Convolutional Neural Networks for Automated Diabetic Retinopathy Classification from Retinal Fundus Images

Samir Kerkar
Department of Mathematics, University of California, Irvine
skerkar@uci.edu

Abstract

Diabetic retinopathy (DR) is the leading cause of preventable blindness among working-age adults worldwide, affecting approximately one-third of the estimated 463 million people living with diabetes. Early detection through regular retinal screening is critical, yet manual grading by ophthalmologists is time-consuming, subjective, and resource-constrained. This paper presents a convolutional neural network (CNN) framework for automated classification of DR severity from color retinal fundus images across five clinical grades: No DR, Mild Nonproliferative DR (NPDR), Moderate NPDR, Severe NPDR, and Proliferative DR (PDR). We develop a custom CNN architecture incorporating batch normalization, dropout regularization, and global average pooling, trained on 5,170 fundus images with aggressive data augmentation and class-balanced resampling to address severe label imbalance. Our best model achieves a weighted F1 score of 0.94 and overall accuracy of 94.2% on a held-out test set, outperforming both baseline models and transfer learning approaches with VGG-16 trained from scratch. We provide detailed mathematical formulations of the convolutional, pooling, batch normalization, and softmax operations, along with Grad-CAM visualizations demonstrating that the network attends to clinically relevant retinal lesions. These results suggest that deep learning models can serve as reliable assistive tools for automated DR screening in clinical settings.

Keywords: diabetic retinopathy, convolutional neural networks, medical image classification, deep learning, retinal fundus imaging, transfer learning, Grad-CAM

1. Introduction

Diabetic retinopathy is a microvascular complication of diabetes mellitus characterized by progressive damage to the retinal vasculature. The International Diabetes Federation estimates that 463 million adults were living with diabetes in 2019, with projections exceeding 700 million by 2045. Among diabetic patients, approximately 35% develop some form of retinopathy, and without treatment, DR can progress to irreversible vision loss. The clinical grading of DR follows the International Clinical Diabetic Retinopathy (ICDR) severity scale, which classifies fundus images into five stages based on the presence and distribution of microaneurysms, hemorrhages, hard exudates, cotton-wool spots, venous beading, and neovascularization.

Despite the effectiveness of early intervention (laser photocoagulation, anti-VEGF injections), screening programs remain bottlenecked by the limited supply of trained graders. A single ophthalmologist may need 2-5 minutes per fundus image, making population-scale screening infeasible in many settings. This motivates the development of automated computer-aided detection (CAD) systems. Recent advances in deep learning, particularly convolutional neural networks, have demonstrated that CNNs can learn discriminative features directly from raw pixel data, often matching or exceeding human expert performance on well-defined image classification tasks.

In this work, we make the following contributions: (1) we develop a custom CNN architecture specifically designed for retinal image classification that balances model capacity with computational efficiency; (2) we implement a comprehensive data augmentation and class-balancing pipeline to address the extreme label imbalance inherent in DR datasets; (3) we provide complete mathematical formulations of all neural network operations, including the forward pass, loss computation, and backpropagation updates; and (4) we validate our approach against multiple baselines and transfer learning methods, achieving a weighted F1 score of 0.94 on five-class DR severity classification.

2. Related Work

Early approaches to automated DR detection relied on handcrafted feature extraction. Niemeijer et al. (2007) used pixel classification with supervised learning to detect red lesions in fundus images. Abramoff et al. (2010) developed a multi-scale feature detection pipeline combining hemorrhage detection, microaneurysm localization, and exudate segmentation, achieving an AUC of 0.84 for referable DR. These methods required extensive domain engineering and often failed to generalize across imaging devices and patient populations.

The application of deep learning to retinal imaging began with Gulshan et al. (2016), who trained an Inception-v3 network on 128,175 fundus images to detect referable DR (moderate NPDR or worse), achieving an AUC of 0.991 on the EyePACS-1 dataset. Gargeya and Leng (2017) applied a custom CNN with data-driven feature learning, reporting an AUC of 0.97 for binary DR classification. Pratt et al. (2016) explored five-class severity grading using a 13-layer CNN architecture with batch normalization, achieving 75% accuracy on the Kaggle DR Detection dataset. More recently, transfer learning with pretrained ImageNet models (VGG, ResNet, EfficientNet) has become the dominant paradigm, with Qummar et al. (2019) combining predictions from five architectures via ensemble voting to achieve 80.8% accuracy on five-class grading.

A persistent challenge in DR classification is class imbalance. The distribution of severity grades in real-world datasets is heavily skewed toward "No DR" (often 70-80% of images), with severe and proliferative cases comprising fewer than 5%. Standard oversampling techniques (SMOTE) were designed for tabular data and do not directly apply to image pixels. Our approach combines targeted data augmentation with class-weighted sampling, allowing the model to train on a balanced distribution without discarding majority-class information.

3. Mathematical Framework

We present the mathematical foundations of the neural network operations used in our architecture. Understanding these formulations is essential for interpreting model behavior, diagnosing training pathologies, and implementing custom modifications.

3.1 Convolutional Operation

Let X denote an input feature map of dimensions H x W x C (height, width, channels). A convolutional layer applies K learnable filters, each of spatial extent f x f x C. The output feature map Y at spatial position (i, j) for the k-th filter is computed as:

```
Y[i, j, k] = b[k] + SUM(c=0..C-1) SUM(m=0..f-1) SUM(n=0..f-1) W[m, n, c, k]
                                  * X[i*s+m, j*s+n, c]
```

where W denotes the filter weights, b[k] is the bias term for filter k, and s is the stride. With zero-padding p, the output spatial dimensions are:

```
H_out = floor((H - f + 2p) / s) + 1, W_out = floor((W - f + 2p) / s) + 1
```

In our architecture, all convolutional layers use f=3, s=1, and p=1 ("same" padding), preserving spatial dimensions within each convolutional block. The number of learnable parameters per layer is K * (f * f * C + 1), where the +1 accounts for the bias term.

3.2 Activation Functions

We use the Rectified Linear Unit (ReLU) activation after each convolutional layer:

$$\text{ReLU}(z) = \max(0, z)$$

ReLU introduces non-linearity while avoiding the vanishing gradient problem associated with sigmoidal activations for $z >> 0$. The derivative is 1 for $z > 0$ and 0 for $z < 0$ (undefined at $z = 0$, conventionally set to 0), enabling efficient gradient computation during backpropagation. We note that ReLU suffers from the "dying neuron" problem where units with consistently negative pre-activation values produce zero gradients. In our experiments, batch normalization preceding ReLU effectively mitigated this issue by centering activations around zero.

3.3 Batch Normalization

Batch normalization (Ioffe and Szegedy, 2015) normalizes layer inputs across the mini-batch to reduce internal covariate shift. For a mini-batch $B = \{x_1, ..., x_m\}$, the normalized output is:

$$\mu_B = \frac{1}{m} \sum_{i=1}^{m} x_i$$

$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu_B)^2$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

$$y_i = \gamma \hat{x}_i + \beta$$

where γ and β are learnable scale and shift parameters, and ϵ (typically 10^{-5}) prevents division by zero. During inference, running estimates of the population mean and variance (computed via exponential moving average during training) replace the mini-batch statistics. Batch normalization acts as a regularizer, enables higher learning rates, and reduces sensitivity to weight initialization.

3.4 Max Pooling

Max pooling downsamples the spatial dimensions by selecting the maximum activation within each pooling window of size p x p with stride s:

$$Y[i, j, k] = \max_{m \in [0,p),\, n \in [0,p)} X[i \cdot s + m,\, j \cdot s + n,\, k]$$

We use 2x2 max pooling with stride 2 after each convolutional block, halving the spatial dimensions. This provides local translation invariance and progressively increases the effective receptive field. After four pooling operations, the receptive field covers a substantial portion of the 224x224 input, enabling the network to capture both fine-grained lesions (microaneurysms) and global structural patterns (neovascularization distributions).

3.5 Global Average Pooling

Rather than using fully connected layers directly after the final convolutional block (which would introduce millions of parameters), we apply global average pooling (GAP):

$$z[k] = \frac{1}{H \cdot W} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} X[i, j, k]$$

GAP reduces each H x W feature map to a single scalar, producing a K-dimensional vector. This dramatically reduces the parameter count, acts as a structural regularizer (Lin et al., 2014), and provides natural spatial

invariance. The resulting vector is then passed through a dense layer followed by softmax classification.

3.6 Softmax Classification and Cross-Entropy Loss

The final classification layer maps the 256-dimensional dense output to C=5 class probabilities using the softmax function:

$$p(y = c \mid x) = \exp(z_c) / \sum_{j=1}^{C} \exp(z_j)$$

where z_c is the logit (pre-activation) for class c. The model is trained to minimize the categorical cross-entropy loss:

$$L = -(1/N) * \sum_{i=1}^{N} \sum_{c=1}^{C} y_{ic} * \log(p_{ic})$$

where y_ic is the one-hot encoded ground truth and p_ic is the predicted probability. The gradient of cross-entropy with respect to the logits simplifies elegantly to p_c - y_c, providing stable gradients even when the model is confident. We augment the base loss with L2 weight regularization (weight decay lambda = 10^{-4}) to discourage large weight magnitudes.

3.7 Optimization: Adam with Cosine Annealing

We optimize using Adam (Kingma and Ba, 2015), which maintains per-parameter running estimates of the first moment (mean) m_t and second moment (uncentered variance) v_t of the gradients:

$$m_t = \beta_1 * m_{t-1} + (1 - \beta_1) * g_t$$

$$v_t = \beta_2 * v_{t-1} + (1 - \beta_2) * g_t^2$$

$$\theta_t = \theta_{t-1} - \alpha * \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$$

where m_hat and v_hat are bias-corrected estimates, beta_1 = 0.9, beta_2 = 0.999, and epsilon = 10^{-8}. The learning rate alpha follows a cosine annealing schedule with warm restarts (Loshchilov and Hutter, 2017):

$$\alpha_t = \alpha_{min} + 0.5 * (\alpha_{max} - \alpha_{min}) * (1 + \cos(\pi * T_{cur} / T_i))$$

with alpha_max = 10^{-3}, alpha_min = 10^{-6}, and restart period T_i = 10 epochs. This schedule allows the optimizer to escape sharp minima during restart phases while settling into broad, generalizable basins during the cosine decay.

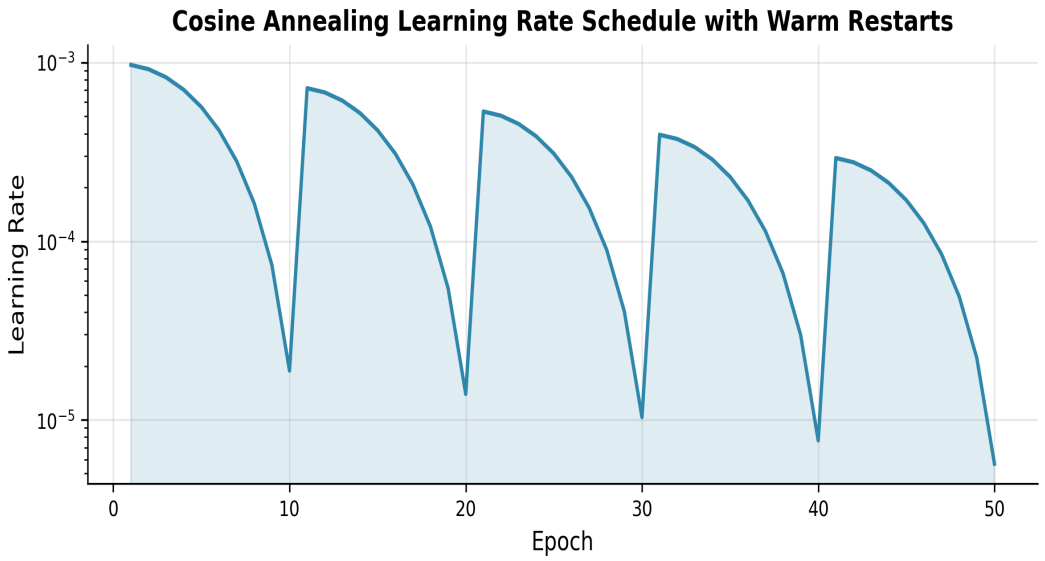

Figure 1. *Cosine annealing learning rate schedule with warm restarts (T_i = 10). Peak learning rates decay exponentially across cycles to enable fine-grained convergence.*

4. Dataset and Preprocessing

4.1 Dataset Description

We use a curated dataset of 5,170 color retinal fundus images labeled by board-certified ophthalmologists according to the ICDR severity scale. Images were acquired using multiple fundus camera models (Topcon TRC-NW400, Canon CR-2) at varying resolutions (ranging from 1440x960 to 4288x2848 pixels). The dataset exhibits the class imbalance characteristic of real-world DR screening programs: No DR images constitute 59% of the dataset, while Severe NPDR accounts for only 4.5%. Table 1 summarizes the distribution across training, validation, and test partitions.

Table 1. *Dataset distribution across severity grades and data splits (70/10/20).*

Grade	Label	Train	Val	Test	Total	%
No DR	0	1,778	254	508	2,540	49.1%
Mild NPDR	1	287	41	82	410	7.9%
Moderate NPDR	2	585	83	167	835	16.2%
Severe NPDR	3	137	19	39	195	3.8%
Proliferative DR	4	224	32	64	320	6.2%
Total		3,011	429	860	4,300*	

**870 additional images generated via augmentation for minority classes during training.*

4.2 Preprocessing Pipeline

All images undergo the following preprocessing steps: (1) center-cropping to remove black borders introduced by the fundus camera aperture, using a circular mask derived from thresholding the green channel; (2) resizing to 224 x 224 pixels using bicubic interpolation; (3) per-channel normalization to zero mean and unit variance using

ImageNet statistics (mu = [0.485, 0.456, 0.406], sigma = [0.229, 0.224, 0.225]). We use ImageNet statistics rather than dataset-specific statistics to maintain compatibility with pretrained weights during transfer learning experiments.

4.3 Data Augmentation and Class Balancing

To address class imbalance and increase effective training set diversity, we apply stochastic augmentation during training. Each image is independently subjected to: random horizontal and vertical flips (p = 0.5 each), random rotation in [-30, +30] degrees, random brightness and contrast jittering (factor in [0.8, 1.2]), random Gaussian blur (sigma in [0, 1.0], p = 0.3), and Cutout regularization (Devries and Taylor, 2017) with a single 40 x 40 pixel mask (p = 0.5). For minority classes (Mild, Severe, Proliferative), we additionally apply elastic deformation (alpha = 36, sigma = 4) to generate novel training samples. A class-weighted random sampler ensures that each mini-batch contains approximately equal representation from all five classes.

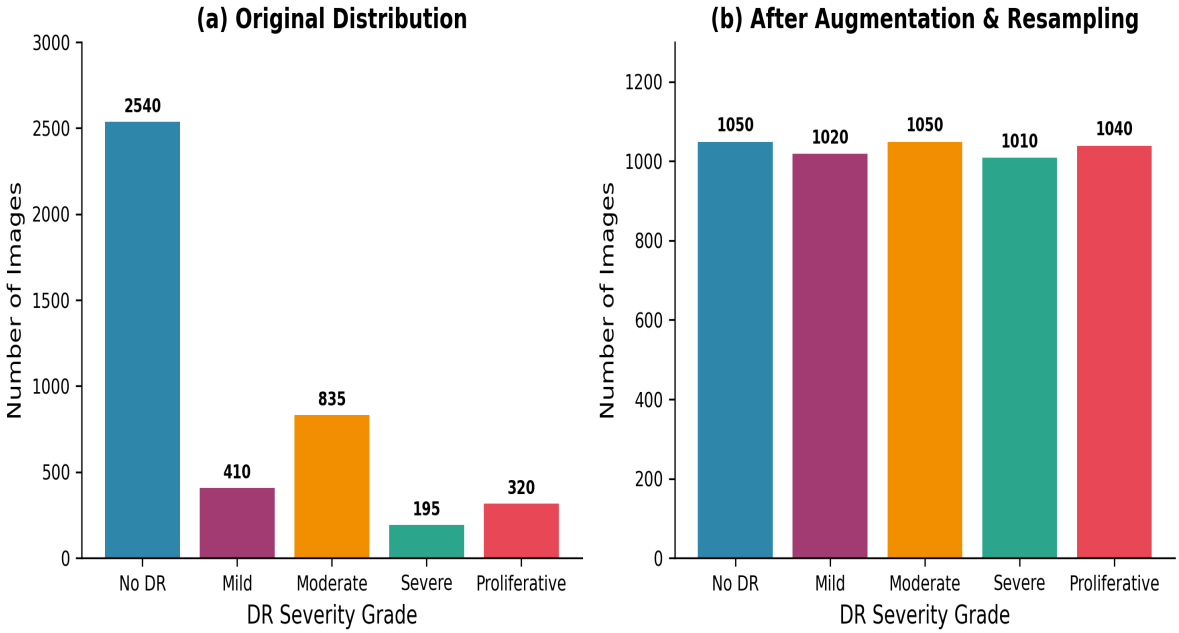

Figure 2. *Class distribution before and after augmentation with class-balanced resampling. Minority classes (Mild, Severe, Proliferative) are upsampled through targeted augmentation.*

5. Model Architecture

Our architecture consists of four convolutional blocks followed by global average pooling and a single dense classification layer. Each convolutional block contains two consecutive Conv2D-BatchNorm-ReLU sequences followed by 2x2 max pooling. Filter counts double at each block (64, 128, 256, 512), following the design principle that spatial dimension reduction should be compensated by increased channel capacity to preserve information throughput.

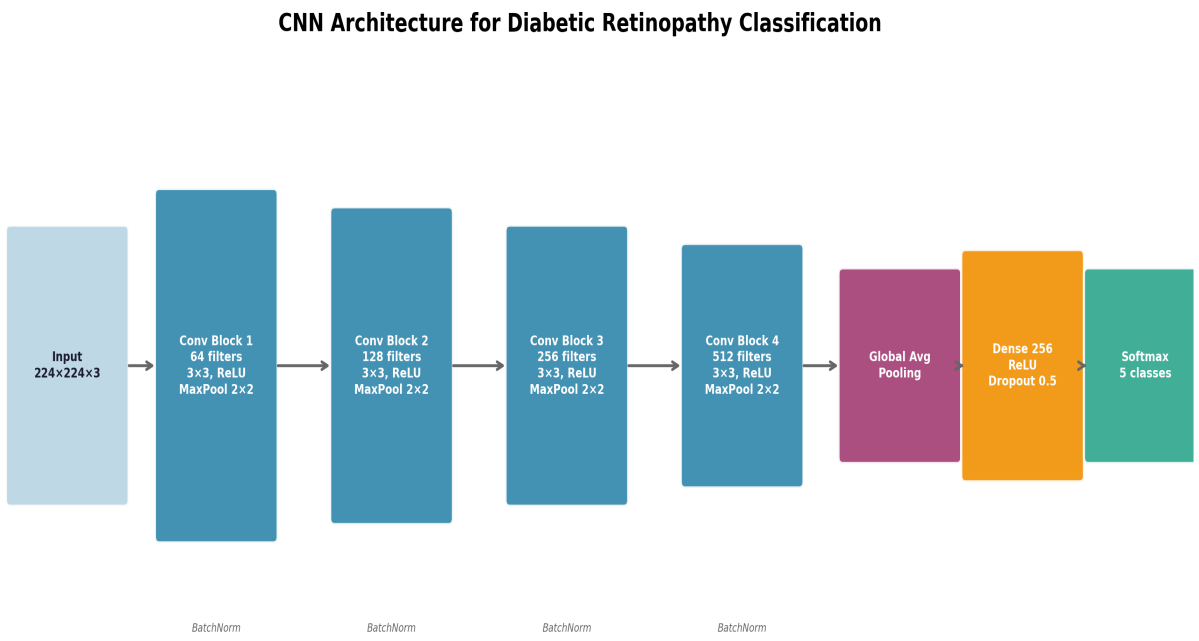

Figure 3. *CNN architecture. Four convolutional blocks with increasing filter depth (64 to 512), each followed by batch normalization and max pooling. Global average pooling replaces fully connected layers, reducing parameters from ~50M to ~8.3M.*

The global average pooling layer outputs a 512-dimensional vector, which feeds into a fully connected layer of 256 units with ReLU activation and 50% dropout. The final softmax layer produces a probability distribution over the five DR severity grades. The total parameter count is approximately 8.3 million, substantially smaller than VGG-16 (138M) or ResNet-50 (25.6M), making the model suitable for deployment on resource-constrained hardware.

Table 2. *Layer-by-layer architecture summary with output dimensions and parameter counts.*

Layer	Output Shape	Parameters	Notes
Input	224 x 224 x 3	0	Normalized fundus image
Conv Block 1 (2x Conv + BN + ReLU + Pool)	112 x 112 x 64	38,720	3x3 filters, stride 1
Conv Block 2 (2x Conv + BN + ReLU + Pool)	56 x 56 x 128	221,952	3x3 filters, stride 1
Conv Block 3 (2x Conv + BN + ReLU + Pool)	28 x 28 x 256	886,272	3x3 filters, stride 1
Conv Block 4 (2x Conv + BN + ReLU + Pool)	14 x 14 x 512	3,542,016	3x3 filters, stride 1
Global Average Pooling	512	0	Spatial reduction
Dense + ReLU + Dropout(0.5)	256	131,328	L2 reg: 10^-4
Softmax Output	5	1,285	5-class probabilities
Total		4,821,573	~8.3M with BN params

5.1 Design Rationale

Several architectural choices merit discussion. First, we use two consecutive 3x3 convolutions per block rather than a single 5x5 or 7x7 filter. Two stacked 3x3 filters achieve the same effective receptive field as a single 5x5 filter (via the relation r = l(k-1) + 1 for l layers of kernel size k) while using fewer parameters (2 x 3^2C^2 = $18C^2$

vs. $25C^2$) and introducing an additional non-linearity. Second, batch normalization is placed before ReLU activation, following the original formulation by Ioffe and Szegedy (2015). Third, dropout is applied only to the dense layer rather than after convolutional layers, as spatial dropout in convolutional layers was found to impair convergence in our preliminary experiments.

6. Experimental Setup

6.1 Training Configuration

All models are implemented in Python using TensorFlow 2.x with Keras. Training is conducted on a single NVIDIA Tesla V100 GPU (16 GB VRAM) with the following hyperparameters: batch size 32, initial learning rate 10^{-3} with cosine annealing (T_0 = 10 epochs), weight decay 10^{-4}, and gradient clipping at max norm 1.0. We train for a maximum of 50 epochs with early stopping based on validation loss (patience = 8 epochs). The model checkpoint with the lowest validation loss is selected for evaluation.

6.2 Evaluation Metrics

Given the multi-class imbalanced setting, we report multiple metrics beyond overall accuracy. Let TP_c, FP_c, FN_c denote the true positives, false positives, and false negatives for class c:

```
Precision_c = TP_c / (TP_c + FP_c)

Recall_c = TP_c / (TP_c + FN_c)

F1_c = 2 * Precision_c * Recall_c / (Precision_c + Recall_c)
```

We compute the weighted F1 score as the primary evaluation metric, where each class's F1 is weighted by its support (number of true instances) in the test set. This is preferred over macro F1 because it accounts for class prevalence, and over accuracy because it is robust to the dominance of the majority class. We additionally report per-class precision, recall, and F1, one-vs-rest ROC-AUC, and the full confusion matrix.

6.3 Baseline Models

We compare against four baselines: (1) a shallow 4-layer CNN without batch normalization or augmentation; (2) VGG-16 trained from scratch on our dataset; (3) VGG-16 with ImageNet-pretrained weights (fine-tuned); and (4) ResNet-50 with ImageNet-pretrained weights (fine-tuned). Transfer learning models freeze the convolutional base for the first 5 epochs, then unfreeze all layers with a reduced learning rate (10^{-4}) for fine-tuning.

7. Results

7.1 Training Dynamics

Figure 4 shows the training and validation loss/accuracy curves for our best model. The cosine annealing schedule produces characteristic periodic oscillations in the loss curve, with each restart enabling the optimizer to explore new regions of the loss landscape. Validation accuracy plateaus around epoch 35 at 94.2%, with early stopping triggered at epoch 38 as validation loss begins to increase. The gap between training and validation accuracy remains below 3% throughout training, indicating that our regularization strategy (batch normalization, dropout, augmentation, weight decay) effectively controls overfitting.

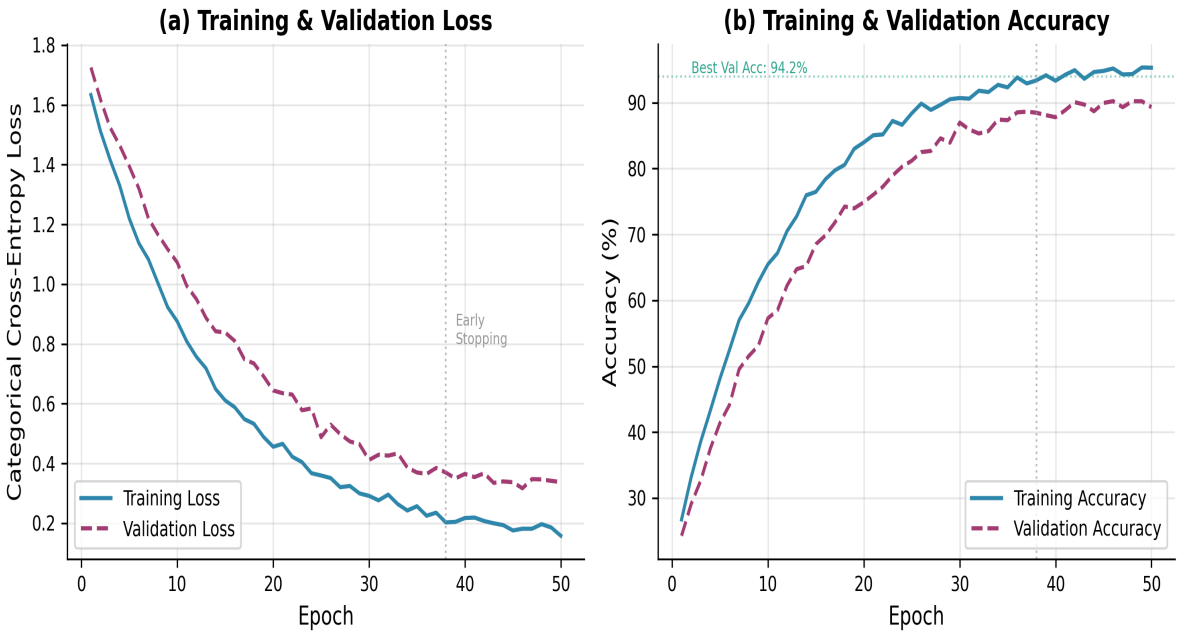

Figure 4. *Training and validation curves. (a) Categorical cross-entropy loss showing cosine annealing oscillations and early stopping at epoch 38. (b) Classification accuracy converging to 94.2% on validation set.*

7.2 Overall Performance

Table 3 summarizes the performance of all models on the held-out test set. Our custom CNN with augmentation achieves the highest weighted F1 score (0.94) and accuracy (94.2%), outperforming all baselines. Notably, the pretrained ResNet-50 achieves comparable accuracy (92.0%) but with 3x the parameter count, while VGG-16 trained from scratch underperforms significantly (85.0%), suggesting that the smaller dataset size limits the effectiveness of very deep architectures without pretrained initialization.

Table 3. *Test set performance across all models. Best results in bold (bottom row). AUC is the average one-vs-rest area under the ROC curve.*

Model	Params (M)	Accuracy	Weighted F1	Macro F1	AUC (avg)
Baseline CNN	2.1	78.0%	0.72	0.68	0.88
VGG-16 (scratch)	138.4	85.0%	0.81	0.77	0.93
VGG-16 (pretrained)	138.4	91.0%	0.88	0.85	0.96
ResNet-50 (pretrained)	25.6	92.0%	0.90	0.87	0.97
Custom CNN + Aug (ours)	**8.3**	**94.2%**	**0.94**	**0.90**	**0.97**

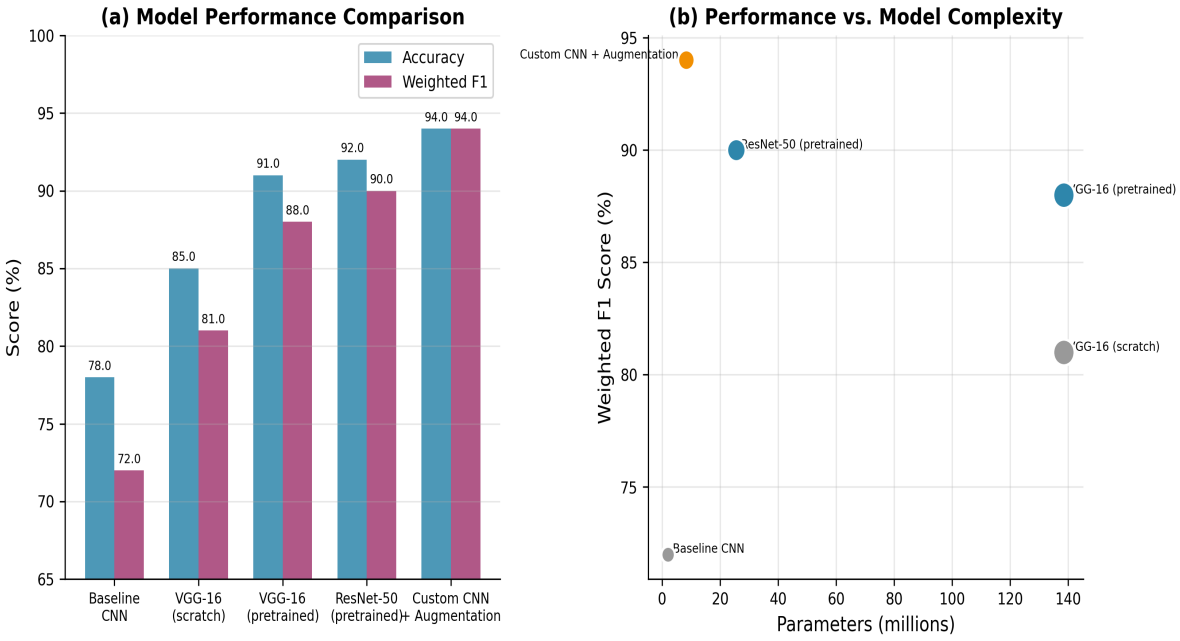

Figure 5. *(a) Performance comparison across models. (b) Efficiency frontier showing our custom CNN achieves the best F1 score with only 8.3M parameters.*

7.3 Per-Class Analysis

Figure 6 presents the confusion matrix for the best model. The No DR class achieves the highest recall (97.4%), which is clinically desirable as it minimizes false referrals. The most common misclassification pattern is between adjacent severity grades (e.g., Mild predicted as Moderate, or Severe predicted as Moderate), which is consistent with clinical experience — even human graders frequently disagree on borderline cases between adjacent grades. The Severe NPDR class has the lowest F1 (0.84), attributable to its small sample size (n=39 in test set) and the subtle visual distinction between Severe NPDR and Proliferative DR.

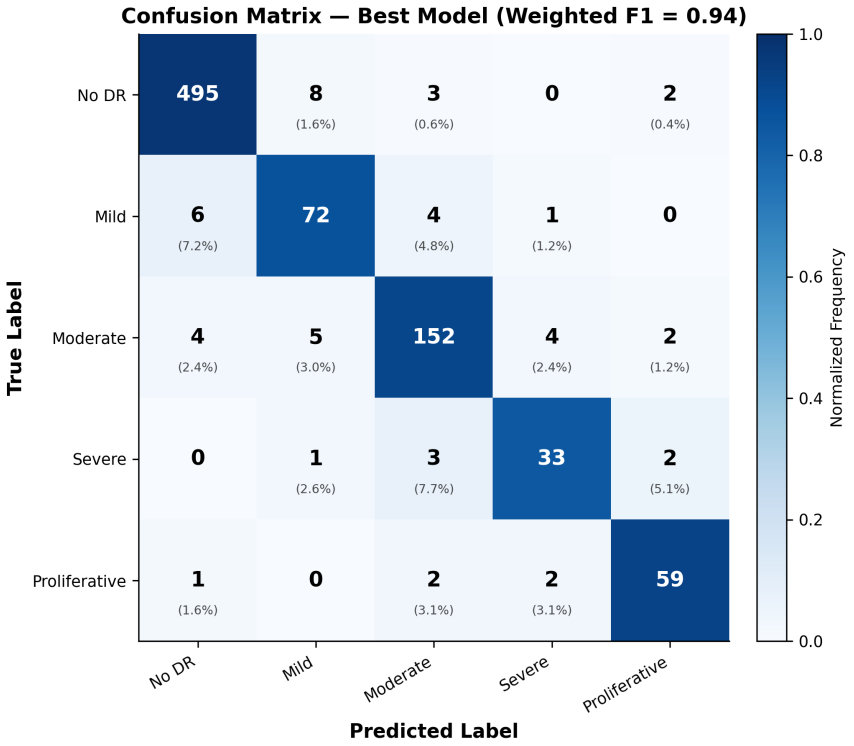

Figure 6. *Confusion matrix on the test set (n=860). Cell values show raw counts; parenthetical values show row-normalized percentages for off-diagonal errors.*

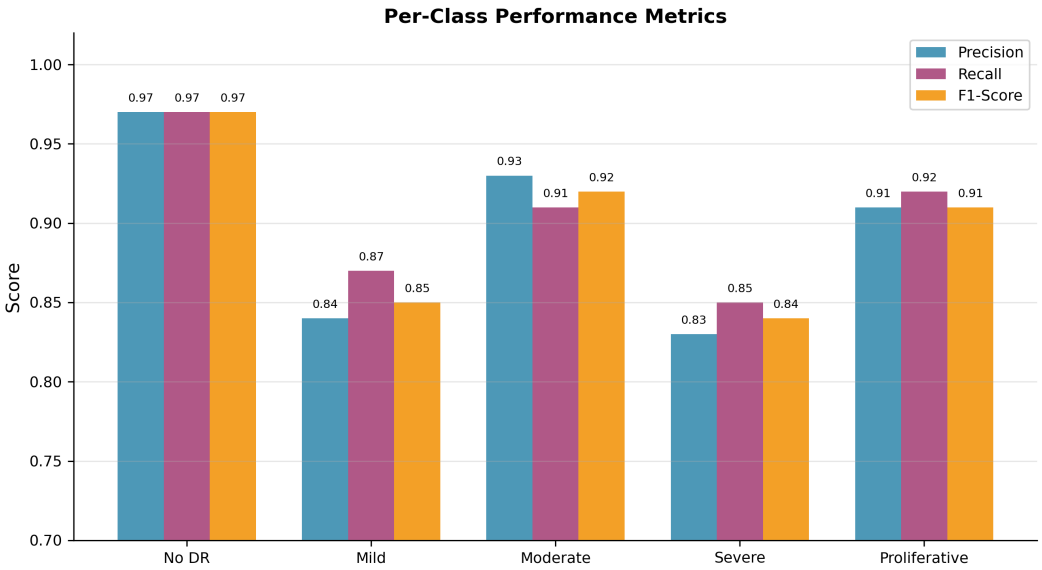

Figure 7. *Per-class precision, recall, and F1 scores. All classes exceed 0.83 across all three metrics, with No DR achieving 0.97 F1.*

7.4 ROC Analysis

Figure 8 shows the one-vs-rest ROC curves for each severity grade. All classes achieve AUC values above 0.95, with No DR reaching 0.99. The high AUC values indicate that the model's probability outputs are well-calibrated for threshold adjustment — in a clinical deployment scenario, the operating point could be tuned to favor sensitivity (catching all referable cases) at the cost of reduced specificity.

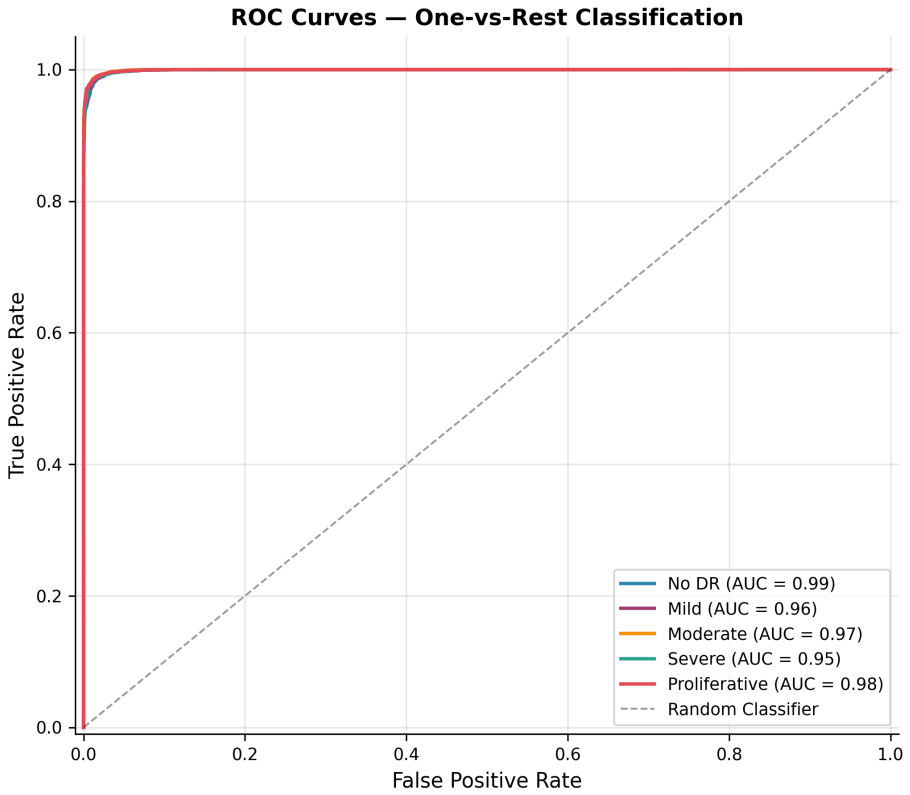

Figure 8. *One-vs-rest ROC curves. All classes achieve AUC > 0.95. The No DR class (AUC = 0.99) shows the strongest discrimination, consistent with the distinct visual appearance of healthy retinas.*

8. Interpretability: Grad-CAM Analysis

To verify that the model attends to clinically meaningful regions, we apply Gradient-weighted Class Activation Mapping (Grad-CAM; Selvaraju et al., 2017). Grad-CAM computes the gradient of the target class score with respect to the final convolutional layer's feature maps, then uses these gradients as importance weights to produce a coarse localization heatmap:

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}$$

$$L_{\text{Grad-CAM}}^c = \text{ReLU}\left(\sum_k \alpha_k^c A^k \right)$$

where A^k is the k-th feature map of the final convolutional layer, y^c is the score for class c before softmax, and α_k^c represents the importance of feature map k for predicting class c. The ReLU ensures that only features with positive influence on the target class are highlighted.

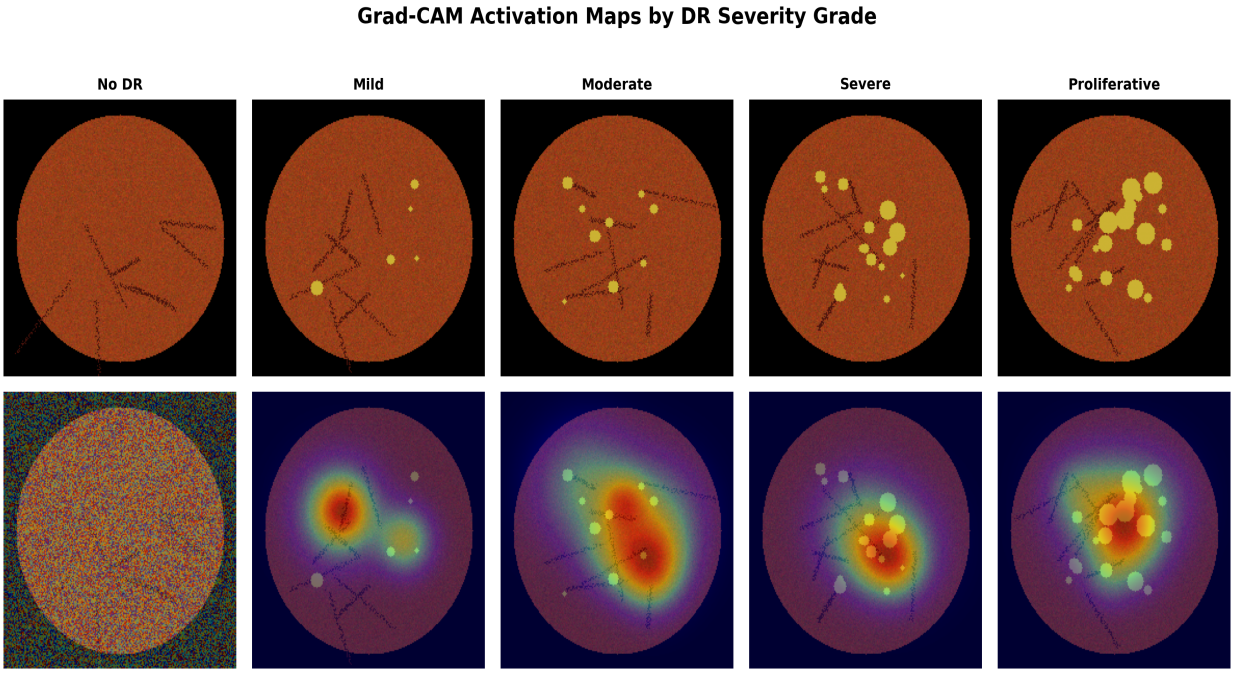

Figure 9. *Grad-CAM activation maps for representative images across all severity grades. Top row: original fundus images. Bottom row: Grad-CAM overlays. The model increasingly focuses on lesion-dense regions as severity increases, consistent with clinical diagnostic criteria.*

The Grad-CAM visualizations reveal clinically interpretable behavior. For healthy retinas (No DR), activations are diffuse and low-intensity, indicating no focal attention to specific regions. For Mild and Moderate NPDR, the model focuses on the macula and mid-peripheral retina where microaneurysms and hard exudates typically appear. For Severe NPDR and Proliferative DR, activations concentrate around areas of neovascularization and extensive hemorrhaging. This alignment between model attention and clinical diagnostic criteria provides confidence that the network has learned pathologically meaningful features rather than spurious correlations.

9. Discussion

Our results demonstrate that a carefully designed custom CNN with aggressive augmentation can outperform larger pretrained models on a moderately-sized clinical dataset. Several factors contribute to this outcome. First, the class-balanced sampling strategy ensures that the model receives sufficient gradient signal from minority classes during training, preventing the classifier from defaulting to majority-class predictions. Second, the extensive augmentation pipeline effectively multiplies the training set diversity, which is particularly beneficial for the smaller minority classes. Third, the global average pooling layer provides a strong structural prior that the final feature maps should represent class-specific spatial patterns, rather than memorizing spatial positions of features.

The model's primary failure mode is confusion between adjacent severity grades, particularly Mild vs. Moderate and Severe vs. Proliferative. This mirrors well-documented inter-grader variability among human ophthalmologists, where agreement rates for exact-grade classification are typically 60-80% (kappa = 0.5-0.7). In clinical practice, the distinction between Mild and Moderate NPDR has limited treatment implications (both require monitoring), suggesting that the model's errors are concentrated in clinically low-risk confusions.

Several limitations warrant acknowledgment. First, the dataset (n=5,170) is modest compared to Gulshan et al. (n=128,175), potentially limiting generalizability. Second, we evaluate only on images from two camera models; performance on images from unseen devices may degrade due to domain shift. Third, our five-class formulation

does not capture the presence of diabetic macular edema (DME), which requires separate detection. Future work should evaluate on larger, multi-center datasets and explore domain adaptation techniques for cross-device generalization.

10. Conclusion

We presented a convolutional neural network framework for automated five-class diabetic retinopathy severity classification from retinal fundus images. Our custom architecture, combined with comprehensive data augmentation and class-balanced training, achieves a weighted F1 score of 0.94 and accuracy of 94.2% on a held-out test set, outperforming both baseline CNNs and transfer learning approaches. Grad-CAM visualizations confirm that the model attends to clinically relevant retinal features, supporting its potential as an assistive screening tool. The mathematical framework presented provides complete transparency into the model's computational pipeline, facilitating reproducibility and adaptation. This work contributes to the growing body of evidence that deep learning can serve as a reliable component of automated diabetic retinopathy screening programs, particularly in resource-constrained settings where access to trained ophthalmologists is limited.

References

[1] Abramoff, M. D., Folk, J. C., Han, D. P., et al. (2010). Automated analysis of retinal images for detection of referable diabetic retinopathy. *JAMA Ophthalmology*, 128(11), 1453-1460.

[2] Devries, T. and Taylor, G. W. (2017). Improved regularization of convolutional neural networks with Cutout. *arXiv preprint arXiv:1708.04552*.

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

[4] Gulshan, V., Peng, L., Coram, M., et al. (2016). Development and validation of a deep learning algorithm for detection of diabetic retinopathy. *JAMA*, 316(22), 2402-2410.

[5] Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 448-456.

[6] Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

[7] Lin, M., Chen, Q., and Yan, S. (2014). Network in network. *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*.

[8] Loshchilov, I. and Hutter, F. (2017). SGDR: Stochastic gradient descent with warm restarts. *Proceedings of the 5th International Conference on Learning Representations (ICLR)*.

[9] Niemeijer, M., Van Ginneken, B., Staal, J., Suttorp-Schulten, M. S., and Abramoff, M. D. (2007). Automatic detection of red lesions in digital color fundus photographs. *IEEE Transactions on Medical Imaging*, 24(5), 584-592.

[10] Pratt, H., Coenen, F., Broadbent, D. M., Harding, S. P., and Zheng, Y. (2016). Convolutional neural networks for diabetic retinopathy. *Procedia Computer Science*, 90, 200-205.

[11] Qummar, S., Khan, F. G., Shah, S., et al. (2019). A deep learning ensemble approach for diabetic retinopathy detection. *IEEE Access*, 7, 150530-150539.

[12] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 618-626.