Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 6, 2026

IDENTIFICATION OF DIABETIC DISEASES THROUGH RETINAL SCAN BY USING CONVOLUTIONAL NEURAL NETWORK

*[1]Jawad Akbar Maitlo, [2]Dr. Zahid Ali, [3]Tariq Ali, [4]Asma Imam Somro
*[1]Department of Computer Sciences. ILMA University, Karachi, Sindh Pakistan.
[2]Department of Computer Sciences. ILMA University, Karachi, Sindh Pakistan.
[3]Department of Computer Sciences. ILMA University, Karachi, Sindh Pakistan.
[4]Department of Computer Sciences. ILMA University, Karachi, Sindh Pakistan.

DOI:- https://doi.org/10.5281/zenodo.21106411

Keywords
Diabetic disease, Retinal illness, eyes diseases Computer vision, Deep learning, QW kappa metric, Deterioration

Article History
Received: 25 May, 2026
Accepted: 29 June, 2026
Published: 30 June, 2026

Corresponding Author: *
Jawad Akbar Maitlo

Abstract
Finding the diabetic disease using naked eye is crucial and intervention for human at early stage is important for early clinical and interventions according to medical treatment. To safely visualize these conditions, optical coherence tomography (OCT) is widely preferred as a non-invasive, non-contact imaging modality. Given the widespread shortage of specialized diagnostic technology and clinical support. To address this need, we developed a parameterized, lightweight framework that fuses a CNN and a transformer for multi-class retinal disease classification. By leveraging this hybrid design, the CNN captures fine-grained local lessions while the transformer encoder models long range dependencies across the entire OCT image, significantly boosting diagnostic sensitivity. This pipeline is further reinforced by a specialized convolutional block designed to maximize feature representation with a low parameter footprint. We evaluated our proposed framework against several bassline architectures. On the OCT-c8 dataset, our model achieved the highest accuracy score of 0.9800 alongside competitive recall, while simultaneously utilizing the fewest parameters and requiring the shortest pre-image inference latency. Furthermore, evaluation on the broader OCT2017 dataset demonstrated that our model outperforms on four stage, recent state of the art architecture and matches the performance of a fifth, achieving a remarkable average accuracy, precision, recall, specificity, and F1-score of 0.9985, 0.9970, 0.9990, and 0.9970, respectively. These performance metrics were achieved with a highly compact footprint. The model requires only 1.28 million parameters, enabling a rapid average processing speed of 2.5 milliseconds per image scan.

Introduction

a. Diabetic Retinopathy

Diabetic retinopathy (DR) is an unadorned visual complication of diabetes mellitus characterized by progressive damage to the light sensitive retinal tissue lining the posterior of the eye, which can ultimately lead to irreversible blindness. High-impact medical data reveals a striking truth for identification of retinal disease, up to 90% for emergent diabetic retinopathy cases are entirely avoidable through pro-active diagnostic screen and vigilant retinal monitoring. The primary risk vector scales temporally, that mean the patient is considering diabetic illness and increase the number of patient throughout the time. When evaluate the patient medical expert map the progress according to the five tier ordinal severity index, starting stage is 0 for no apparent of retinopathy, shifting sequentially through 1, 2, 3, and 4 stage 01 mild, stage 02 moderate, stage 03 severe state called non-proliferative state, before we considering the peaking at proliferative called stage 04. Beneath the lens, medical signature drive this stages contain mcroaneurysms, active micro vascular haemorrhaging, muscular edema, and abnormal neovascularization, and localized nerve fiber layer degradation [7]. Designed the automated system to track and identify the illness at early stage make useful for medical revolutions in the real time immensely chanllanging due to the operational constraints of medical eye images, because of manual human intervention suffer from multi-day turnaround baclogs, patient routinely face fragment windows that accelerate permanent vision disorder. Medical expert successfully ranking the diseases by mapping with lesions and vascular irregularities, yet manual finding imposes an unsustainably high resource burden. Specialized training and hardware resources, which are essential pre-conditions, are unavailable in underdeveloped areas. In high prevalence and unserved communities. An automated system with high screening networks are vital rule in global blindness rate through diabetic disease. Traditional system workflows often require a days to processed, this cause leading to missed follow-ups, communication gaps, and therapeutic delays that compound and irreversible visual detriment. While manual identify by expert is highly accurate, its operational demands severely limit medical throughput. Specialized medical expertise and diagnostic machines are not fairly distributed make the shortage of medical equipment in region, it should be share on the basis of diabetic patients to avoid any inconvenience and making rapid, automated screen system to important rule during the pandemic situation.

From the past few decades, computer vision and computer aided technology have reduce this gap, leveraging machines learning utilizing deep learning to execute pattern recognition tasks over large retinal cohorts [7]. Early detection in this domain depends on rule-based engineering, using explicit mathematical notations to isolate lesions and microaneurysms, in recent, deep learning architecture have many features toward automatic binary classification, scanning the healthy and effective categories [14, 15, 16].

During the research we introduces a unified, end to end deep learning technique and architecture optimized for multiclass diabetic staging we define earlier by bypassing manual derivation or sourcing of features, the proposed model and algorithm architecture autonomously extracts and prioritizes the layered deep structural representations vital exact identification.

Fig 1: Normal human vision [7]

Fig 2: Human Vision with diabetics [7]

B. Convolutional Neural Networks

Convolutional neural network have precise and better engagement to success rate using machine learning algorithm for large scale images recognition [1]. The rapid achievement using CNN has been accelerate by the advent of massive public images repository, as like ImageNet and other datasets, using high-performance computing hardware GPUs devices and machines. In particular, the ImageNet is the large Scale pictorial recognition challenge with ILSVRC has played a decisive role in proceeding computer vision algorithms. CNN has functioned as a standard for algorithmic efficiency for classification of images frameworks, driving the transition from high-dimensional, superficial features extraction to deep convolutional architecture [3].

Table no. 1: Class Distribution

Class	Name	No. of Images	Percentage
0	No DR	25810	73.48%
1	Mild DR	2443	6.96%
2	Moderate DR	5292	15.07%
3	Severe DR	873	2.48%
4	Proliferative DR	708	2.01%

The above table shows the class division for different classes in datasets, the fragment accidentally dropped in the middle of table as detailing the simonyan & Zisserman paper (2014) due to the large number of repositories.

By structural combination of layers, CNN contains more than one convolutional layers which interlaced with down sampling method or pooling operations followed more than one interconnected conv layers skip to standard multiplayer perceptions. The covo architecture is intentionally designed to divide the two-dimensional hierarchical structure of input images or three-dimensional signals using technique that achieved localized

receptive fields and weight-sharing mechanisms, which are subsequently processed by pooling operations that summarize local activations to yield translation-invarient features. Furthermore, CNN exhibit remarkable parameter efficiency, rendering them less computationally intensive and easier to train than fully connected networks with equivalent hidden units [6]. By stacking multiple processing layers, CNN seamlessly learn hierarchical representations during the training phase, enabling the simultaneous extraction of low-level primitive edges and high-level semantic features. [2]

Convolutional networks ConvNets have achieved widespread success in large scale image and video recognition tasks (Krizhevsky et al., 2012 [1]; and Zeiler & Fergus, 2013; Semanet et al., 2013).

Dataset

The dataset utilized in this study comprises 35,126 high-resolution, color funds retinal images, each labelled into one of five distinct classes corresponding to the progressive severity stages of diabetic retinopathy outlined in table 1. The associated test set consists of 53,376 images were obtained from the opem-source EyePACS respository, a digital platform dedicated to accessible retinal image and scored the severity of the pathology on an ordinal ranging from 0 to 4 [5]. The dataset for featured images from varied model and different type of camera, resulting in inconsistencies in visual appearance and orientation. Rental images are presented in two ways, anatomically corrected or inverted, as viewed through a microscope condensing lens. Furthermore, the dataset contains noise, varying resolution and images with artefacts, poor focus, or improper exposure in outline table no 1, the distribution of images classes.

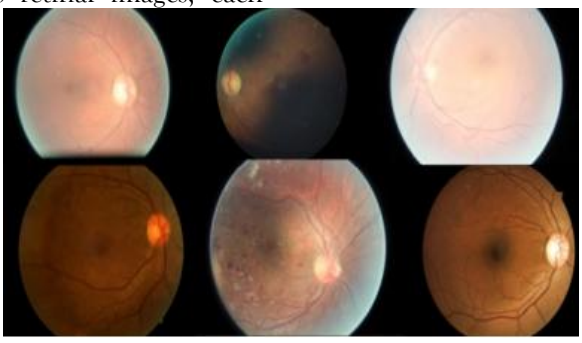

Figure no. 3: Dataset Images sample

Data Pre-Processing

We standardize the images during training session to 512 pixels by 512 pixels, due to non-standard inherent to the raw dataset and image resolutions, we could not utilized the images directly to get the result. Consequently, all fundus photographs were down sampled to a fixed resolution of 512 x 512 pixels. We scaled down the images to fix the resolution size to form a standardized dataset for prediction. The all images which scaled 512 pixels by 512 pixels for all three colour channels processing tri-channel RGB matrics at this resolution imposed severe memory bottlenecks, required high memory requirements. Due to inconvenience, the images must converted to a single channel extraction strategy was implemented rather than converting the images to grayscale blindly. After multiple attempt, we found that the green channel images work properly rather than other images taken information better than the other channel images. This spectrum preserves the highest density of pathological information, and retinal hemorrhages and microaneurysms display peak contrast against the background stroma in this band. Following channel isolation, contrast limited histogram equalization was executed to mitigate illumination was across pixels. To conclude the pre-processing pipeline, min-max normalization was spplied to map pixel intensities uniformly, effectively dampening background artifacts and preventing the CNN from learning non-functional noise signatures.

Figure no. 4: Original Image

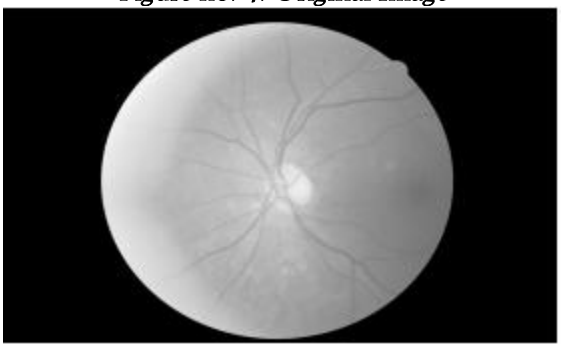

Figure no. 5: Green Channel Image

Artificial Neural Network Architecture

In deep learning, the multiple architecture for deep network is shown in chart no 2. That network with 5 consecutive sets of neural network combinations is specifically designed for features extraction with higher accuracy, for higher resolution 512 x 512 pixel image, stacking five sets of these combinations.

Figure no. 6: Histogram image Equalization with green channel

The extracted representation are routed through pooling layers followed by two sequential fully connected hidden layers, which map the features to the final classification output. To preserve the spatial dimensions of the intermediate feature maps throughout the network, zero-padding was systematically applied within all convolutional layers.

Convolutional Layer

Within the combinations of current network are stacked consecutively without interleaved spatial pooling. According to Andrew and Zisserman et al. [3], this design choice expend the effective receptive field and enhances the discriminative capacity of the decision function while simultaneously reducing the number of learnable parameters, thereby enabling greater network depth information. The very first convolutional layer comprises 16 filters of size 5 by 5 size operating with stride of 2. The third layer of convolutional layer have 32 filters with 3 by 3 size, which is convolve over the preceding features mapping, also with a stride of 2.the configuration of all remaining convo neural network is detail in chat no 2.

Pooling Layer

The pooling layer uses a 2 by 2 filters for max pooling, which computes the maximum value within each local regional basis across the input depth slice. The down-sampling processing the features mapping by taking the maximum value for every 2 by 2 area, preserving down-samples preservative the most significant activations.

Dropout Layer

The role of dropouts in neural network to mitigate over-fitting and enhanced supervised learning performance, we employ dropout as proposed by Srivastava et al. [10]. The dropout rate increase uniformly from 0.1 to 0.4 across the first four layers, and is held constant at 0.5 for the fifth and sixth layers.

Hidden layer and feature Pooling Layers

In the initial architecture the capacity of both fully connected hidden layers was set to 400 units. For the second and third models, this configuration was downscaled to 256 hidden units per layer to reduce model complexity. Each fully connected layer was regularization using a maxout activation function implemented via a feature pooling layer with a pool size of 2. Maxout is an activation function that takes the maximum across a set of affine feature maps, essentially pooling across channels (hidden units) rather than spatial pixels. It acts as a natural partner to dropout, offering optimization benefits and approximately model averaging. Applying this max pooling operation across the hidden units allows the network to learn piecewise linear approximations of arbitrary continues functions, thereby capturing highly discriminative features while inheriting the optimization and model averaging benefits of dropout regularization [9].

Table no. 2: *CNN Architectures of 3 different models*

Layers	Model 1	Model 2	Model 3
Input	1*512*512	1*512*512	1*512*512
Conv 1	16*256*256	16*256*256	16*256*256
Conv 2	16*256*256	16*256*256	16*256*256
Pool 1	16*128*128	16*128*128	16*128*128
Dropout 1	16*128*128	16*128*128	16*128*128
Conv 3	32*64*64	32*64*64	32*64*64
Conv 4	32*64*64	32*64*64	32*64*64
Pool 2	32*32*32	32*32*32	32*32*32
Dropout 2	32*32*32	32*32*32	32*32*32
Conv 5	48*32*32	64*32*32	64*32*32
Conv 6	48*32*32	64*32*32	64*32*32
Conv 7	48*32*32	64*32*32	64*32*32
Pool 3	48*16*16	64*16*16	64*16*16
Dropout 3	48*16*16	64*16*16	64*16*16
Conv 8	64*16*16	128*16*16	96*16*16
Conv 9	64*16*16	128*16*16	96*16*16
Conv 10	64*16*16	128*16*16	96*16*16
Pool 4	64*8*8	128*8*8	96*8*8
Dropout 4	64*8*8	128*8*8	96*8*8
Conv 11	128*8*8	256*8*8	128*8*8
Conv 12	128*8*8	256*8*8	128*8*8
Pool 5	128*4*4	256*4*4	128*4*4
Dropout 5	128*4*4	256*4*4	128*4*4
Hidden 1	400	256	256
Maxout 1	200	128	128
Dropout 6	200	128	128
Hidden 2	400	256	256
Maxout 2	200	128	128
Output	5	5	5

Activation Function

The convolutional and hidden layers utilized a leaky rectified linear unit (Leaky ReLU) activation function. The convolutional and hidden layers utilized a leaky rectifier activation function we compared to traditional activation rectifier maintain a non-zero gradients for negative inputs, which often improves convergence [11]. For classification, the output layer employed a five way for activation function like softmax activation function to produce a probability distribution across the five target classes.

Background

Maximizing the diagnostic efficacy of the proposed deep neural network necessitated an architecture capable of maping highly intricate pathological features. However, optimizing a highly capable modelon the baseline dataset posed an immediate risk of over fitting. To broaden the network boundaries, we implemented an aggressive data augmentation pipeline utilizing diverse geometric transformations, specifically includeing coordinate sharing, horizontal reflections, and make matrix transpositions.

Table No. 3: *Class Distribution of dataset images*

Class	Name	No. of images	Percentage
0	No DR	25810	35.65%
1	Mild DR	12215	16.87%
2	Moderate DR	26460	36.55%
3	Severe DR	4365	6.02%
4	Proliferative DR	3540	4.89%

Initial exploratory data analysis revealed a pronounced class skew, which threatened to induce severe majority-class distribution toward Class 0, and Class 2, while the other-hand suppressing the network capacity to learn minority-class during training. We mitigated this systemic imbalance by selecting down sampling for newly instances within class 0, this technique creating a more uniformly distribution of dataset. During the intervention of this process we generally getting the dataset for training set at 72,390 describe in table no. 3. Though the resulting distribution is not perfectly symmetrical, it offers a dramatic structural improvement over the original state. This improve the final result of kappa coefficient.

Initiating the training process required optimizing memory allocation across both system ram and gpu respectively, it is important to make the batches for images during training session, so the batch denotes the number of images that can be readable by algorithm for training process. Other-hand, the mini-batches represent the the size of images that improve the ram and gpu performance that transferred from ram to gpu for further computations. After evaluating various architectural parameters like as, stride, filters size, activation function, image dimensions et cetera. We found that the optimal size of batch is 64 is ideal for network. That result produce the batch size include 2413 images. To address the multi-class identification. For this multi-classification we add softmax as activation function during output unit that is able to produce categorical entropy loss function that choose the entropy between classes and target values.

To minimize the object functions we trained the model for 250 epochs using stochastic gradient descent incorporating Nesterov Momentum. This gradient update the approach follow the framework established by [12]. The batch size is according to the mini as we discussed earlier. The number of epoch for training is 250. Mathematically, we looked ahead ahead velocity and weight adjustment the equation describe the follow as:

$$V = (M \times V) - (\text{learning rate} * \text{gradient}),$$

$$\text{Weight} = \text{Weight} + \text{Velocity}$$

V for velocity, M for momentum, In the above rule, the momentum parameter is set to 0.9 during the training session, we maintained the values as mitigate stochastic noise introduce order to variations in batches, the result depends on the number of epochs was employed rather than fixed the learning rate. Learning rate schedule was 0.003 for the first 200 epochs and 0.0003 for the remaining 50 epochs. Total epoch consider is 250.

Model weights and biases were initialized using a Glorot uniform distribution. This specific initialization strategy facilitates accelerated convergence of the objective function by stabilizing using early-stage gradient variances. [13]. Deep learning neural network are inherently prone to over fitting, multiple regularization techniques were deployed simultaneously. In tandem with the previously described data augmentation pipeline, dropout layers were integrated into each layer of the network structure according to the specific ratios outlined in the architectural designed.

Implementation

Data pre-processing workflows were executed using a combination of the ImageMagick command-line

utility and the OpenCV python library. To accelerate the training phase of the convolutional nueral network, computation were offloaded to an nvidia gtx 780 series gpu, support by 8th generation system ddr4 ram system, hardware interfacing and gpu acceleration were managed via the theano library [17, 18]. Used. For the network deployment architectural design and model training routines were streamlined using the Lasange [19] python packages.

Results

The evaluating this model required a performance metric that extends beyond a binary count of correct and incorrect test image classification. Crucially, the evaluation framework needed to quantify the distance or magnitude of misclassifications across ordinal classes and penalize the performance score proportionally. Consequently, model performance was evaluated using the quadratic weighted kappa (QWK) metric, which quantifies the degree of agreement between two sets of ratings. The QWK score theoretically ranges from 0, indicating agreement equivalent to random chance, to 1. Representing agreement between raters. Conversely, if the metric yields a negative value. Here, the quadratic eighted kappa is computed between the ground-truth labels assigned by human experts and the network prediction score [8].

The dataset images are classified using an ordinal scale across five distinct ratings, 0, 1, 2, 3, 4, and each sample is characterized by an evaluation tuple (e_a , e_b), where e_a represents the ground-truth label assigned by rater A (Human expert) and e_b denotes the predicted score generated by rater B (the model). The quadratic weighted kappa (k) is computed through a three-steep matrix formulation. First and N x N confusion matix O is constructed, where each element $O_{i,j}$ represents the total number of images that received a rating of i by rater A and a rating of j by rater B. second, an N x N weight matrix W is calculated to quantify the severity of disagreement between the two ratings

$$W_{i,j} = (i - j)^2 \text{ divided by } (N - 1)^2$$

Third, an N x N expected agreement matrix E is generated under the null hypothesis of completely uncorrelated ratings. This matrix is calculated as the outer product of the individual rater histogram vectors, normalized such that the sum of the individual rater histogram vectors, normalized such that sum of the elements in E equals the sum of the elements is O. utilizing these three matrices, the final quadratic weighted kappa coefficient is defined as:

$$k = 1 - \frac{\sum_{i,j} w_{i,j} * O_{i,j}}{\sum_{i,j} w_{i,j} * E_{i,j}}$$

Following the training phase outlined in table 2, model 1 was evaluated against the 5,000 images in the te4st set, yielding a quadratic weighted kappa score of 0.3066. This modest performance suggested that the extensive number of hidden units may have induced slight overfitting. To test this hidden units to 256. Concurrently, the number of feature maps in the subsequent convolution layers was increased to compensate for the reduction in fully connected parameters. To validate this assumption, model 2 was configuration with reduced hidden layer capacity of 256 units. To offset this reduction in dense layers, the filter count in the latter convolutional layer was systematically increased.

Training and testing model 2 we produced a quadratic kappa score of 0.35, making a considerable improvement over the baseline model. This confirmed that down-sampling the hidden layers enhanced generalization, however, the increased filter count appeared to introduce excessive architectural complexity. Consequently, the filter dimensions in the latter convolutional layers were scaled down in model 3. This refinement pushed the quadratic kappa score up to 0.3865, continuing the upward performance trend. Finally, rather than averaging out errors blindly, we implemented weighted ensemble averaging to integrate the distinct feature representation learned by each network. Empirical optimization of the ensemble weights yielded a top quadratic kappa score of 0.399, achieved with a combining ratio of 0.075 from model 1 and 0.125 from model 2 and 0.85 from model 3. the score of each model show below.

Table No. 4: *Model Comparison*

Model	Quadratic kappa score
Model 1	0.3066
Model 2	0.35
Model 3	0.386
Ensemble	0.3996

Table no. 4 provides an evaluation of the ensemble model by cross-referencing the ground-truth annotations of the test dataset against the predicted classifications for each category. Additionally, the matrix explicity quantifies the margin of error by detailing the exact number of images within each of the five classes that were misclassified by a distance of exactly one class.

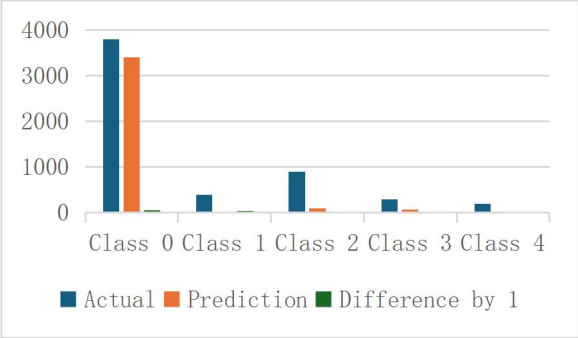

Figure no. 7: Model Performance Comparison

Conclusion

This study present the design, architecture and implementation of deep convolutional neural networks for automatic detection and classification of diabetic retinopathy from color funds retinal images. We have demonstrated the design and deployment of deep convolutional neural networks engineered to automate the detection and multi-class grading of diabetic retinopathy using color funds photographs. To assess model grading with human expert diagnostics, performance evaluations relied on the quadratic weighted kappa metric. Our multi-stage tuning workflow yielded three distinct CNN mode, each model offering unique architectural trade-off. Rather than relying on a single network, we implemented an ensemble paradigm to fuse the strengths of each model. This collective approach successfully suppressed individual classification errors, generating a top quadratic kappa score of 0.3996.

References

[1] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Network. Advances in Neural Information Processing System 25, 2012

[2] Y. LeCun, K. Kavukcuoglu, and C. Farabet. Convolutional networks and applications in vision. In Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on, pages 253256. IEEE, 2010.

[3] Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR 2015.

[4] D. Nouri(2014, Dec 17). Using Convolutional Neural Nets To Detect Facial Keypoints Tutorial [Online]. Available: http://danielnouri.org/notes/2014/12/17/using-convolutional-neuralnets-to-detect-facial-keypoints-tutorial/

[5] Diabeticretinopathy Image dataset [Online]. Available: https://www.kaggle.com/c/diabetic-retinopathy-detection/data

[6] Unsupervised Feature Learning and Deep Learning Tutorial by Stanford University [Online]. Available: http://ufldl.stanford.edu/tutorial/

[7] Diabetic retinopathy [Online]. https://en.wikipedia.org/wiki/Diabetic retinopathy

[8] Quadratic Weighted able Fig. 7: Visualization of Predictions: Kappa Metric Available [Online]. : Avail https://www.kaggle.com/c/diabetic-retinopathy detection/details/evaluation

[9] Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, Yoshua Bengio. Maxout Networks arXiv:1302.4389v4 [stat.ML] 20 Sep 2013

[10] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research 15 (2014).

[11] Andrew L. Maas, Awni Y. Hannun, Andrew Y. Ng. Rectifier Nonlin earities Improve Neural Network Acoustic Models. Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013.

[12] Ilya Sutskever1, James Martens, George Dahl, Geofrey Hinton. On the importance of initialization and momentum in deep learning. Pro ceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013.

[13] Xavier Glorot, Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. Proceedings of the 13th Interna tional Conference on Artificial Intelligence and Statistics (AISTATS) 2010. Volume 9 of JMLR.

[14] R.Priya, P. Aruna. Diagnosis of Diabetic Retinopathy using Machine Learning Techniques. ICTACT Journal on Soft Computing 2013.

[15] Gilbert Lim, Mong Li Lee, Wynne Hsu, Tien Yin Wong. Trans formed Representations for Convolutional Neural Networks in Dia betic Retinopathy Screening. Modern Artificial Intelligence for Health Analytics: Papers from the AAAI-14.

[16] Sohini Roychowdhury, Dara D. Koozekanani, Keshab K. Parhi. DREAM: Diabetic Retinopathy Analysis Using Machine Learning. IEEE Journal of Biomedical and Health Informatics, Vol. 18, No. 5, September 2014.

[17] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. Goodfellow, A. Bergeron, N. Bouchard, D. Warde-Farley and Y. Bengio. Theano: new features and speed improvements. NIPS 2012 deep learning workshop.

[18] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley and Y. Bengio. Theano: A CPU and GPU Math Expression Compiler. Proceedings of the Python for Scientific Computing Conference (SciPy) 2010. June 30-July 3, Austin, TX

[19] Zenodo (2015, Aug 13) Lasagne [Online]. Available: https://zenodo.org/record/27878#.Vc0Iz3UVhHx