

FC layer aims to perform higher-level reasoning by computing the class scores. Each neuron in this layer is connected to all neurons in the previous layer to generate global semantic information.

The last layer of CNN's is an output layer (O), here the Softmax operator is commonly used for classification tasks. The optimum parameters ( $\theta$ , a common notation for both  $w$  and  $b$ ) for a particular task can be determined by minimizing the loss function ( $L$ ) defined for the task. Mathematically, for  $N$  input-output relations  $\{(I^n, O^n); n \in [1, \dots, N]\}$  and corresponding labels  $G^n$  the loss can be derived as:

$$L = \frac{1}{N} \sum_{n=1}^N \ln(\theta; G^n, O^n) \quad (2)$$

Where  $N$  denotes the number of training images,  $I^n$ ,  $O^n$  and  $G^n$  correspond to  $n^{th}$  training image. Here, a critical challenge in training CNN's arises from the limited number of training samples as compared to the number of learnable parameters that need to be optimized for the task at hand. Recent studies have developed some key techniques to better train and optimize the deep models such as data augmentation, weight initialization, Stochastic Gradient Descent (SGD), batch normalization, shortcut connections, and regularization. For more understanding related to advances in CNN's, the reader is recommended to refer a paper by Gu et al. (2018).

The growing use of CNN's as the backbone of many visual tasks, ready for different purposes (such as segmentation, classification or localization) and the available data, has made architecture search a primary channel in solving the problem.

In this challenge, mainly for disease severity grading problem, participants either directly utilized existing variants of CNN's or ensembled them to demarcate the input image to one of the classes mentioned in Table 4. Several configurations and variants of CNN's are available in the literature; some of the most popular are AlexNet (Krizhevsky et al., 2012), VGG (Simonyan and Zisserman, 2014), GoogLeNet (Szegedy et al., 2015) and ResNet (He et al., 2016) due to their superior performance on different benchmarks for object recognition tasks. A typical trend with the evolution of these architectures is that the networks have gotten deeper, e.g., ResNet is about 19, 8 and 7 times deeper than AlexNet, VGGNet and GoogLeNet respectively. While the increasing depth improves feature representation and prediction performance, it also increases complexity, making it difficult to optimize and even becomes prone to overfitting. Further, the increasing number of layers (i.e., network depth) lead to vanishing gradient problems as a result of a large number of multiplication operations. Hence, many teams chose the DenseNet (Iandola et al., 2014) which connects each layer to every other layer in a feed-forward fashion, reducing the number of training parameters and alleviates the vanishing gradient problem. DenseNet exhibits  $\ell(\ell + 1)/2$  connections in  $\ell$  layer network, instead of only  $\ell$ , as in the networks mentioned above. This enables feature reuse throughout the network that leads to more compact internal representations and in turn, enhances its prediction accuracy. Another opted approach, Deep Layer Aggregation (DLA) structures (Yu et al., 2017), extends the "shallow" skip connections in DenseNet to incorporate more depth and sharing of the features. DLA uses two structures – iterative deep aggregation (IDA) and hierarchical deep aggregation (HDA) that iteratively and hierarchically fuse the feature hierarchies (i.e. semantic and spatial) to make networks work with better accuracy and fewer parameters. Recent Fully Convolutional Network (FCN) (Long et al., 2015) adapt and extend deep classification architectures (VGG and GoogLeNet) into fully convolutional networks and transfer their learned representations by fine-tuning to the segmentation task. It defines a skip architecture that combines semantic information from a deep, coarse layer with appearance in-

formation from a shallow, fine layer to produce accurate and detailed segmentations.

For lesion segmentation task, most of the participating teams exploit U-Net architecture (Ronneberger et al., 2015). The main idea in U-Net architecture is to supplement the usual contracting network through a symmetric expansive path by addition of successive layers, where upsampling (via deconvolution) is performed instead of the pooling operation. The upsampling part consists of a large number of feature channels, that allow the network to propagate context information to higher-resolution layers. The high-resolution features from the contracting path are merged with the upsampled output and fed to soft-max classifier for pixel-wise classification. This network works with very few training images and enables the seamless segmentation of high-resolution images by means of an overlap-tile strategy. Other similar architecture SegNet (Badrinarayanan et al., 2015) was opted by a team; it consists of an encoder and decoder network, where the encoder network is topologically identical to CONV layers in VGG16 and in which FC layer is replaced by a Softmax layer. Whereas, the decoder network comprises a hierarchy of decoders, one corresponding to each encoder. The decoder uses max-pooling indices for up-sampling its encoder input to produce sparse feature maps. Later, it convolves the sparse feature maps with a trainable filter bank to densify them. At last, decoder output is fed to a soft-max classifier for the generation of segmentation map. One team choose Mask R-CNN (He et al., 2017), a technique primarily based on a Region Proposal Network (RPN) that shares convolutional features of an entire image with the detection network, thus enabling region proposals to localize and further segment normal and abnormal structures in the retina. RPN is a fully convolutional network that contributes to concurrently predicting object bounds and "objectness" scores at each position.

Following subsections present the solutions designed by participating teams with respect to three sub-challenges. Table 6 summarizes data augmentation, normalization and preprocessing tasks performed by each team.

### 5.1. Sub-challenge – 1: Lesion segmentation



For a given image, this task seeks to get the probability of a pixel being a lesion (either MA, HE, EX or SE). Although different retinal lesions have distinct local features, for instance, MA, HE, EX, SE have a different shape, color and distribution characteristics, these lesions share similar global features. Hence, the majority of participating teams built a general framework that would be suitable for the segmentation of different lesions, summarized as follows:

#### 5.1.1. VRT (Jaemin Son et al.)

Son et al. modified U-Net (Ronneberger et al., 2015) in such a way that upsampling layers have the same number of feature maps with layers concatenated. It was based on the motivation that features in initial layers and upsampled layers are equally important to segmentation. Additionally, they adjusted the number of max-pooling so that the radius of the largest lesion spans a pixel in the coarsest layer. In case of EX and HE, max-pooling is done six times, whereas for SE and MA it is done four times and twice. Further, for dealing with MA's, they used inverse pixel shuffling to convert a  $1280 \times 1280 \times 3$  pixels image to  $640 \times 640 \times 12$  for network input and pixel shuffling (Shi et al., 2016) to convert  $640 \times 640 \times 4$  segmentation map into  $1280 \times 1280 \times 1$  pixels. Later, the pairs of a normalized fundus image and reference ground truths were fed to the network to generate segmentation result in the range [0, 1]. They used weighted binary cross entropy (Murphy, 2012) as loss

**Table 6**

Summary of data augmentation, normalization and pre-processing in the competing solutions. Where, RF, RR, RS, RT, RC represent random flip, rotation, scaling, translation and crop respectively.

Task	Team name	Data augmentation						Data normalization	Data preprocessing
		RF	RR	RS	RT	RC	Other		
Sub-challenge - 1	 VRT	✓	✓	✓	✓	✓	shear	✓	FOV cropping, division by 255 then mean subtraction
	 iFLYTEK	✓	✓	✓	✓	✓	×	✓	lesion patch extraction
	 PATech	✓	✓	×	✓	×	color <sup>a</sup>	✓	RGB to LUV, contrast adjustment
	 SDNU	✓	✓	×	×	×	×	–	–
	 SOONER	✓	✓	×	×	✓	×	✓	mean subtraction, lesion patch extraction
	 LzyUNCC	✓	×	×	×	✓	stochastic and photo-metric <sup>b</sup>	–	FOV cropping, image enhancement
	 SAIHST	✓	✓	×	×	×	×	✓	CLAHE, Gaussian smoothing
Sub-challenge - 2	 LzyUNCC	✓	×	×	×	✓	color <sup>a</sup> , stochastic and photo-metric <sup>b</sup>	–	FOV cropping, image enhancement
	 VRT	×	×	×	×	×	×	✓	mean subtraction
	 Mammoth	✓	✓	✓	✓	×	color	×	morphological opening and closing
	 AVASAVA	✓	×	×	×	✓	×	✓	intensity scaling
	 HarangiM1	×	×	×	×	×	×	✓	FOV cropping
	 HarangiM2	×	×	×	×	×	×	✓	–
Sub-challenge - 3	 DeepDR	×	×	×	×	✓	OD, fovea region	✓	FOV cropping, mean subtraction
	 VRT	✓	✓	✓	✓	✓	shear and cropped OD	✓	FOV cropping, contrast adjustment
	 ZJU-BII-SGEX	×	×	×	×	×	×	✓	FOV cropping
	 SDNU	✓	×	✓	×	×	×	–	–
	 IITkgpKLIV	✓	✓	×	×	×	×	✓	–
	 CBER	×	×	×	×	×	×	–	–

<sup>a</sup> Reference: Krizhevsky et al. (2012)

<sup>b</sup> Reference: Howard (2013)

function given by

$$L = \frac{1}{N} \sum_{n=1}^N \left[ -\alpha G^n \log O^n - (1 - G^n) \log(1 - O^n) \right] \quad (3)$$

where  $N$  denotes the number of the pairs in a batch,  $G^n$  and  $O^n$  represent true segmentation and predicted segmentation for  $n^{th}$  image. The value of  $\alpha$  was determined as follows:

$$\alpha = \frac{B_0^i}{\gamma F_1^i} \quad (4)$$

where  $B_0^n$  and  $F_1^n$  denote the number of background and foreground pixels in  $n^{th}$  image. Since background overwhelms foreground in lesion segmentation task, this loss function was designed to penalize false negatives in order to boost sensitivity, an important factor in detecting lesions. Also,  $\gamma$  was left as a hyper-parameter and chosen out of {0.25, 0.5, 1, 2, 4, 8, 16, 32, 64, 256, 512} to yield the highest AUPR on validation set. The final selected  $\gamma$  values for different lesions are summarized in Table 7.

They trained the network over 300 epochs using Adam optimizer (Kingma and Ba, 2014) with hyper-parameters of  $\beta_1 = 0.5$ ,  $\beta_2 = 0.999$  and learning rate of  $2e^{-4}$  until 250 epochs and  $2e^{-5}$  until the end. All implementation was done by Keras 2.0.8

**Table 7**

$\gamma$  values in Eq. (4).

EXs	SEs	HEs	MAAs
64	512	8	32

with tensorflow backend 1.4.0 using a server with 8 TITAN X (pascal). The source code is available at [https://bitbucket.org/woalsdnd/isbi\\_2018\\_fundus\\_challenge](https://bitbucket.org/woalsdnd/isbi_2018_fundus_challenge).

### 5.1.2. IFLYTEK-MIG (Fengyan Wang et al.)

Wang et al. proposed a novel cascaded CNN based approach for retinal lesion segmentation with U-Net (Ronneberger et al., 2015) as a base model. It consists of three stages, the first stage is a coarse segmentation model to get initial segmentation masks, then the second stage is a cascade classifier which was designed for false-positive reduction, at last, a fine segmentation model was used to refine results from previous stages. First stage model was trained using the patches of size  $256 \times 256$  pixels centered on a particular lesion amongst MA, HE or EX and  $320 \times 320$  pixels for SE, resulting in the coarse segmentation outcome. Results of the previous stage are coarse due to the fact that non-focus regions

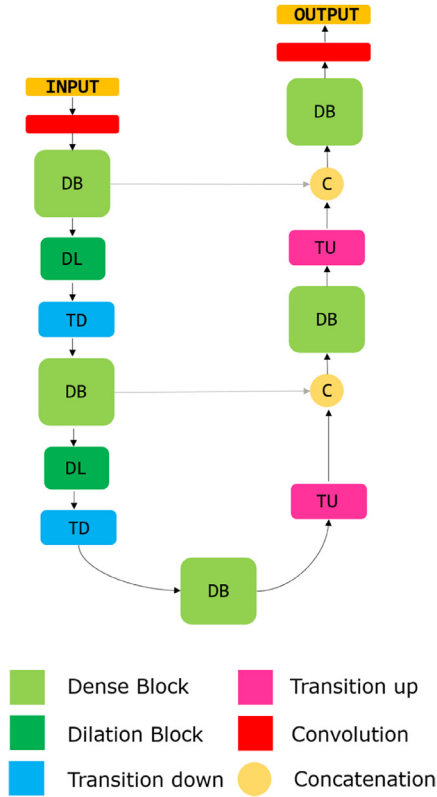


Fig. 4. Proposed architecture for lesion segmentation.

(non-target lesions) were not utilized in the learning process leading to high false-positive count. In the second stage, unlike the first segmentation model which used a lesion centered sample from input dataset pool, candidate regions were extracted using probability maps from the previous stage. Here, the input size fed to model for SE was  $320 \times 320 \times 3$  pixels, for HE and EX it was  $256 \times 256 \times 3$  pixels, and for MA it was modified to  $80 \times 80 \times 3$  pixels considering its small appearance. In this step, a candidate region was regarded as a positive sample if its intersection-over-union with the ground truth was greater than the given threshold (i.e. 0.5). In this way, most trivial non-focus regions were effectively rejected. However, it was identified in the test that a small proportion of false positives still exist, so an additional model was introduced to refine the segmentation results. In the last stage, candidate regions survived from the second stage were utilized as the input patches resulting in more accurate segmentation results. For the first and third stage, they used binary cross-entropy or dice loss function (multi-model training), whereas, for the second stage, they used only binary cross-entropy as a loss function. The first, second and third stage models were trained for 100, 300 and 100 epochs respectively with the momentum of 0.9. In which, the initial learning rate for the first and third stage was set 0.1 and is reduced by 10 times every 30 epochs, and for the second stage it was set to 0.001 reduced by 10 times every 80 epochs. MXNET platform was used for training the models.

### 5.1.3. PATech (Liu lihong et al.)

Lihong et al. developed a novel patch-based CNN model (as shown in Fig. 4) in which they innovatively combined the DenseNets (Iandola et al., 2014) and dilation block with U-Net (Ronneberger et al., 2015) to capture more context information and multi-scale features.

The model is composed of a down-sampling path with 4 Transitions Down (TD), 4 Dilation Block (DL) and an up-sampling path

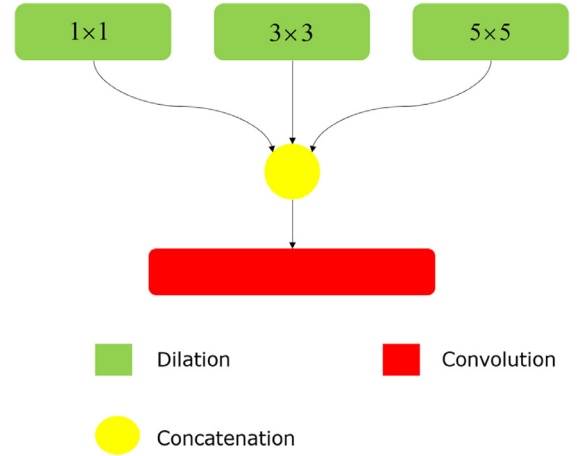


Fig. 5. Architecture for dilation block.

with 4 Transitions Up (TU). To capture multi-scale features, DL (see Fig. 5) is used with dilation rate of 1, 3 and 5 are concatenated for the convolution. The dense block (DB) is constructed by four layers. The idea behind novel combination of dilation convolution is to better deal with the lesions appearing at different scales, where small dilation rate pay closer attention to the characteristics of tiny lesions, larger dilation rate focus on large lesions. On the other hand, use of DB's enabled a deeper and more efficient network.

Initially, they extracted regions within FOV from the images and then normalized them to eliminate local contrast differences and uneven illumination. Later, they used small patches  $256 \times 256$  pixels at a stride of 64 (128 for MA) to generate the training samples (only patches that overlap with the lesion ground truth) followed by data augmentation before feeding to the model. To deal with highly imbalanced spread of data, they designed a loss function that is a combination of dice function (Sudre et al., 2017) and 2D cross Entropy as follows:

$$L = -\text{mean}(w_{10} * G * \log(O) + w_{11} * (1 - G) * \log(1 - O) + w_2 * \text{dice}(G)) \quad (5)$$

where  $w_{10}$  and  $w_{11}$  are the factors utilized to keep a balance between the positive and negative pixels, and  $w_2$  is the factor utilized to control significance between dice and cross entropy loss. The values of  $w_{10}$ ,  $w_{11}$  and  $w_2$  were empirically set to 0.7, 0.3 and 0.4 respectively. The models were trained using Adam optimizer with default parameters,  $\beta_1 = 0.9$  and  $\beta_2 = 0.999$ . The initial learning rate was set to  $2 \times 10^{-4}$ , and then divided by 20 in every 20 epochs. This model was implemented with PyTorch1.12 and Tesla M60 platform was utilized for training on the CentOS 7.2 operating system.

### 5.1.4. SOONER (Yunzhi Wang et al.)

Wang et al. adopted U-Net (Ronneberger et al., 2015) architecture for solving retinal lesion segmentation problem. The network takes a  $380 \times 380$  pixels fundus image patch as input and predicts the binary mask of the retinal lesion within  $196 \times 196$  pixels central region of an input patch. They pre-processed fundus images by subtracting the local mean of each color channel and performed random flipping for data augmentation. Batch normalization was utilized to improve training efficiency and all convolution operations adopted 'valid' paddings. For training, they followed a three-stage process for each type of lesions (i.e. MA, HE, EX and SE). For the first stage, they extracted positive image patches from the training set according to given ground truth mask, and randomly extracted negative image patches from fundus images with

and without apparent retinopathy. The objective function was a summation of cross-entropy loss functions for MA, HE, EX and SE. Adam algorithm was employed to optimize the parameters. In the second stage, they fine-tuned U-Net using the extracted patches for each lesion type. Subsequently, they applied optimized U-Net on fundus images in the training set and extracted false-positive patches generated by U-Net. They further fine-tuned U-Net using positive image patches together with false-positive patches (hard negative patches) as a third stage. In the testing phase, they extracted overlapped image patches using a sliding window and fed these patches into the network to get corresponding probability maps. The initial learning rate was set to  $e^{-4}$  and the fixed number of steps was used as a stopping criterion. They implemented U-Net architecture based on TensorFlow library with Nvidia GeForce GTX 1080Ti GPU.

#### 5.1.5. LzyUNCC (Zhongyu Li et al.)

Li et al. developed a method based on FCN by embedding DLA structure (Yu et al., 2017) for segmentation of EX's and SE's. As the lesions are located dispersively and irregularly, the embedding of DLA structure with FCN enables better aggregation of semantic and spatial information from local and global level provides a boost in recognizing their presence. They used retinal images with pixel-level ground truth annotations from both IDRiD and E-Ophtha database. They first adopted a series of methods for data preprocessing and augmentation. Subsequently, considering the correlation between EX's and SE's, they first trained an initial model for segmentation of EX. They chose a smaller model, i.e., DLA-34 to train the segmentation network with binary cross-entropy as a loss function. At last, the trained deep model was fine-tuned for segmentation of SE. While the model training of EX segmentation, a trade-off parameter (penalty) was assigned in loss function to control the weights of foreground pixels, and tried different penalty value from 1 to 16. At last, these segmentation results were fused to adaptively compute the best performance. They adopted original DLA cityscapes segmentation experimental settings and trained the model for 100 epochs with batch size 4, where the poly learning rate was  $(1 - \frac{\text{epoch}-1}{\text{total epoch}})^{0.9}$  with the momentum of 0.9. The initial learning rate was set to 0.01.

#### 5.1.6. SAIHST (Yoon Ho Choi et al.)

Choi et al. proposed a model for segmentation of EX based on U-Net (Ronneberger et al., 2015), in which CONV layers of encoder path are replaced with DB's. Whereas, the decoder path of their model was kept identical to that of general U-Net. They built DB with a growth factor of 12 and  $3 \times 3$  CONV layers, batch normalization, and ReLU activation. The last layer generates a pixel level prediction map for EXs through the sigmoid activation function. For training, they utilized only the green channel of fundus image and enhanced it using Contrast Limited Adaptive Histogram Equalization (CLAHE). Later, each image was padded to a size of  $4352 \times 3072$  pixels and cropped into 204 patches of  $512 \times 512$  pixels. These patches are further augmented and used for training. The losses were calculated by binary cross-entropy. The model was trained for 20 epochs with a mini-batch size of 10 and they used Adam optimizer with an initial learning rate of  $2e^{-4}$ ,  $\beta_1$  of 0.9 and  $\beta_2$  of 0.999. The model was programmed in Keras 2.1.4 served with TensorFlow 1.3.0 backend.

#### 5.1.7. SDNU (Xiaodan sui et al.)

Sui et al. proposed a method based on Mask R-CNN structure to segment lesions from the fundus image. They adopted the implementation of Mask R-CNN from Abdulla (2017) for solving the problem. This method could detect different objects while simultaneously generating instance segmentation mask. Network training precedes the data augmentation process and binary cross-entropy

was used as a loss function. The initial learning rate was set to 0.02 with a momentum of 0.9. They chose ResNet-101 as a backbone. They implemented an algorithm in Keras with Tensorflow as backend and processed on 8 NVIDIA TITAN Xp GPUs. The experimental environment was built under Ubuntu 16.06.

### 5.2. Sub-challenge – 2: Disease grading

For a given image, this task seeks to get a solution to produce severity grade of the diseases i.e. DR (5 class problem) and DME (3 class problem). The summary of participating solutions is as follows:

#### 5.2.1. LzyUNCC (Zhongyu Li et al.)

Li et al. developed a method based on ResNet by embedding DLA structure for automated grading of DR and DME. For this work, they used IDRiD and Kaggle dataset. Initially, for the given training images, they perform data preprocessing and data augmentation. Subsequently, based on the designed ResNet with DLA structure, initial models are trained using 35,000 retinal images from Kaggle dataset. Later, they fine-tuned the model using IDRiD dataset through 5 fold cross-validation technique. Finally, the five outputs are ensembled together as final grades for input images. It is important to note that networks for grading of DR and DME were trained separately. The training was performed by SGD with a mini-batch size of 64, while the learning rate starts from 0.001 and it is then divided by 10 every 20 epochs, for 30 epochs in total. The other hyper-parameters are fixed to settings of original DLA ImageNet classification (Yu et al., 2017).

#### 5.2.2. VRT (Jaemin Son et al.)

Son et al. used network (Son et al., 2018) for DR grading. Kaggle dataset was initially used to pre-train the network and then the model was fine-tuned using IDRiD dataset. The penultimate layer was Global Average Pooled (GAP) and connected with FC layer. The entire output is a single value from which L2 loss was calculated against the true label. SGD was used with Nesterov momentum of 0.9 as an optimizer. Learning rate was set to  $10^{-3}$ . The model was trained for 100 epochs. Fundus image was normalized in the range [0, 1] and the mean was subtracted channel-wise. For grading of DME, segmented EXs (using the segmentation network proposed in sub-challenge – 1), localized fovea and segmented OD (using the segmentation network proposed in sub-challenge – 3) were utilized for making the final decision. With this information, the semi-major axis of segmented OD ( $r$ ) was estimated. Further, the fundus image was divided into three regions as macular region:  $\|x - c\| < r$ , near macular region:  $r < \|x - c\| < 2r$  and remaining region:  $2r < \|x - c\|$ , where  $x$  denotes a point in the image. Furthermore, several features such as sum of intensity for segmented EX, the number of pixels above threshold (178 in the [0, 255] scale), the number of pixels for smallest and largest blob, mean of the number of pixels for blobs are extracted for each area, and binary flag that indicates whether the OD is segmented. Now, features with high importance were selected among numerous features in the initial training due to gradient boosting (for instance, XGBoost) was likely to overfit when provided with overly redundant features. Messidor dataset was added to the given data and out of which 10% of images were left as the validation set. Set of hyper-parameters were searched by grid-search approach. The combination of hyper-parameters that yielded the highest accuracy in the validation set was min child weight: 2, subsample: 0.2, colsample by tree: 0.2,  $\lambda$ : 9.0,  $\alpha$ : 1.0, and depth: 6. Other hyper-parameters are set to default values. All implementations were done by PyTorch v0.4.1 using a server with 8 TITAN X (pascal). The source code is available at [https://bitbucket.org/woalsdnd/isbi\\_2018\\_fundus\\_challenge](https://bitbucket.org/woalsdnd/isbi_2018_fundus_challenge).