In this work, we explored how diabetic retinopathy and diabetic macular edema can be detected on fundus images. We revisited the problem of combining the hand-crafted features with those extracted by neural networks. We built upon our previous work [8] to show that several architectures can be used to extract the CNN features and denoted this version as V1. We also noted that choosing the optimal

Table 8
A summary of the class-level accuracies measured on the DME dataset.

Network	*Type*	DME0	DME1	DME2
LightCNN [38]	SVM	0.806 ± 0.001	0.816 ± 0.019	0.777 ± 0.057
LightCNN [38]	RF	0.811 ± 0.010	0.806 ± 0.001	0.772 ± 0.086
LightCNN [38]	MLP	0.801 ± 0.010	0.816 ± 0.019	0.782 ± 0.067
LightCNN [38]	DT	0.816 ± 0.038	0.825 ± 0.001	0.786 ± 0.057
Hand-crafted	Softmax	0.553 ± 0.001	0.903 ± 0.001	0.563 ± 0.019
Hand-crafted	SVM	0.675 ± 0.085	0.903 ± 0.001	0.675 ± 0.085
Hand-crafted	MLP	0.782 ± 0.010	0.893 ± 0.001	0.752 ± 0.010
AlexNet	original	0.753 ± 0.028	0.859 ± 0.047	0.801 ± 0.029
MobileNetv3	original	0.816 ± 0.019	0.879 ± 0.009	**0.869** ± **0.010**
ResNet-50	original	0.850 ± 0.009	0.874 ± 0.038	0.840 ± 0.047
AlexNet	V1	0.772 ± 0.028	0.893 ± 0.020	0.762 ± 0.047
MobileNetv3	V1	0.826 ± 0.038	0.893 ± 0.020	0.840 ± 0.047
ResNet-50	V1	0.869 ± 0.010	**0.913** ± **0.019**	**0.869** ± **0.010**
AlexNet	V2	0.801 ± 0.010	0.908 ± 0.010	0.797 ± 0.038
MobileNetv3	V2	0.777 ± 0.076	0.893 ± 0.020	0.782 ± 0.105
ResNet-50	V2	0.762 ± 0.010	0.884 ± 0.019	0.748 ± 0.038
AlexNet	V3	0.762 ± 0.010	0.898 ± 0.029	0.719 ± 0.038
MobileNetv3	V3	**0.884** ± **0.019**	0.888 ± 0.010	0.864 ± 0.037
ResNet-50	V3	0.869 ± 0.010	**0.913** ± **0.019**	**0.869** ± **0.010**

Table 9
The weighted accuracies measured on the DME dataset.

Network	*Type*	ACC_w
LightCNN [38]	SVM	0.793 ± 0.029
LightCNN [38]	RF	0.792 ± 0.044
LightCNN [38]	MLP	0.793 ± 0.029
LightCNN [38]	DT	0.803 ± 0.043
Hand-crafted	Softmax	0.592 ± 0.009
Hand-crafted	SVM	0.697 ± 0.077
Hand-crafted	MLP	0.779 ± 0.009
AlexNet	original	0.783 ± 0.026
MobileNetv3	original	0.847 ± 0.012
ResNet-50	original	0.852 ± 0.022
AlexNet	V1	0.779 ± 0.037
MobileNetv3	V1	0.837 ± 0.043
ResNet-50	V1	0.874 ± 0.007
AlexNet	V2	0.809 ± 0.024
MobileNetv3	V2	0.789 ± 0.082
ResNet-50	V2	0.764 ± 0.018
AlexNet	V3	0.754 ± 0.028
MobileNetv3	V3	0.875 ± 0.011
ResNet-50	V3	0.874 ± 0.007

architecture is challenging and can greatly affect the final results. Then, we proposed two more methods, which increase the computational capabilities of V1. In V2, we separated the computational graph into two distinct paths: one that processes only the hand-crafted features and one that processes those extracted by the neural network. We noted that this separation may not always be optimal, as it totally isolates these features. To overcome this issue, we proposed another method, V3, where both kinds of features are processed at the same time, layer by layer, resulting in the detection of more complex patterns.

We have shown that each of the proposed versions has advantages and disadvantages, which makes choosing the best architecture a really difficult task. We measured the performance of all of these algorithms, as well as their original networks on our test set, and also compared these results to several baselines that only used the hand-crafted features as well as different architectures proposed in [38] and used a variety of metrics to make the evaluation more precise and fair. We have shown that all of the algorithms that used the hand-crafted features achieved substantially better results than the networks that did not use them and outperformed the aforementioned baselines and architectures as well. Then, we concluded that while the overall performance of the different versions of the proposed algorithm were close to each other, for our problem the V3 version of MobileNetv3 and the V1 version of ResNet 50 architectures performed the best. We made the hand-crafted features used in this paper openly available in FigShare at [39] and the code at [40].

CRediT authorship contribution statement

Gergo Bogacsovics: Conceptualization, Methodology, Investigation, Writing - original draft, Writing - review & editing, Software, Validation. **Janos Toth:** Methodology, Investigation, Writing - original draft, Writing - review & editing, Validation. **Andras Hajdu:** Conceptualization, Writing - review & editing, Validation. **Balazs Harangi:** Conceptualization, Methodology, Investigation, Writing - original draft, Writing - review & editing, Validation.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research was supported by the ÚNKP-21–3-I-DE-99 and the ÚNKP-21-5-DE-485 New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund. Moreover, the research was supported in part by the Janos Bolyai Research Scholarship of the Hungarian Academy of Sciences. This paper was supported also by the FIKP-20428–3/2018/FEKUTSTRAT project of the University of Debrecen.

References

[1] World Health Organization, Global report on diabetes, World Health Organization, 2016.
[2] A.D. Fleming, S. Philip, K.A. Goatman, G.J. Prescott, P.F. Sharp, J.A. Olson, The evidence for automated grading in diabetic retinopathy screening, Curr. Diabetes Rev. 7 (4) (2011) 246–252.
[3] J. Shan, L. Li, A deep learning method for microaneurysm detection in fundus images, in: Connected Health: Applications, Systems and Engineering Technologies (CHASE) 2016 IEEE First International Conference on, IEEE, 2016, pp. 357–358.
[4] J.H. Tan, H. Fujita, S. Sivaprasad, S.V. Bhandary, A.K. Rao, K.C. Chua, U. R. Acharya, Automated segmentation of exudates, haemorrhages, microaneurysms using single convolutional neural network, Inf. Sci. 420 (2017) 66–76.
[5] T. Walter, P. Massin, A. Erginay, R. Ordonez, C. Jeulin, J.-C. Klein, Automatic detection of microaneurysms in color fundus images, Med. Image Anal. 11 (6) (2007) 555–566.
[6] G. Quellec, M. Lamard, P.M. Josselin, G. Cazuguel, B. Cochener, C. Roux, Optimal wavelet transform for the detection of microaneurysms in retina photographs, IEEE Trans. Med. Imaging 27 (9) (2008) 1230–1241.
[7] C. Agurto, V. Murray, E. Barriga, S. Murillo, M. Pattichis, H. Davis, S. Russell, M. Abramoff, P. Soliz, Multiscale AM-FM methods for diabetic retinopathy lesion detection, IEEE Trans. Med. Imaging 29 (Feb 2010) 502–512.
[8] B. Harangi, J. Toth, A. Baran, A. Hajdu, Automatic screening of fundus images using a combination of convolutional neural network and hand-crafted features, in: 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), IEEE, 2019, pp. 2699–2702.
[9] T. Zhang, Y. Zeng, B. Xu, Hcnn: A neural network model for combining local and global features towards human-like classification, Int. J. Pattern Recognit Artif Intell. 30 (01) (2016) 1655004.
[10] X. Zhou, Y. Li, W. Liang, Cnn-rnn based intelligent recommendation for online medical pre-diagnosis support, IEEE/ACM Trans. Comput. Biol. Bioinf. (2020).
[11] M. Polsinelli, L. Cinque, G. Placidi, A light cnn for detecting covid-19 from ct scans of the chest, Pattern Recogn. Lett. 140 (2020) 95–100.
[12] S. Gehlot, A. Gupta, R. Gupta, Sdct-auxnetθ: Dct augmented stain deconvolutional cnn with auxiliary classifier for cancer diagnosis, Med. Image Anal. 61 (2020), 101661.
[13] N. Liu, M. Rogers, H. Cui, W. Liu, X. Li, P. Delmas, Deep convolutional neural networks for regular texture recognition, PeerJ Comput. Sci. 8 (2022), e869.
[14] N. Baker, H. Lu, G. Erlikhman, P.J. Kellman, Local features and global shape information in object classification by deep convolutional neural networks, Vision Res. 172 (2020) 46–61.
[15] P. Porwal, S. Pachade, R. Kamble, M. Kokare, G. Deshmukh, V. Sahasrabuddhe, F. Meriaudeau, Indian diabetic retinopathy image dataset (idrid): A database for diabetic retinopathy screening research, Data 3(3) 2018.
[16] Kaggle Inc, Diabetic Retinopathy Detection, Accessed: 2021-08-29.
[17] E. Decencière, X. Zhang, G. Cazuguel, B. Lay, B. Cochener, C. Trone, P. Gain, R. Ordonez, P. Massin, A. Erginay, B. Charton, J.-C. Klein, Feedback on a publicly distributed database: the messidor database, Image Anal. Stereol. 33 (Aug 2014) 231–234.
[18] J.P. Havlicek, AM-FM image models. PhD thesis, The University of Texas at Austin, 1996.
[19] B. Antal, A. Hajdu, Improving microaneurysm detection using an optimally selected subset of candidate extractors and preprocessing methods, Pattern Recogn. 45 (1) (2012) 264–270.
[20] K. Zuiderveld, Contrast limited adaptive histogram equalization, in Graphics Gems IV (P.S. Heckbert, ed.), pp. 474–485, Academic Press Professional, Inc., 1994.
[21] A. Youssif, A. Ghalwash, A. Ghoneim, Comparative study of contrast enhancement and illumination equalization methods for retinal vasculature segmentation, in: Cairo International Biomedical Engineering Conference, 2006, pp. 1–5.
[22] A. Criminisi, P. Perez, K. Toyama, Object removal by exemplar-based inpainting, IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2 (2003) 721–728.
[23] T. Walter, J.-C. Klein, Automatic detection of microaneurysms in color fundus images of the human retina by means of the bounding box closing, in: in Medical Data Analysis: Third International Symposium (ISMDA), 2002, pp. 210–220.
[24] I. Lazar, A. Hajdu, Retinal microaneurysm detection through local rotating cross-section profile analysis, IEEE Trans. Med. Imaging 32 (2) (2013) 400–407.
[25] T. Walter, P. Massin, A. Erginay, R. Ordonez, C. Jeulin, J.-C. Klein, Automatic detection of microaneurysms in color fundus images, Med. Image Anal. 11 (6) (2007) 555–566.
[26] B. Zhang, X. Wu, J. You, Q. Li, F. Karray, Detection of microaneurysms using multi-scale correlation coefficients, Pattern Recogn. 43 (6) (2010) 2237–2248.
[27] B. Nagy, B. Harangi, B. Antal, A. Hajdu, Ensemble-based exudate detection in color fundus images, in: Symposium on Image and Signal Processing and Analysis, 2011, pp. 700–703.
[28] G.D. Finlayson, B. Schiele, J.L. Crowley, Comprehensive colour image normalization, in Computer Vision — ECCV'98 (H. Burkhardt and B. Neumann, eds.), (Berlin, Heidelberg), pp. 475–490, Springer, Berlin Heidelberg, 1998.
[29] P. Soille, Morphological Image Analysis: Principles and Applications, Springer-Verlag, 2004.
[30] A. Sopharak, B. Uyyanonvara, S. Barman, T.H. Williamson, Automatic detection of diabetic retinopathy exudates from non-dilated retinal images using mathematical morphology methods, Comp. Med. Im. Grap. 32 (8) (2008) 720–727.
[31] D. Welfer, J. Scharcanski, D. Marinho, A coarse-to-fine strategy for automatically detecting exudates in color eye fundus images, Comput. Med. Imaging Graph. 34 (3) (2010) 228–235.
[32] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Commun. ACM 60 (6) (2017) 84–90.
[33] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, et al., Searching for mobilenetv3, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 1314–1324.
[34] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[35] American academy of ophthalmology. international clinical diabetic retinopathy disease severity scale, detailed table.http://www.icoph.org/dynamic/attachments/resources/diabetic-retinopathy-detail.pdf. Accessed: Oct 14, 2016.
[36] H. Robbins, S. Monro, A stochastic approximation method, Ann. Math. Stat., pp. 400–407, 1951.
[37] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980, 2014.
[38] S. Gayathri, V.P. Gopi, P. Palanisamy, A lightweight cnn for diabetic retinopathy classification from fundus images, Biomed. Signal Process. Control 62 (2020), 102115.
[39] G. Bogacsovics, B. Harangi, J. Toth, A. Hajdu, Handcrafted features for fundus image classification. doi: 10.6084/m9.figshare.16543107.v2 Accessed: 2021-08-31.
[40] G. Bogacsovics, B. Harangi, J. Toth, A. Hajdu, Fundus Image Classification software,https://github.com/gergobogacsovics/FundusImageClassification Accessed: 2021-08-31.