International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Convolutional Neural Networks for Diabetic Retinopathy Detection in Retinal Images

Bhaskar Marapelli*[1], D. Baburao[2], Vedavyas Gurla[3], Samatha Konda[4], Ch.Anil Carie[5], Gandla Shivakanth[6]

Submitted: 27/01/2024 **Revised**: 05/03/2024 **Accepted**: 13/03/2024

Abstract: Diabetic retinopathy is a degenerative eye disease associated with uncontrolled diabetes, high blood pressure, blood sugar, cholesterol, and body weight. If diabetes-related diabetic retinopathy is not detected and treated in its early stages, it might result in visual impairment. Traditional methods of diagnosis often require manual examination by trained ophthalmologists, which can be time-consuming and subjective. Deep learning models called Convolutional Neural Networks (CNNs) are renowned for their capacity to recognize small patterns and features in images, which makes them very useful for challenging medical imaging applications. In this study, we proposed two CNN architectures, namely CNN-Plain and CNN-BN-D to automate and enhance the diagnostic process for diabetic retinopathy (DR) detection. The model's effectiveness is assessed using a range of criteria, such as accuracy, precision, recall, and F1Score, to confirm that it is as effective as current diagnostic techniques. The results demonstrate the CNN-BN-D model with 0.94 accuracy on train and test data exhibits superior performance and generalization compared to the simpler CNN-Plain architecture with 0.89 accuracy in the task of diabetic retinopathy detection.

Keywords: *Diabetic retinopathy, Ocular ailment, Uncontrolled diabetes, Elevated blood, sugar, Convolutional Neural Networks (CNN), Deep learning models, Medical imaging*

1. Introduction

Diabetes frequently results in diabetic retinopathy (DR), a common consequence that, if not promptly detected and treated in its early stages, can cause vision impairment and blindness. This presents a serious public health concern. The current diagnostic landscape relies heavily on manual examination by trained ophthalmologists, a process that is both time-consuming and inherently subjective. As the prevalence of diabetes continues to rise globally, there is an urgent need for more efficient and objective methods of DR detection.

The degenerative eye condition known as diabetic retinopathy is linked to uncontrolled diabetes, high blood pressure, cholesterol, and sugar and blood sugar levels, as well as weight gain. In its early stages, called non-proliferative retinopathy, damage to the retina's blood vessels may lead to swelling and leakage, causing macular edema [1]. While vision is usually unaffected at this stage, close monitoring is essential to prevent progression to the more severe proliferative stage.

Proliferative diabetic retinopathy is characterized by neovascularization, where abnormal blood vessels grow in response to the retina's deprived oxygen and nutrients [2]. Due to their fragility, these veins run the risk of leaking blood into the vitreous, which could lead to visual distortions like floaters or poor vision. Complications can include scar tissue formation, leading to retinal wrinkling or detachment. Additionally, abnormal blood vessel growth may block the eye's drainage angle, resulting in increased eye pressure and neovascular angle-closure glaucoma. Timely intervention and regular monitoring are crucial to prevent further vision loss.

Enhancements in computer-aided design and analysis have been achieved through the integration of advanced machine-learning algorithms and processing techniques, aiming to improve the accuracy of detection. One notable machine learning approach employed for diverse applications in image processing is the CNN [3]. The CNN, known for its effectiveness in recognizing patterns and features in visual data, has proven to be a valuable tool in various domains, contributing to the refinement of automated detection systems. By leveraging the capabilities of CNN and other sophisticated techniques, the field of computer-aided design

[1] *Associate Professor, Department of Computer Science and Information Technology, Koneru Lakshmaiah Education Foundation, Vaddeswaram 522502, Andhra Pradesh, India. Email: bhaskarmarapelli@gmail.com, ORCID ID: https://orcid.org/0000-0002-0101-9083.*
[2] *Associate Professor, Computer Science and Engineering, Vidya Jyothi Institute of Technology (VJIT), Aziz Nagar Gate, C.B. Post, Hyderabad–500 075, Telangana, India, Email: dbaburao@hotmail.com.*
[3] *Associate Professor,Dept. of CSE-AIML, Geethanjali College of Engineering and Technology (An Autonomous Institution & Affiliated to JNTUH), Cheeryal(V), Keesara(M), Medchal District-501301. Telangana, India. Email: ved.gurla@gmail.com .*
[4] *Assistant Professor,Dept. of IT,Muffakam Jah Engineering College, (Sultan -UL- Uloom Education Society),Hyderabad, Telangana, India, Email: samatha.phd22@gmail.com.*
[5] *Assistant Professor, Dept of Computer Science and Engineering, SRM University, Amaravathi, Andhra Pradesh, India. Email: carieanil@gmail.com.*
[6] *Associate professor, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Hyderabad-500075, Telangana, India, Email: shvkanth0@gmail.com.*
** Corresponding Author Email: bhaskarmarapelli@gmail.com*

and analysis continues to evolve, offering more robust and precise solutions for detecting various phenomena and patterns within digital imagery [4].

Overfitting is a common issue with CNN models in deep learning, where the model performs remarkably well on the training data but finds it difficult to generalize to new, unseen data. Regularization strategies are essential for resolving this problem and enhancing the model's overall performance and generalizability [5]. Regularization in traditional machine learning typically involves adding penalty terms to the coefficients of features to control model complexity and prevent overfitting. However, regularization is applied to the weight matrices of neural network nodes in deep learning. By doing this, the scope of potential solutions is constrained, overfitting is deterred, and generalization performance is enhanced. Different regularization strategies impose different types of restrictions on the weight matrices [5].

The chosen methodology for this research involves initially preparing a base model for CNN training. Subsequently, experimentation with various regularization techniques is conducted, followed by performance evaluation and fine-tuning of the model.

2. Related Work

Untreated diabetic retinopathy can result in vascular bleeding, retinal detachment, and blindness from glaucoma. It is caused by high blood sugar levels destroying retinal blood vessels. Diagnosis in crucial areas is hindered by a lack of expertise and technology [6]. R Vignesh et., al. Utilized retinal images from the eyepacs diabetic retinopathy database, achieved accuracy above 90%, providing a more efficient and cost-effective alternative to manual tests. In order to solve the difficult and time-consuming manual identification process carried out by specialists, the research [7] investigates the use of a CNN for the detection of diabetic retinopathy in retinal pictures. The CNN algorithm underwent training using a dataset comprising over 1000 fundus images, followed by testing on an additional set of more than 50 images. The achieved accuracy of the CNN model was approximately 93.8%. The paper [8] introduces a specialized deep learning approach, the Customized Convolutional Neural Network (CCNN), designed for detecting Diabetic Retinopathy (DR) in individuals with diabetes through the analysis of fundus images. The CCNN model proposed attains an impressive test accuracy of 97.24% on the MESSIDOR Dataset, surpassing the performance of existing algorithms.

Neural networks have found success in diverse applications; however, the concern of overfitting arises due to the substantial number of parameters involved. To mitigate overfitting, regularization is a widely adopted strategy, involving the imposition of penalties on neural network weights. The paper [9] endeavors to present a comprehensive framework for neural networks incorporating regularization, establishing its consistency through theoretical proofs. The method of sieves and insights from minimal neural network theory is applied to address the challenge of parameter unidentifiability. To improve neural networks' verified robustness against adversarial cases, the research [10] presents the Misclassification Aware Adversarial Regularization (MAAR) defense technique. Unlike existing provable defense methods that treat all examples equally during training, the discrepancy in certified robustness between cases that are correctly and incorrectly identified is addressed by MAAR. By introducing a novel consistency regularization term, MAAR significantly improves the certified robustness of a network by ensuring consistency in the constraint of misclassified and correctly classified examples. Results from experiments on the CIFAR-10 and MNIST datasets show that, while keeping comparable accuracy, when it comes to proven robustness, MAAR outperforms several other cutting-edge algorithms.

3. Methodology

The proposed methodology outlines a systematic approach for developing a CNN model. It begins with defining the CNN architecture, specifying layers, activation functions, and relevant parameters, tailored to the task or problem domain. Data preprocessing follows, involving proper formatting and division into training, validation, and test sets, with input data normalization or standardization. Baseline model training without regularization establishes initial performance, evaluated on the validation set to detect potential overfitting. Subsequent steps involve experimenting with various regularization techniques, such as dropout and L1/L2 regularization, each assessed separately with different strengths. Performance evaluation compares models using metrics like accuracy, precision, and recall, considering the impact of regularization on generalization to unseen data. Fine-tuning incorporates insights from experimentation, adjusting chosen regularization techniques and other hyperparameters. Validation on the validation set ensures generalization and testing on an unseen dataset assesses the final model's overall performance.

3.1. Convolutional Neural Network (CNN)

A CNN is structured with multiple hidden layers designed to effectively extract meaningful information from images [11]. Figure 1 describes the common architecture of the CNN model. The four pivotal layers within a CNN are crucial for its image-processing capabilities. The Convolution layer plays a key role in applying filters to input images, capturing essential features and patterns. The fundamental component of CNN feature extraction is this convolution process, where filters are trained to identify unique patterns or features in the input data [12].

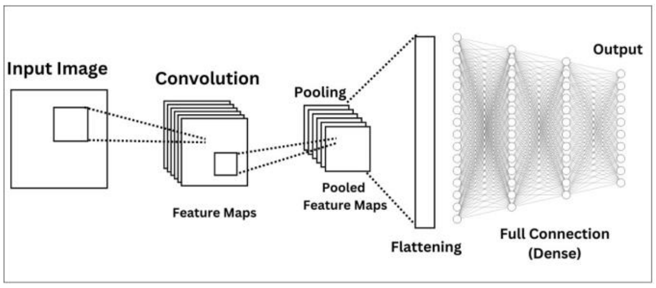

Figure 1: CNN architecture

For a two-dimensional convolution, commonly used in image processing in CNNs, the operation is extended to a double summation [13]:

$$Y[i,j] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} X[i-m, j-n].H[m,n]$$

Here:

H [m, n] signifies the filter value at the position (m, n).

The input value at location (i-m, j-n) is indicated by X [i-m, j-n].

The value of the output feature map at the point (i, j) is represented by Y [i, j].

The summations are carried out over the dimensions of the filters (M) and (N).

The Rectified Linear Unit (ReLU) layer enhances the network's non-linearity and helps the model understand complex correlations in the data by selectively activating neurons [14].

It is defined mathematically as:

$$f(x) = max(0, x)$$

In this instance, f(x) represents the output of the ReLU activation function given an input of x. The function returns the maximum of zero and the input value x. After that, the Pooling layer helps to minimize the input data's spatial dimensions, preserving important information while lowering the computational burden [15].

Mathematically, if we denote the input matrix as (X) and the pooled output matrix as (Y), the max pooling operation is expressed as:

$$Y[i,j] = max(X[2i,2j], X[2i,2j+1], X[2i+1,2j], X[2i+1,2j+1])$$

In this case, the output value at position (i,j) in the pooled matrix is $Y[i,j]$, and the maximum value within the 2x2 pooling window centered at $(2i,2j)$ is computed using the max function [16].

Lastly, a complete comprehension and classification of the input image are made possible by the connections between each neuron in the Fully Connected layer and every other neuron in the network [17].

The output vector $Y$of the Fully Connected (FC) layer is given by:

$$Y = X.W + b$$

Here:

The input vector from the layer above is represented by X

W comprises the weights that link every neuron in the input layer to the output layer, W weight matrix.

The bias vector b is added elementwise to the product of X and W.

Collectively, these layers contribute to the hierarchical feature extraction process that makes CNNs particularly effective in image recognition tasks. Figure 1 describes the graphical representation of CNN architecture.

3.2. Regularization techniques

Regularization strategies are essential for reducing overfitting and enhancing CNNs' capacity for generalization [18,5]. Several key regularization techniques are commonly employed in CNNs to enhance their performance and prevent overfitting. A popular method called "dropout" involves arbitrarily excluding some neurons during training to reduce the network's dependence on neurons and encourage the learning of more robust features [19].

During each training iteration, each neuron's output is multiplied by a binary mask m_i mi drawn from a Bernoulli distribution with probability p:

$$\text{dropout operation}: y_i = m_i . x_i$$

Here:

x_i is the original output of the neuron.

A binary mask with probability p and 0 with probability $1 - \text{p}$ is m_i.

y_i is the final output after applying dropout.

During the inference or testing phase, when all neurons are used, the outputs are scaled by p to ensure the expected value remains the same as during training:

$$\text{inference operation}: \hat{y} = \text{p} . x_i$$

The purpose of dropout is to prevent co-adaptation of neurons and encourage the network to learn more robust features by reducing reliance on specific neurons.

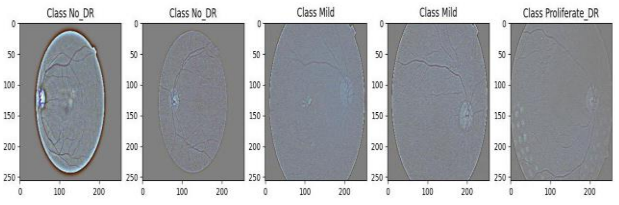

Figure 2: Sample images from dataset

Batch Normalization, another technique, normalizes layer inputs within mini-batches, mitigating internal covariate shifts and stabilizing the training process. This normalization reduces sensitivity to initial weights and hyperparameters.

The normalized input Y is then given by:

$$Y = {X - \mu}\Big/{\sqrt{\sigma^2 + \varepsilon}}$$

Here:

- The layer's input is X
- The mean of X over the mini-batch is represented by μ.
- The standard deviation of X over the mini-batch is represented by σ.
- A little constant added for numerical stability is ε

Next, a learnable parameter σ is used to scale the normalized input γ, and another learnable parameter β is used to shift it:

$BatchNorm(X) = \gamma . Y + \beta$

Here:

- Scaling parameter γ
- Shifting parameter β

During training, the mean and standard deviation are computed for each mini-batch, and the parameters γ and β are updated through backpropagation. Moving averages from the training phase are often used to calculate the mean and standard deviation during the inference step.

Applying random changes, including flips and rotations, to training data improves the model's ability to generalize to many input variations and diversifies the dataset. This process is known as data augmentation. A random transformation function T to input sample X to obtain augmented sample X':

$X' = T(X)$

Here, T represents a transformation function.

The common transformations include rotations, flips, zooms, and changes in brightness or contrast. These regularization techniques collectively contribute to creating more robust and effective CNN models.

4. Experimental setup

To evaluate the two models' performance, tests were conducted.

4.1. Data set

The data sets used consist of Gaussian-filtered retina scan images used for diabetic retinopathy detection. The main dataset originates from the APTOS 2019 Blindness Detection project, which focuses on identifying diabetic retinopathy using retinal images. This dataset can be found on Kaggle. Five classes of images represent different levels of severity of diabetic retinopathy, ranging from the absence of the condition (No_DR) to various stages of severity, including Mild, Moderate, Severe, and Proliferate$_{DR}$. Figure 2 shows sample images from the data set. we perform binary classification to detect whether Diabetic retinopathy is present or not present.

Figure 4, Table 1 provides the distribution of diabetic retinopathy levels of severity in the dataset based on the count of images for each severity level. The severity levels are typically assigned based on the progression of diabetic retinopathy, a medical condition affecting the retina due to diabetes.

Levels of Severity of Diabetic Retinopathy	No_DR	Moderate	Mild	Proliferate_DR	*Severe*
Count	*1805*	*999*	*370*	*295*	*193*

Table 1: Levels of Severity of Diabetic Retinopathy images count

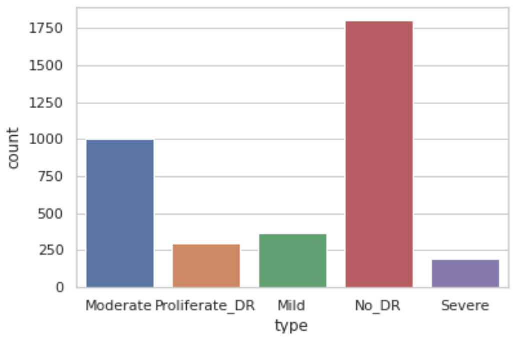

Figure 4: Diabetic retinopathy levels of severity image count

4.2. CNN-Plain Model

A CNN model is built in this experimental setup with the TensorFlow and Keras packages. The definition of the model architecture is a layer-by-layer stack. Convolutional layer (Conv2D) with 8 filters of size (3,3) is the initial layer, which uses the ReLU activation function. A max-pooling layer (MaxPooling2D) that downsamples the spatial dimensions comes next, with a pool size of (2,2). The model also includes extra convolutional layers with max-pooling in between. There are 16 filters in the second convolutional layer, and 32 filters with different kernel sizes in the third. To transform the 3D tensor into a 1D tensor, a flattening layer (named Flatten) is inserted after the convolutional layers. Then, two fully connected dense layers (Dense) are added: one with two units and softmax activation for binary classification and the other with 32 units and ReLU activation. The model is put together using the Adam optimizer, binary cross-entropy loss function, and accuracy as the evaluation metric. Using the supplied training batches, the training is carried out for 30 epochs, and the validation is done on the validation batches.

4.3. CNN-BN-D Model

In this altered experimental setup, the CNN model undergoes modifications by adding certain layers for comparison purposes. The model architecture remains a sequential stack of layers implemented using TensorFlow and Keras. The first layer is a convolutional layer (Conv2D) with eight filters of size (3,3) and employs the ReLU activation function. A second max-pooling layer (MaxPooling2D) with a pool size of (2,2) is employed for downsampling.

The architectural changes involve the addition of batch normalization layers (BatchNormalization) after each convolutional layer. To reduce overfitting, a dropout layer (Dropout) with a dropout rate of 0.15 is introduced after the first dense layer. The purpose of adding batch normalization and dropout layers to this modified CNN architecture is to see how these regularization approaches affect the model's performance in training and validation.

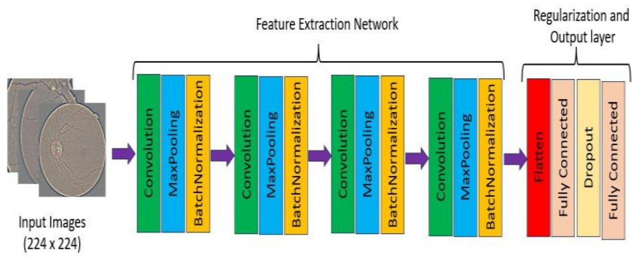

Figure 3: CNN-BN-D Model

Figure 3: shows the architecture of the CNN-BN-D Model.

4.4. Training process

The model architecture is chosen, and then it is assembled using the 1e-5 learning rate Adam optimizer. Using the binary cross-entropy loss function, accuracy is monitored as a statistic throughout the training phase. The Gaussian-filtered retina scan pictures used for the diabetic retinopathy detection dataset are fed into the model for 30 iterations of training. Throughout this training phase, the model's internal parameters are dynamically adjusted through the optimization algorithm and computed loss, enhancing its proficiency in binary outcome classification. The training data is presented iteratively in batches to the model, and parameter updates are performed to minimize the loss.

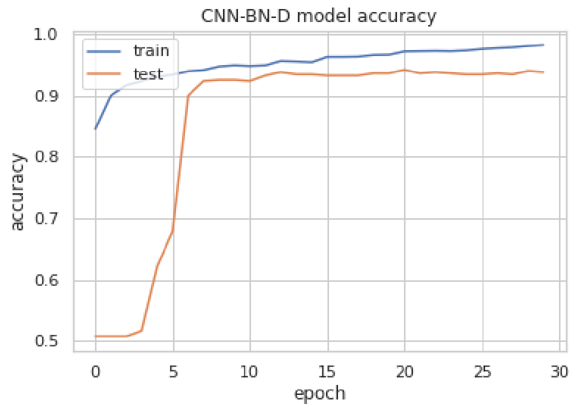

Figure 5 (a). CNN-BN-D Accuracy

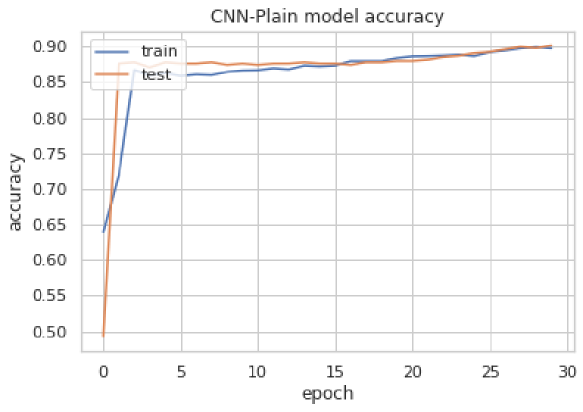

Figure 5 (b). CNN-Plain Accuracy

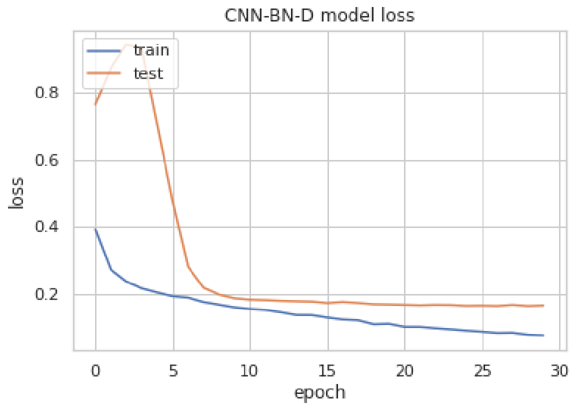

Figure 5 (c). CNN-BN-D Loss

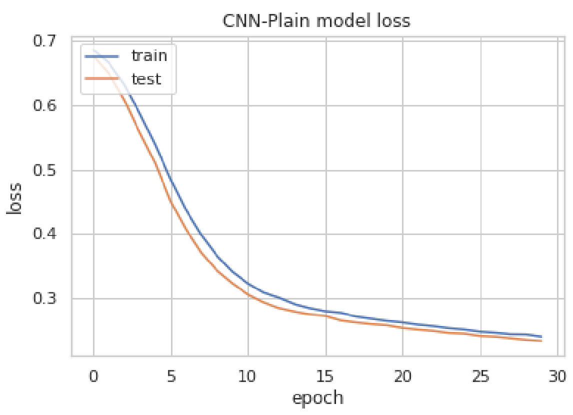

Figure 5 (d). CNN-Plain Loss

Figure 5 Comparison of Accuracy and Loss for CNN-BN-D and CNN-Plain Models.

Table 2: summary of the model's performance

Metric	Model	Class	
		DR	No_DR
Precision	CNN-Plain	0.88	0.9
	CNN-BN-D	0.95	0.93
Recall	CNN-Plain	0.91	0.87
	CNN-BN-D	0.93	0.94
F1-score	CNN-Plain	0.9	0.89
	CNN-BN-D	0.94	0.94
Support	CNN-Plain	279	271
	CNN-BN-D	279	271
Train accuracy	CNN-Plain	0.89	
	CNN-BN-D	0.94	
Test accuracy	CNN-Plain	0.89	
	CNN-BN-D	0.94	

5. Results and Discussion

Table 2 provides a detailed summary of the performance metrics for two models, CNN-Plain (Convolutional Neural Network without Batch Normalization and Dropout) and CNN-BN-D (Convolutional Neural Network with Batch Normalization and Dropout), across two classes: 'DR' (Diabetic Retinopathy) and ' No_DR ' (No Diabetic Retinopathy). The performance metrics measure highlights that CNN-BN-D performs better than CNN-Plain across precision, recall, $F1-Score$, and accuracy for both classes.

5.1. Performance Metrics

1. **Precision:** Precision assesses how well the model predicts favorable outcomes.

$$P_{\mathrm{DR}} = \frac{TP_{DR}}{TP_{DR} + FP_{DR}}$$

$$P_{\mathrm{No\_DR}} = \frac{TP_{No\_DR}}{TP_{No\_DR} + FP_{No\_DR}}$$

- TP_{DR} (True Positives for 'DR'): The number of cases that the model accurately identified as ' DR'
- FP_{DR} (False Positives for 'DR'): The number of cases where the model misclassified some occurrences as 'DR' when they actually belong to ' No_DR'.
- TP_{No_DR} (True Positives for ' No_DR '): The number of cases that the model accurately categorized as 'No_DR'.
- FP_{No_DR} (False Positives for ' No_DR '): The number of cases when the model misclassified some occurrences as ' No_DR when they actually fall under ' DR '.

For ' DR,' CNN-BN-D outperforms CNN-Plain with a precision of 0.95 compared to 0.88. For ' No_DR,' CNN-Plain has a slightly higher precision of 0.9 compared to CNN-BN-D's 0.93.

2. **Recall:** Also referred to as sensitivity, measures the model's ability to accept each positive example.

$$R_{\mathrm{DR}} = \frac{TP_{DR}}{TP_{DR} + FN_{DR}}$$

$$R_{\mathrm{No\_DR}} = \frac{TP_{No\_DR}}{TP_{No\_DR} + FN_{No\_DR}}$$

- TP_{DR} (True Positives for ' DR '): The number of cases that the model accurately categorized as ' No_DR '.
- FN_{DR} (False Negatives for ' DR '): The number of cases where the model misclassified some occurrences as ' DR ' when they actually belong to ' No_DR '.
- TP_{No_DR} (True Positives for ' No_DR '): The number of cases that the model accurately categorized as No_DR '.
- FN_{No_DR} (False Negatives for ' No_DR '): The number of cases when the model misclassified some occurrences as ' No_DR ' when they actually fall under ' DR '.

CNN-BN-D has higher recall values for both 'DR' (0.93) and 'No_DR' (0.94) compared to CNN-Plain (0.91 and 0.87, respectively).

3. **F1-Score:**

The F1-Score computes the harmonic mean of precision and recall achieving a balance.

$$\mathrm{F1}_{\mathrm{DR}} = \frac{P_{DR} \,.\, R_{DR}}{P_{DR} + R_{DR}}$$

$$\mathrm{F1}_{No\_DR} = \frac{P_{No\_DR} \,.\, R_{No\_DR}}{P_{No\_DR} + R_{No\_DR}}$$

- P_{DR} (Precision for 'DR')
- R_{DR} (Recall for 'DR')
- P_{No_DR} (Precision for 'No_DR')
- R_{No_DR} (Recall for 'No_DR')

CNN-BN-D shows improvements in F1-Score for both 'DR' (0.94) and 'No_DR' (0.94) compared to CNN-Plain (0.9 and 0.89, respectively).

4. **Support:** The number of actual instances of each class in the test dataset is represented by the term "support." Both models have the same support values for 'DR' (279) and 'No_DR' (271), indicating a balanced dataset.
5. **Train Accuracy:** Training accuracy assesses the model's accuracy specifically on the training dataset. CNN-BN-D achieves a higher train accuracy of 0.94 compared to CNN-Plain's 0.89.
6. **Test Accuracy:** Test accuracy measures how well the model works on untrained data, and CNN-BN-D performs better than CNN-Plain, scoring 0.94 against 0.89 for CNN-Plain.

The inclusion of Batch Normalization and Dropout in CNN-BN-D contributes to improved model performance [20].

From the accuracy curves shown in Figure 5(a) and Figure 5(b), it is evident that CNN-BN-D consistently outperforms CNN-Plain in terms of accuracy throughout the training epochs. CNN-BN-D has demonstrated smoother convergence and higher overall accuracy, which suggests that it has a superior ability to learn and generalize from the training set.

The loss depicted in Figure 5(c) and Figure 5(d) CNN-BN-D exhibits a more favorable reduction in loss compared to CNN-Plain. The steeper decline and lower final loss value for CNN-BN-D suggest that it converges more effectively during training, emphasizing its better optimization and generalization capabilities.

The presented graphs strongly support the superiority of the CNN-BN-D model over CNN-Plain in terms of both accuracy and loss. The integration of batch normalization and dropout in CNN-BN-D contributes to its enhanced performance, making it a robust and reliable model for the classification problem under consideration.

6. Conclusion

In this work, we assessed how well two CNN architectures CNN-Plain and CNN-BN-D performed in the task of detecting diabetic retinopathy (DR). The models were trained and tested on a dataset with multiple severity levels, including 'No_DR,' 'Mild,' 'Moderate,' 'Severe,' and 'Proliferate$_{\mathrm{DR}}$.' Our findings indicate that CNN-BN-D, which incorporates batch normalization and dropout layers, outperforms CNN-Plain across various evaluation metrics. CNN-BN-D shows improved F1-score, recall, and precision for both the 'DR' and 'No_DR' classes. Additionally, it achieves superior overall accuracy on the test set compared to CNN-Plain. The experiment results suggest that the regularization techniques, namely batch normalization and dropout, contribute to the robustness and generalization capability of the CNN model in the context of diabetic retinopathy detection. These methods improve the model's capacity to generalize to new data and help avoid overfitting. The CNN-BN-D model with 0.94 accuracy on train and test data exhibits superior performance and generalization compared to the simpler CNN-Plain architecture with 0.89 accuracy in the task of diabetic retinopathy detection.

Acknowledgements

We would like to express our sincere gratitude to all those who have contributed to the completion of this work. We thank our colleagues from KLEF deemed to be University who provided insight and expertise that greatly assisted the research.

Author contributions

Bhaskar Marapelli: Conceptualization, Methodology, Software, Writing-Original draft preparation **D. Baburao, Vedavyas Gurla[3]** Data curation, Software, Validation **Samatha Konda, Ch.Anil Carie, Gandla Shivakanth:** Visualization, Investigation, Writing-Reviewing and Editing.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] S. Rykov, D. Chugaiev, and S. Ziablitsev, “Blood selectin levels as a predictive fac- tor for diabetic retinopathy and diabetic macular edema in type 2 diabetes.” Journal of Ophthalmology (Ukraine)/Oftalmologicˇeskij Žurnal, no. 3, 2023.

[2] N. u. Huda, A. A. Salam, N. S. Alghamdi, J. Zeb, and M. U. Akram, “Prolifera- tive diabetic retinopathy diagnosis using varying-scales filter banks and double-layered thresholding,” Diagnostics, vol. 13, no. 13, p. 2231, 2023.

[3] A. Shanthini, G. Manogaran, G. Vadivu, K. Kottilingam,P. Nithyakani, and C. Fancy, “Threshold segmentation based multi-layer analysis for detecting diabetic retinopathy using convolution neural net- work,” Journal of Ambient Intelligence and Humanized Computing, pp. 1–15, 2021.

[4] S. Burewar, A. B. Gonde, and S. K. Vip- parthi, “Diabetic retinopathy detection by retinal segmentation with region merging using cnn,” in 2018 IEEE 13th International Conference on Industrial and Information Sys- tems (ICIIS). IEEE, 2018, pp. 136–142.

[5] P. H. N. Castro, G. C. Fortuna, R. A. B. de Queiroz, and G. J. P. Moreira, “Regu- larization through simultaneous learning: A case study for hop classification,” arXiv preprint arXiv:2305.13447, 2023.

[6] R. Vignesh, N. Muthukumaran, and M. P. Austin, “Detection of diabetic retinopathy image analysis using convolution graph neural network,” in 2023 International Con- ference on Inventive Computation Technolo- gies (ICICT). IEEE, 2023, pp. 921–929.

[7] K. Swarnalatha, U. A. Nayak, N. A. Benny,H. Bharath, D. Shetty, and S. D. Kumar, “Detection of diabetic retinopathy using convolution neural network,” in Emerging Research in Computing, Information, Com- munication and Applications: Proceedings of ERCICA 2022. Springer, 2022, pp. 427–439.

[8] D. Mane, S. Sangve, P. Kumbharkar, S. Rat- naparkhi, G. Upadhye, and S. Borde, “A diabetic retinopathy detection using cus- tomized convolutional neural network.”

[9] X. Shen and J. Lin, “Consistency of neu- ral networks with regularization,” arXiv preprint arXiv:2207.01538, 2022.

[10] M. Xu, T. Zhang, Z. Li, and D. Zhang, “Im- proving the certified robustness of neural networks via consistency regularization,” arXiv preprint arXiv:2012.13103, 2020.

[11] P. Sinha, I. Psaromiligkos, and Z. Zilic, “A structurally regularized cnn architec- ture via adaptive subband decomposition,” arXiv preprint arXiv:2306.16604, 2023.

[12] M. M. Farag, “Design and analysis of con- volutional neural layers: A signal process- ing perspective,” IEEE Access, vol. 11, pp. 27 641–27 661, 2023.

[13] G. Michelogiannakis, X. S. Li, D. H. Bailey, and J. Shalf, “Extending summation preci- sion for network reduction operations,” In- ternational Journal of Parallel Programming, vol. 43, pp. 1218–1243, 2015.

[14] C. Chen, F. Min, Y. Zhang, and H. Bao, “Relu-type hopfield neural network with analog hardware implementation,” Chaos, Solitons & Fractals, vol. 167, p. 113068, 2023.

[15] M. Monnet, H. Gebran, A. Matic-Flierl, F. Kiwit, B. Schachtner, A. Bentellis, and J. M. Lorenz, “Pooling techniques in hybrid quantum-classical convolu- tional neural networks,” arXiv preprint arXiv:2305.05603, 2023.

[16] S. M. Ko, S. Cho, D.-W. Jeong, S. Han, M. Lee, and H. Lee, “Grouping matrix based graph pooling with adaptive num- ber of clusters,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, no. 7, 2023, pp. 8334–8342.

[17] D. Sun, J. Wulff, E. B. Sudderth, H. Pfis- ter, and M. J. Black, “A fully-connected layered model of foreground and back- ground flow,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2013, pp. 2451–2458.

[18] C. Zhao, G. Dong, S. Zhang, Z. Tan, and A. Basu, “Frequency regularization: Restricting information redundancy of convolutional neural networks,” arXiv preprint arXiv:2304.07973, 2023.

[19] Z. Liu, Z. Xu, J. Jin, Z. Shen, and T. Darrell, “Dropout reduces underfitting,” arXiv preprint arXiv:2303.01500, 2023.

[20] Kim, Bum Jun, Hyeyeon Choi, Hyeonah Jang, Donggeon Lee, and Sang Woo Kim. "How to Use Dropout Correctly on Residual Networks with Batch Normalization." arXiv preprint arXiv:2302.06112 (2023).