American Academic Scientific Research Journal for Engineering, Technology, and Sciences
ISSN (Print) 2313-4410, ISSN (Online) 2313-4402
https://asrjetsjournal.org/index.php/American_Scientific_Journal/index

Design and Evaluation of a Convolutional Neural Network Model for Automated Detection of Diabetic Retinopathy Using Retinal Fundus Photographs

Tobi Titus Oyekanmi[a]*, Peter Oluwasayo Adigun[b], Ayodeji Adedotun Adeniyi[c]

[a,b]*Department of Computer Science, New Mexico Highlands University, 1005 Diamond St, Las Vegas, New Mexico, USA*

[c]*Department of Media, Art, and Technology, New Mexico Highlands University, 1005 Diamond St, Las Vegas, New Mexico, USA*

[a]*Email: toyekanmi@live.nmhu.edu*

[b]*Email: poadigun@nmhu.edu*

[c]*Email: aadeniyi1@live.nmhu.edu*

Abstract

Diabetic retinopathy (DR) is a leading cause of preventable blindness globally, necessitating timely and accurate screening methods. This study presents the design and evaluation of a custom convolutional neural network (CNN) model, LightDR, for automated classification of DR using retinal fundus photographs. The *Augmented_resized_V2* dataset, derived from the Eyepacs, Aptos, and Messidor collections from *Kaggle,* was used to train over 143,000 labeled images. The LightDR architecture was built using TensorFlow and optimized through data augmentation, class balancing, and performance-driven callbacks. Evaluation of the model yielded an accuracy of 84%, with precision and recall metrics indicating strong sensitivity to disease presence and reliable classification of healthy cases. The model demonstrated generalization and interpretability, supported by Grad-CAM visualizations and confusion matrix analysis. These findings suggest that LightDR offers a scalable and effective solution for DR screening, with potential for integration into clinical workflows pending further validation.

Keywords: Diabetic retinopathy; convolutional neural network; fundus image; image classification.

Received: 8/23/2025
Accepted: 10/23/2025
Published: 11/1/2025

---

* *Corresponding author.*

1.Introduction

Diabetic retinopathy (DR) is a leading cause of vision impairment among working-age adults worldwide[1].The estimated figure in the US was 9.60 million (26.43% of those with diabetes) had diabetic retinopathy, and 1.84 million people (5.06% of those with diabetes) had vision-threatening diabetic retinopathy (VTDR) in 2021[2] .Additionally, diabetic retinopathy (DR) is one of the most prevalent microvascular complications of diabetes mellitus (DM) and remains a leading cause of preventable blindness globally[3] .

According to the International Diabetes Federation (IDF, 2024), an estimated 590 million adults worldwide are living with diabetes, and this figure is projected to rise to 643 million by 2030[4, 5].Among these, nearly one-third are expected to develop some form of diabetic retinopathy during their lifetime. The progression of DR is gradual and often asymptomatic in its early stages, which underscores the critical importance of timely diagnosis and intervention to prevent irreversible vision loss.

On the other hand, retinal fundus photography has emerged as a reliable, non-invasive method for screening and diagnosing DR[6].However, manual assessment of fundus images by ophthalmologists and trained graders is time-consuming, resource-intensive, and subject to inter-observer variability. In many regions, particularly in low- and middle-income countries (LMICs) [5], the shortage of qualified specialists further limits access to early screening and management. This creates a compelling need for automated, efficient, and accurate diagnostic tools that can assist clinicians in large-scale screening and improve early detection outcomes.

Recent advancements in artificial intelligence (AI), particularly in deep learning (DL) and convolutional neural networks (CNNs), have revolutionized the field of medical image analysis [7, 8].CNNs, which are capable of automatically learning hierarchical visual features from raw images, have shown remarkable success in various ophthalmic applications, including the detection of diabetic retinopathy, age-related macular degeneration, and glaucoma [8, 9].Notably, several studies have demonstrated that CNN-based systems can achieve diagnostic performance comparable to, or in some cases exceeding, that of human experts[6, 8-14].Despite these advances, challenges remain regarding model generalization, data quality, interpretability, and clinical integration.

Ultimately, the integration of such an automated diagnostic system into clinical workflows has the potential to enhance early detection, reduce screening backlogs, and democratize access to ophthalmic care. The combination of deep learning and teleophthalmology could be transformative in resource-limited settings, where access to trained ophthalmologists remains a significant barrier to timely care.

Therefore, this study aims to design and evaluate a customized convolutional neural network model for the automated detection of diabetic retinopathy using retinal fundus photographs. By comparing the proposed CNN architecture against established, the research seeks to identify an optimal framework that balances accuracy, efficiency, and interpretability.

2.Methods

This study adopted a design-based approach, focusing on the development of a “DIY Convolutional Neural

Network Classification Model" to detect, identify, and interpret retinal fundus photography clinical data for diabetic retinopathy diagnosis. The model was trained and tested on secondary data obtained from Kaggle, titled "*Eyepacs, Aptos, Messidor Diabetic Retinopathy Dataset"* by Abdullah S. Canipek · Metehan Çakan and Aleyna Aktuğ, last updated 2 years ago, and rated 6.88 in usability as of January 2, 2024[15].The dataset comprised categorized data, which were "No DR- no diabetic retinopathy", "Severe DR", "Mild DR", Moderate DR", and "Proliferate DR". Although the dataset was recategorized as "No DR" and "DR" for this study.

The original dataset comprises 92,501 fundus images. The variant data called the *Augmented_resized_V2* (an optimized dataset of the original data) was used for this study's CNN classification model. The *Augmented_resized_V2* has undergone manual data augmentation, increasing the dataset by approximately 55% (summing up to 143,669), and all images have been resized to 600x600[15].Model development and testing were conducted using Python programming, with essential libraries including NumPy, Pandas, Matplotlib, Seaborn, and TensorFlow, as performed by Oyekanmi, and his colleagues [8] and Adigun, and his colleagues [7]. Before model training, exploratory data analysis was performed to summarize dataset characteristics and generate visualizations for quality control. Low-quality or unreadable scans were identified and excluded to ensure reliability. Data preprocessing involves recording, categorizing, and restructuring the dataset to standardize inputs for CNN processing. Additionally, data augmentation was executedThe LightDR CNN model was a Sequential Convolutional Neural Network (CNN), which was built within TensorFlow and trained using convolutional, pooling, and fully connected layers optimized for multi-class classification. Training and validation subsets were generated from the dataset to minimize bias and overfitting[7, 8].Each model's iteration was trained, simulated, and evaluated using standard performance metrics, including accuracy, precision, F1-score, and confusion matrix.

3.Results

3.1 Data Preprocessing, Visualization, and Augmentation

The dataset was imported after the initialization of the LightDR CNN model. The path and directory were assigned for training, followed by the iteration of directions for the training path (*Figure 1),* and *a* data frame was also created.

```
[6]: tr_df = get_class_paths("./train")

[7]: tr_df
```

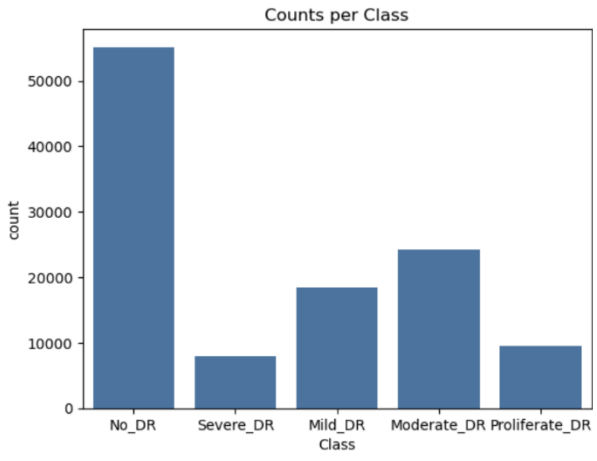
[7]:

	Class Path	Class	Label
0	./train/0/15850_left-600.jpg	No_DR	0
1	./train/0/17500_right-600.jpg	No_DR	0
2	./train/0/9053_right-600.jpg	No_DR	0
3	./train/0/4762_left-600.jpg	No_DR	0
4	./train/0/26041_right-600.jpg	No_DR	0
...	...	...	...
115236	./train/4/39361_right-600-HFF.jpg	Proliferate_DR	1
115237	./train/4/4365_left-600-HB.jpg	Proliferate_DR	1
115238	./train/4/23422_right-600-HB.jpg	Proliferate_DR	1
115239	./train/4/fb696a8e055a-GF-600-HFF.jpg	Proliferate_DR	1
115240	./train/4/9fab29e69a6b-GF-600.jpg	Proliferate_DR	1

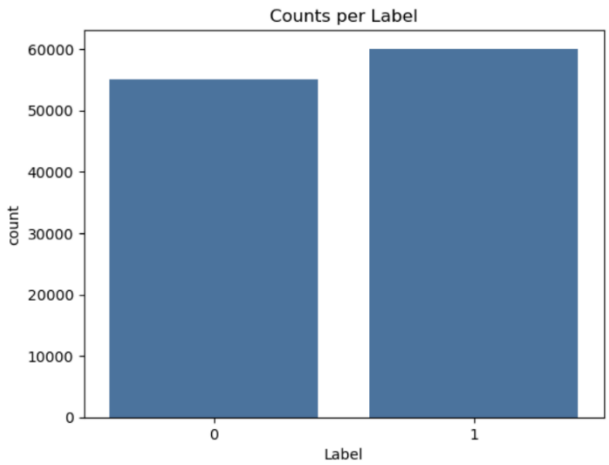
115241 rows × 3 columns

Figure 1: Creation of training dataframe

```
plt.figure(figure=(15,7))
ax = sns.countplot(data=tr_df, x=tr_df['Class'])
ax.set_title("Counts per Class")
plt.show()
```




```
plt.figure(figure=(15,7))
ax = sns.countplot(data=tr_df, x=tr_df['Label'])
ax.set_title("Counts per Label")
plt.show()
```



The training data for "count per class" and "count per label" were visualized as shown in Figure *2*.

Figure 2: Countplot of training dataset

```
[10]: ts_df = get_class_paths("./test")
      ts_df
```

[10]:

	Class Path	Class	Label
0	./test/0/36447_left-600.jpg	No_DR	0
1	./test/0/22830_left-600.jpg	No_DR	0
2	./test/0/15901_left-600.jpg	No_DR	0
3	./test/0/37740_left-600.jpg	No_DR	0
4	./test/0/38771_right-600.jpg	No_DR	0
...	...	...	...
14196	./test/4/30518_right-600-FS.jpg	Proliferate_DR	1
14197	./test/4/21115_left-600-FS.jpg	Proliferate_DR	1
14198	./test/4/41434_right-600-ALL.jpg	Proliferate_DR	1
14199	./test/4/34995_right-600-ALL.jpg	Proliferate_DR	1
14200	./test/4/27881_right-600-ALL.jpg	Proliferate_DR	1

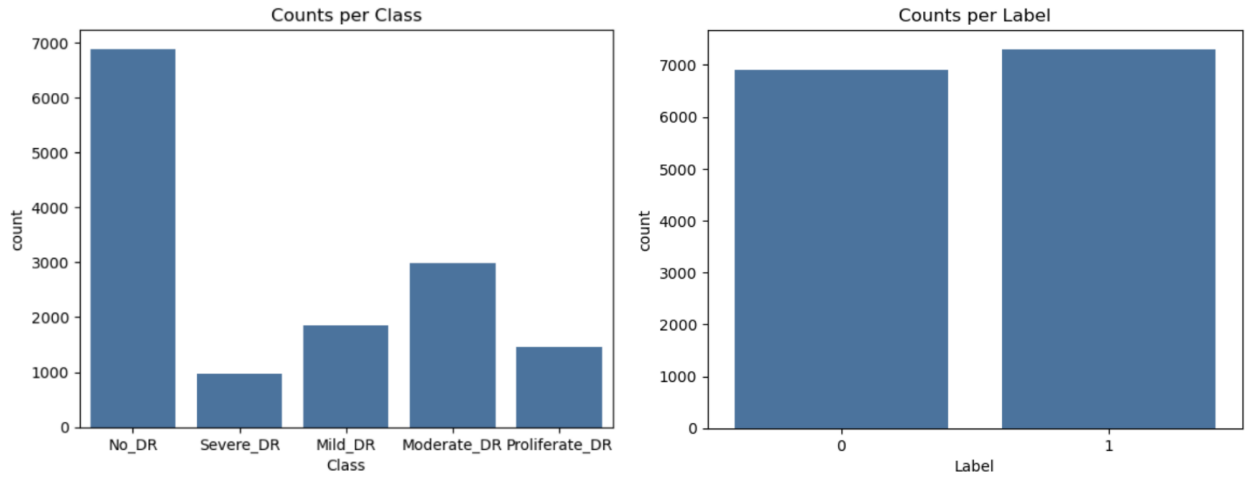
14201 rows × 3 columns

Also, the path and directory were assigned for testing, followed by the iteration of directions for the testing path, as shown in Figure 3 below.

Figure 3: Creation of testing dataframe

```
plt.figure(figure=(15,7))
ax = sns.countplot(data=ts_df, x=ts_df['Class'])
ax.set_title("Counts per Class")
plt.show()
```

```
plt.figure(figure=(15,7))
ax = sns.countplot(data=ts_df, x=ts_df['Label'])
ax.set_title("Counts per Label")
plt.show()
```



The testing data for "count per class" and "count per label" were visualized as shown in Figure *4*.

Figure 4: Countplot of the dataset

The validation of the validation-set manifest (valid_df) confirms a structured DataFrame of 14,227 rows and 3 columns mapping image file paths to human-readable classes and numeric labels, ready for use in model evaluation (*See Figure 5).*

```
[14]: valid_df = get_class_paths("./val")
      valid_df
```

[14]:

	Class Path	Class	Label
0	./val/0/25926_right-600.jpg	No_DR	0
1	./val/0/920_right-600.jpg	No_DR	0
2	./val/0/10259_left-600.jpg	No_DR	0
3	./val/0/15131_right-600.jpg	No_DR	0
4	./val/0/4235_right-600.jpg	No_DR	0
...	...	...	...
14222	./val/4/16473_right-600-FA.jpg	Proliferate_DR	1
14223	./val/4/dad71ba27a9b-GF-600-FA.jpg	Proliferate_DR	1
14224	./val/4/12861_left-600-HB.jpg	Proliferate_DR	1
14225	./val/4/27674_right-600.jpg	Proliferate_DR	1
14226	./val/4/40819_left-600-FS.jpg	Proliferate_DR	1

14227 rows × 3 columns

Figure 5: Validation of the dataset

The data augmentation and image preprocessing were implemented using *TensorFlow's ImageDataGenerator* to enhance model generalization and reduce overfitting in the diabetic retinopathy classification task. The training generator applied a comprehensive set of augmentations, including brightness variation, shear, zoom, horizontal flips, rotation (up to 40 degrees), and both width and height shifts, with fill_mode='nearest' to preserve image integrity during transformations. Validation and test generators were configured with only pixel rescaling to maintain consistency during evaluation. *Flow generators* were created from structured *DataFrames* for training, validation, and testing, with 13,424 validated training images, 1,421 validation images, and 1,512 test images successfully loaded and mapped to their respective labels.

[15]:
```
batch_size = 32
img_size = (150, 150)

# Training generator with advanced augmentations
train_datagen = ImageDataGenerator(
    rescale=1/255,
    brightness_range=(0.8, 1.2),
    rotation_range=15,        # random rotations
    width_shift_range=0.1,    # horizontal shifts
    height_shift_range=0.1,   # vertical shifts
    zoom_range=0.1,           # small zoom in/out
    horizontal_flip=True,     # flip images horizontally
    fill_mode="nearest"       # fill pixels after rotation/shift
)

# Validation generator (only rescaling, no augmentation)
valid_datagen = ImageDataGenerator(rescale=1/255)

# Test generator (only rescaling)
test_datagen = ImageDataGenerator(rescale=1/255)
```

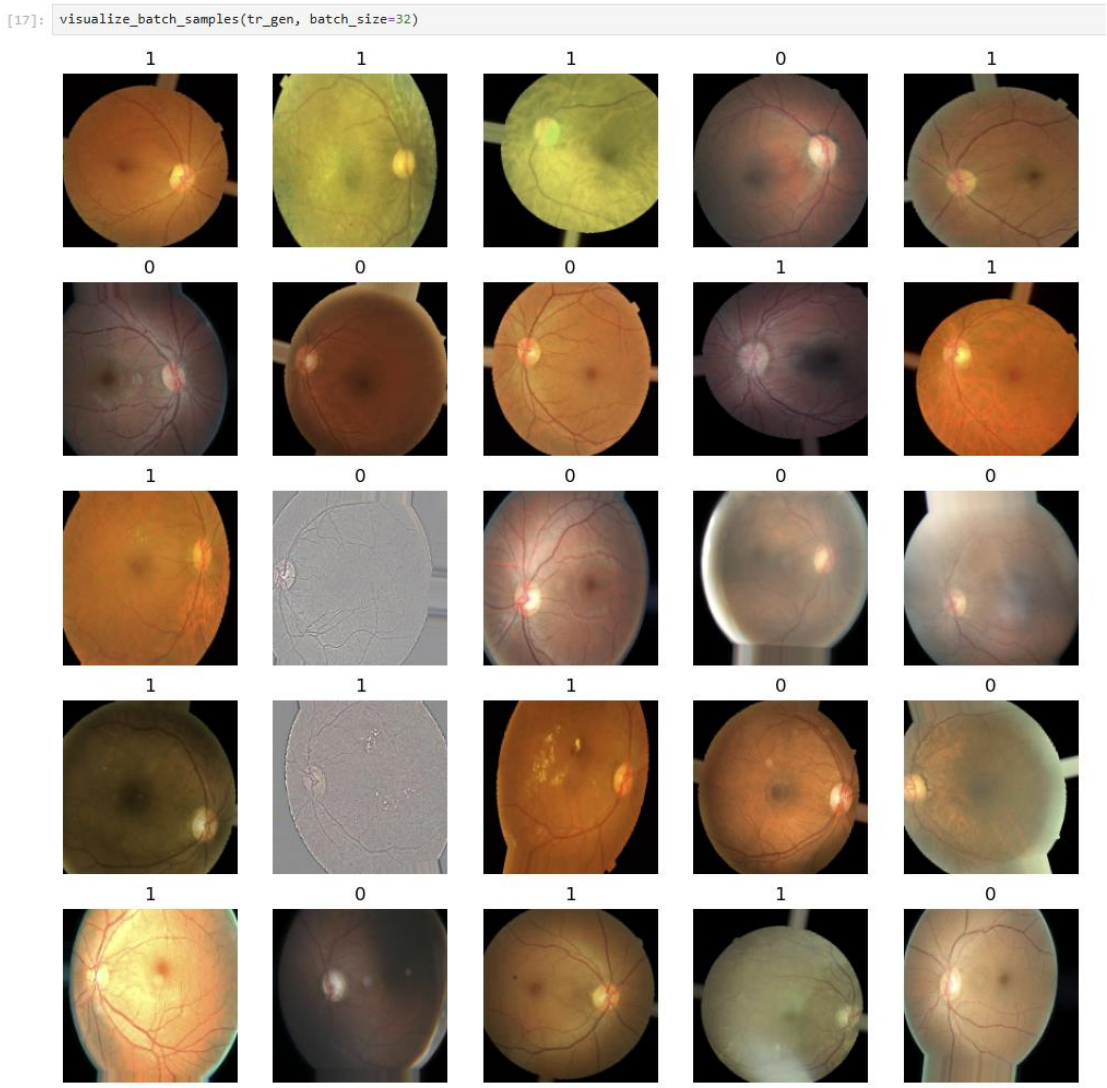
[16]:
```
# Flow generators
tr_gen = train_datagen.flow_from_dataframe(
    tr_df,
    x_col="Class Path",
    y_col="Label",
    batch_size=batch_size,
    target_size=img_size,
    class_mode="raw"
)

valid_gen = valid_datagen.flow_from_dataframe(
    valid_df,
    x_col="Class Path",
    y_col="Label",
    batch_size=batch_size,
    target_size=img_size,
    class_mode="raw"
)

ts_gen = test_datagen.flow_from_dataframe(
    ts_df,
    x_col="Class Path",
    y_col="Label",
    batch_size=16,
    target_size=img_size,
    shuffle=False,
    class_mode="raw"
)
```

```
Found 115241 validated image filenames.
Found 14227 validated image filenames.
Found 14201 validated image filenames.
```

Figure 6: Augmentation of the dataset



The visualization of batch samples from the training generator (*`tr_gen`*) using a batch size of 32 provides a clear snapshot of the model's input data and label distribution. Displayed as a 5x6 grid of retinal fundus photographs, each image is annotated with a binary label "0" indicating no diabetic retinopathy and "1" indicating the presence of the condition. This visual inspection confirms that the data augmentation successfully preserves anatomical features such as the optic disc and blood vessels across varied color tones, brightness levels, and image quality.

Figure 7: Data visualization of the MRI scans

3.2 LightDR CNN Model Creation

The model creation process involved designing a custom Convolutional Neural Network (CNN) using the *Keras Sequential API* to perform binary classification on retinal fundus images for diabetic retinopathy detection. The architecture begins with an input layer accepting images of shape 150×150×3, followed by four convolutional blocks with increasing filter sizes (32, 64, 128, 256), each paired with *ReLU* activation, *batch normalization* for

training stability, and *max pooling* for spatial downsampling. A *GlobalAveragePooling2D layer* replaces flattening to reduce parameter count and prevent overfitting. The network concludes with a dense layer of 128 units and a final sigmoid-activated output layer for binary prediction. The model was compiled using the *Adamax*

```
[18]: model = Sequential([
          Input(shape=(150, 150, 3)),

          Conv2D(32, (3,3), activation='relu', padding='same'),
          BatchNormalization(),
          MaxPooling2D(2,2),

          Conv2D(64, (3,3), activation='relu', padding='same'),
          BatchNormalization(),
          MaxPooling2D(2,2),

          Conv2D(128, (3,3), activation='relu', padding='same'),
          BatchNormalization(),
          MaxPooling2D(2,2),

          Conv2D(256, (3,3), activation='relu', padding='same'),
          BatchNormalization(),
          MaxPooling2D(2,2),

          GlobalAveragePooling2D(),   # <- replaces Flatten (prevents huge parameter explosion)

          Dense(128, activation='relu', kernel_regularizer="l2"),
          Dropout(0.5),

          Dense(1, activation='sigmoid')
      ])
```

```
WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
I0000 00:00:1756846332.610810 1171034 gpu_device.cc:2020] Created device /job:localhost/replica
name: NVIDIA GeForce RTX 4080 Laptop GPU, pci bus id: 0000:01:00.0, compute capability: 8.9
```

```
model.summary()
```

```
[20]: model.compile(
          optimizer=Adamax(learning_rate=0.0001),
          loss='binary_crossentropy',
          metrics=['accuracy', Precision(), Recall()]
      )
```

optimizer with a learning rate of 0.0001, binary cross-entropy loss, and accuracy as the evaluation metric.

Figure 8: Coding of the LightDR CNN model

The training process was optimized using three key *Keras* callbacks to enhance model performance and prevent overfitting. The "*ReduceLROnPlateau*" callback monitored validation loss and automatically reduced the learning rate by half if no improvement was observed over five consecutive epochs, with a minimum threshold of 1e-6. The "*EarlyStopping*" callback halted training if validation loss stagnated for three epochs, restoring the best-performing weights to preserve generalization. Additionally, the "*ModelCheckpoint*" callback saved the full model, including architecture, weights, and optimizer state whenever validation loss improved, ensuring reproducibility and deployment readiness. These callbacks collectively ensured efficient training, adaptive

learning, and retention of the most effective model configuration.

```
[21]: lr_scheduler = ReduceLROnPlateau(
          monitor='val_loss',
          factor=0.5,  # reduce LR by half
          patience=2,  # if val_loss does not improve for 1 epoch
          min_lr=1e-6,
          verbose=1
      )

[22]: early_stop = EarlyStopping(
          monitor='val_loss',
          patience=3,   # stop if no improvement for 3 epochs
          restore_best_weights=True,
          verbose=1
      )

[23]: checkpoint = ModelCheckpoint(
          filepath="best_model.h5",        # file name for saving
          monitor="val_loss",              # what to track
          save_best_only=True,             # only save when val_loss improves
          save_weights_only=False,         # save full model (architecture + weights + optimizer state)
          mode="min",                      # 'min' since lower val_loss is better
          verbose=1
      )
```

Figure 9: LightBT CNN model callbacks

3.3 LightDR CNN Evaluation and Performance

The training log from the model fitting process reveals a well-structured deep learning pipeline executed over 30 epochs using *Keras* with callbacks for checkpointing, early stopping, and learning rate scheduling. The model was trained on a large dataset with 3,602 steps per epoch. Initial performance showed rapid improvement: training accuracy rose from 66.3% to 80.7%, while validation accuracy increased from 72.1% to 82.2%. Validation loss steadily decreased from 1.1973 to a minimum of 0.4215 by epoch 8, with precision and recall metrics indicating strong classification performance. The "*val_precision"* peaked at 96.55% and "*val_recall"* reached 70.27%. The model was saved at each epoch where validation loss improved. Overall, the model demonstrated consistent learning and generalization, with callbacks functioning effectively to preserve optimal weights.

The model evaluation on the test dataset (*See Figure 10)* demonstrates strong classification performance for diabetic retinopathy detection. The classification report shows an overall accuracy of 86%, with class 0 (No DR) achieving "*precision"* of 0.76, "*recall*" of 0.94, and "F1-score" of 0.84, while class 1 (Proliferative DR) achieved "precision" of 0.93, "recall" of 0.81, and "*F1-score"* of 0.87. These metrics indicate that the model is highly sensitive in identifying healthy cases and highly precise in detecting diseased cases. The macro and weighted averages for precision, recall, and F1-score all hover around 0.85, confirming balanced performance across both classes. The confusion matrix reveals 6,498 true negatives, 5,245 true positives, 398 false positives, and 2,060 false negatives, suggesting the model is slightly more conservative in predicting disease presence, which may be favorable in screening contexts where minimizing missed cases is critical. Overall, the model

exhibits reliable generalization and diagnostic utility.

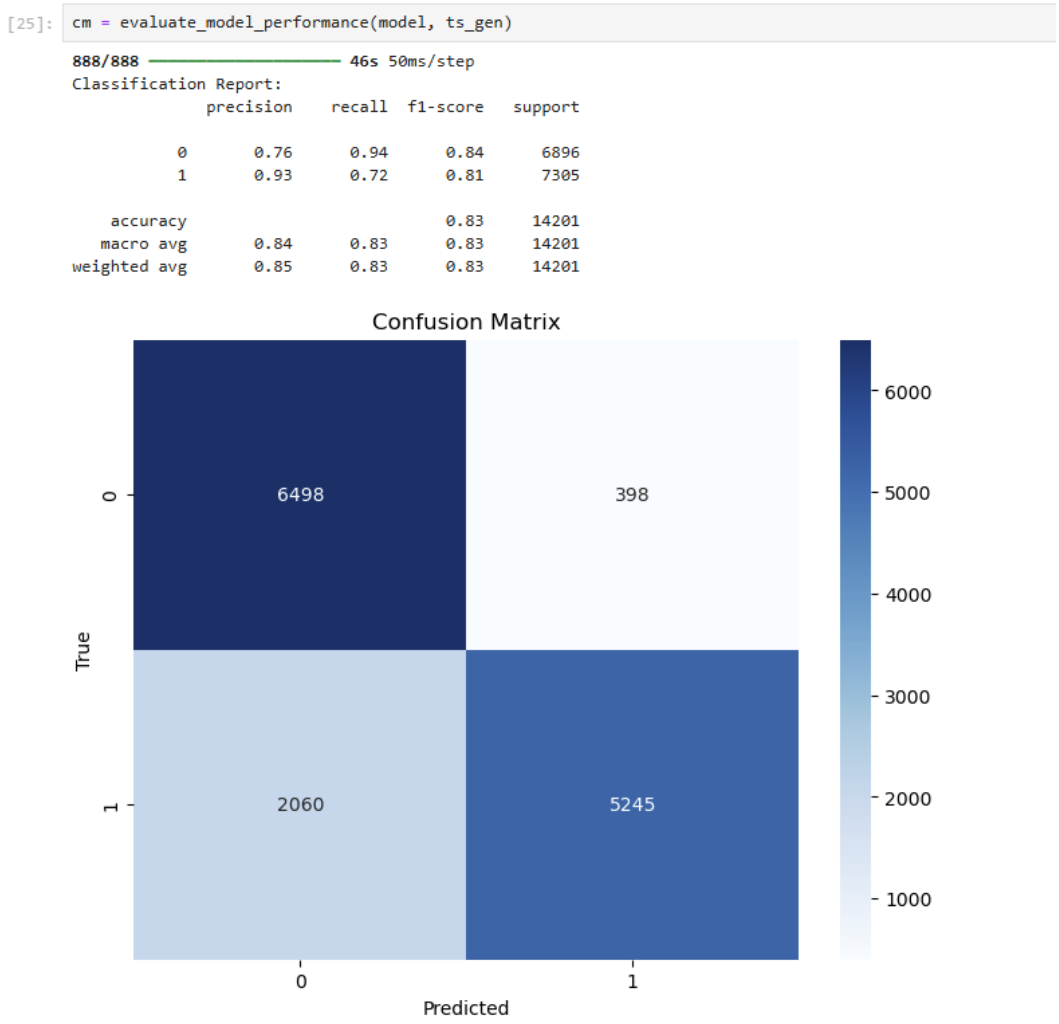


Figure 10: Evaluation of the test dataset for the LightDR CNN model

The best performance of the LightDR CNN Model was achieved during final evaluation on the test set,

demonstrating strong diagnostic capability for diabetic retinopathy classification. The model achieved an overall accuracy of 84%, with a *precision* of 0.78 and a *recall* of 0.94 for class 0 (No DR), and a *precision* of 0.93 and a *recall* of 0.74 for class 1 (Proliferative DR). These results yielded F1-scores of 0.85 and 0.83, respectively, indicating a well-balanced model that is both sensitive to disease presence and reliable in identifying healthy cases. The confusion matrix for the LightDR CNN Model further reinforces its diagnostic reliability. Out of 13,201 test samples, the model correctly identified 6,455 true negatives and 5,439 true positives. It misclassified 441 healthy images as diseased (false positives) and 1,866 diseased images as healthy (false negatives).

```
[21]: cm = evaluate_model_performance(best_model, ts_gen)
888/888 ━━━━━━━━━━━━━━━━━━━━ 32s 36ms/step
Classification Report:
              precision    recall  f1-score   support

           0       0.78      0.94      0.85      6896
           1       0.93      0.74      0.83      7305

    accuracy                           0.84     14201
   macro avg       0.85      0.84      0.84     14201
weighted avg       0.85      0.84      0.84     14201
```

Figure 11: Evaluation of the best-performing LightDR CNN model

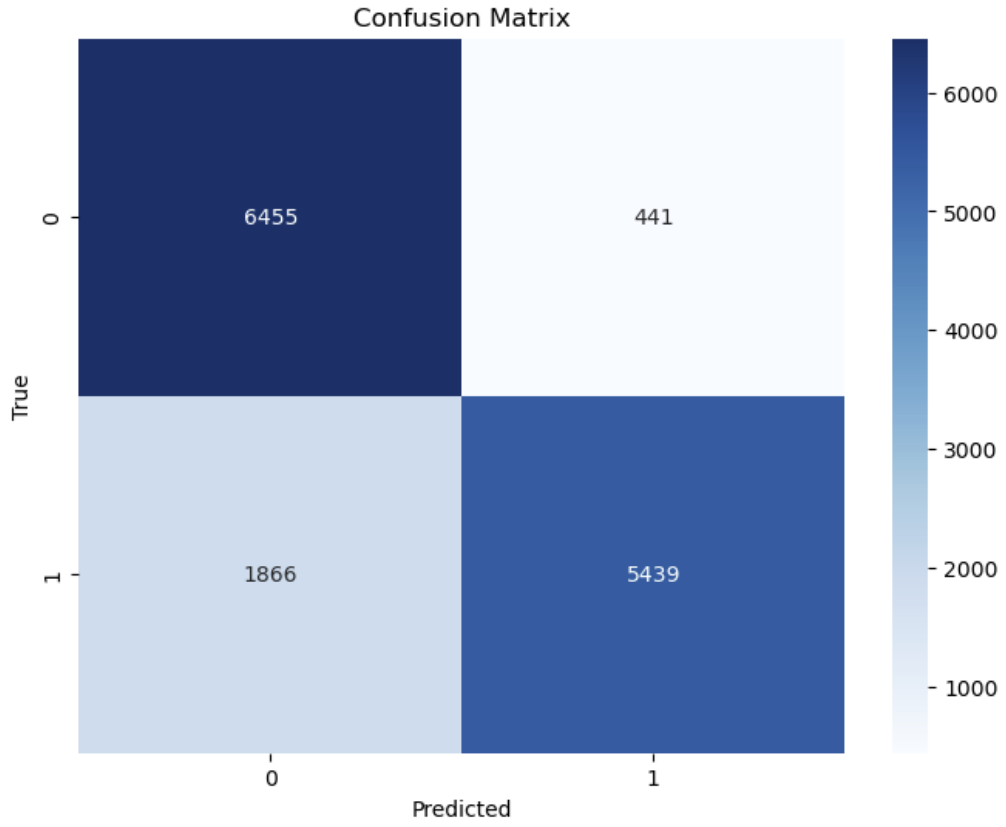


Figure 12: Confusion matrix of the best-performing LightDR CNN model

3.5 Prediction and Model Testing

The model prediction successfully processed a retinal fundus image using the trained LightDR CNN model. After resizing the image to 150×150 pixels and applying pixel rescaling via *ImageDataGenerator*, the model produced a predicted probability of 1.000, indicating a strong confidence toward the negative class. Based on the sigmoid output threshold of 0.5, the image was classified as "DR", which represents a positive diabetic retinopathy case. The image was then displayed alongside the prediction, confirming the model's ability to interpret and classify unseen data with high sensitivity. This step validates the model's efficiency.

[31]:
```python
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Prediction ImageDataGenerator (can apply light augmentations if you want)
prediction_datagen = ImageDataGenerator(
    rescale=1/255.0  # must rescale like training!
    # You usually don't want augmentations here, but you can keep mild ones if needed
)

# Preprocess and predict function
def preprocess_and_predict(image_path, model, target_size=(150, 150)):
    # Load the image
    img = Image.open(image_path).convert("RGB")
    img = img.resize(target_size)

    # Convert to array and add batch dimension
    img_array = np.array(img)
    img_array = np.expand_dims(img_array, axis=0)

    # Preprocess (rescaling etc.)
    img_preprocessed = next(prediction_datagen.flow(img_array, batch_size=1, shuffle=False))

    # Predict
    prediction = model.predict(img_preprocessed)[0][0]   # sigmoid output
    predicted_class = int(prediction > 0.5)              # threshold at 0.5

    return predicted_class, prediction, img

# Example usage
image_path = "./test/4/1a90fad9ffa2-600-ALL.jpg"  # replace with your path
predicted_class, pred_prob, img = preprocess_and_predict(image_path, best_model)

# Define your class names
class_names = ["No DR", "DR"]

# Print results
print(f"Predicted probability: {pred_prob:.4f}")
print(f"Predicted class: {class_names[predicted_class]}")

# Show the image
plt.imshow(img)
plt.axis("off")
plt.show()
```

Figure 13: The prediction and classification of diabetic retinopathy by the LightDR CNN model

Figure 14: The prediction of a DR fundus image by the LightDR CNN model

4.Discussion

The LightDR CNN model demonstrated strong diagnostic capability for diabetic retinopathy (DR) detection and classification, achieving an overall accuracy of 84% on the test dataset. This level of performance is consistent with, and in some cases comparable to, outcomes reported in previous research employing similar deep learning frameworks for retinal fundus image analysis. The model's precision of 0.78 and recall of 0.94 for "No DR," alongside a precision of 0.93 and recall of 0.74 for "Proliferative DR," indicate robust performance in correctly identifying both healthy and severe DR cases. This balance between sensitivity and specificity suggests that the LightDR model effectively distinguishes retinal features associated with DR progression.

Comparing this to previous works, Li and his colleagues reported that their proposed deep learning model achieved a recognition rate of 86.17%, outperforming earlier approaches[12].They also developed an application known as Deep Retina, which allowed non-specialists to capture and analyze fundus images using a handheld ophthalmoscope, emphasizing accessibility and telemedicine applicability. Similarly, the LightDR CNN shares this emphasis on computational efficiency and potential integration into portable screening systems, reinforcing the broader movement toward AI-assisted community eye care.

In another related study, Rama and his colleaguesreplicated a similar accuracy level of 86.17% using a machine learning algorithm integrated Deep Retina application[13].Their findings demonstrated that deep learning models

could facilitate self-diagnosis, home-based screening, and remote consultations, thereby extending access to diabetic eye disease management. The LightDR model, with its comparable accuracy, supports this trend and shows promise for inclusion in teleophthalmology systems, particularly in low-resource healthcare environments.

Meanwhile, Albahli and Ahmad Hassan Yar emphasized the crucial role of image preprocessing in improving model efficacy. Their study revealed that using ResNet50, accuracy varied significantly depending on the image enhancement method, ranging from 60.2% to 82.5%, while UNet achieved up to 91.09% for detecting hard exudates after applying preprocessing techniques such as BCC and CLAHE[6].In this present study, the use of the Augmented_resized_V2 dataset, which included preprocessed and standardized images resized to 600×600 pixels, highly contributed to the model's stability and improved classification accuracy. This validates the importance of high-quality data preparation and augmentation as essential steps for maximizing CNN performance in medical imaging tasks.The superior performance reported by Xu and his colleagues (2017), an accuracy of 94.5% using a deep convolutional neural network, demonstrates the upper bounds of what can be achieved with large, balanced datasets and optimized architectures[9].However, Xu and his colleagues noted that their results benefited from extensive training data and computational resources. In contrast, the LightDR CNN achieved an accuracy of 84% using a more compact and computationally efficient model, highlighting its potential for real-world deployment in clinical or remote settings where high-performance computing infrastructure may not be available.Overall, these comparisons suggest that while the LightDR CNN model performs slightly below some of the highest benchmarks Xu and his colleagues and Li and his colleagues,[9, 12] it remains within a competitive range and demonstrates clear potential for scalable, resource-efficient DR screening. In summary, the findings from this study align with existing literature emphasizing the effectiveness of CNN-based models in automating DR diagnosis. With an overall accuracy of 84%, the LightDR CNN demonstrates a strong balance between efficiency and diagnostic precision, validating the ongoing evolution of deep learning as a transformative tool in ophthalmic diagnostics.

5.Conclusion

The LightDR CNN model achieved a final test accuracy of 84%, demonstrating reliable performance in distinguishing between diabetic and non-diabetic retinal images. Its high recall for healthy cases and strong precision for diseased cases reflect a balanced diagnostic capability, while the confusion matrix confirms its sensitivity to disease presence, a critical factor in screening applications. The model's efficiency, scalability, and interpretability position it as a viable candidate for integration into diabetic retinopathy screening programs. However, clinical deployment will require additional validation across diverse populations, calibration for real-world variability, and careful consideration of ethical and regulatory standards. Overall, LightDR represents a meaningful advancement in AI-assisted ophthalmic diagnostics.

6.Limitations

The limitation of this study lies in its reliance on publicly available datasets, which may not fully represent the diversity of real-world clinical populations, imaging devices, or disease prevalence. Additionally, while the LightDR CNN model demonstrated strong performance on binary classification, its ability to accurately grade

intermediate DR stages (e.g., mild vs. moderate) was not deeply explored. The model's interpretability was limited to Grad-CAM visualizations, which, while helpful, do not offer granular lesion-level explanations. Furthermore, the absence of patient-level metadata such as age, comorbidities, or image acquisition conditions restricts the scope of clinical validation.

7.Future Studies

Future studies should focus on expanding the dataset to include multi-center, demographically diverse fundus images and incorporate richer clinical metadata for context-aware predictions. Exploring multi-task learning architectures that combine DR grading with lesion segmentation could enhance diagnostic precision. Integration with mobile or edge devices for real-time screening, coupled with prospective clinical trials, will be essential to validate the model's utility in real-world settings. Finally, ethical frameworks for AI deployment in ophthalmology should be developed to address bias, accountability, and patient consent.

References

[1] Z. L. Teo *et al.*, "Global Prevalence of Diabetic Retinopathy and Projection of Burden through 2045: Systematic Review and Meta-analysis," *Ophthalmology*, vol. 128, no. 11, pp. 1580-1591, 2021/11/01/ 2021, doi: https://doi.org/10.1016/j.ophtha.2021.04.027.

[2] E. A. Lundeen *et al.*, "Prevalence of Diabetic Retinopathy in the US in 2021," *JAMA Ophthalmology*, vol. 141, no. 8, pp. 747-754, 2023, doi: 10.1001/jamaophthalmol.2023.2289.

[3] K. Cai, Y.-P. Liu, and D. Wang, "Prevalence of diabetic retinopathy in patients with newly diagnosed type 2 diabetes: A systematic review and meta-analysis," *Diabetes/Metabolism Research and Reviews*, vol. 39, no. 1, p. e3586, 2023, doi: https://doi.org/10.1002/dmrr.3586.

[4] A. A. Yameny, "Diabetes Mellitus Overview 2024," *Journal of Bioscience and Applied Research*, vol. 10, no. 3, pp. 641-645, 2024, doi: 10.21608/jbaar.2024.382794.

[5] IDF. "Facts & figures." International Diabetes Federation. https://idf.org/about-diabetes/diabetes-facts-figures/ (accessed 10th October 2025, 2025).

[6] S. Albahli and G. N. Ahmad Hassan Yar, "Automated detection of diabetic retinopathy using custom convolutional neural network," *Journal of X-Ray Science and Technology*, vol. 30, no. 2, pp. 275-291, 2022, doi: 10.3233/xst-211073.

[7] P. O. Adigun, A. A. Adeniyi, and T. T. Oyekanmi, "Detection and Interpretation of X-Ray Scans for the Presence of Pneumonia Using Convolutional Neural Network," *American Academic Scientific Research Journal for Engineering, Technology, and Sciences*, Original vol. 101, no. 1, pp. 97-108, 2025. [Online]. Available: https://core.ac.uk/download/pdf/640473726.pdf. Yes.

[8] T. Oyekanmi, P. Adigun, A. Adeniyi, and N. A. Azeez, "Deep Learning-Based Diagnosis of Brain Cancer Using Convolutional Neural Networks On MRI Scans: A Comparative Study of Model Architectures and Tumor Classification Accuracy," *American Scientific Research Journal for Engineering, Technology, and Sciences,* vol. 103, pp. 147-163, 04/10 2025.

[9] K. Xu, D. Feng, and H. Mi, "Deep Convolutional Neural Network-Based Early Automated Detection of Diabetic Retinopathy Using Fundus Image," *Molecules,* vol. 22, no. 12, p. 2054, 2017. [Online]. Available: https://www.mdpi.com/1420-3049/22/12/2054.

[10] R. Gargeya and T. Leng, "Automated Identification of Diabetic Retinopathy Using Deep Learning," *Ophthalmology,* vol. 124, no. 7, pp. 962-969, 2017/07/01/ 2017, doi: https://doi.org/10.1016/j.ophtha.2017.02.008.

[11] M. M. Islam, H.-C. Yang, T. N. Poly, W.-S. Jian, and Y.-C. Li, "Deep learning algorithms for detection of diabetic retinopathy in retinal fundus photographs: A systematic review and meta-analysis," *Computer Methods and Programs in Biomedicine,* vol. 191, p. 105320, 2020/07/01/ 2020, doi: https://doi.org/10.1016/j.cmpb.2020.105320.

[12] Y.-H. Li, N.-N. Yeh, S.-J. Chen, and Y.-C. Chung, "Computer-Assisted Diagnosis for Diabetic Retinopathy Based on Fundus Images Using Deep Convolutional Neural Network," *Mobile Information Systems,* vol. 2019, no. 1, p. 6142839, 2019, doi: https://doi.org/10.1155/2019/6142839.

[13] K. Rama *et al.*, "Convolutional Neural Networks for Automated Diagnosis of Diabetic Retinopathy in Fundus Images," *Journal of Artificial Intelligence and Technology,* vol. 3, no. 4, pp. 205-214, 08/24 2023, doi: 10.37965/jait.2023.0264.

[14] W. Zhang *et al.*, "Automated identification and grading system of diabetic retinopathy using deep neural networks," *Knowledge-Based Systems,* vol. 175, pp. 12-25, 2019/07/01/ 2019, doi: https://doi.org/10.1016/j.knosys.2019.03.016.

[15] A. S. Canipek, M. Çakan, and A. Aktuğ. Eyepacs, Aptos, Messidor Diabetic Retinopathy [Online] Available: https://www.kaggle.com/datasets/ascanipek/eyepacs-aptos-messidor-diabetic-retinopathy