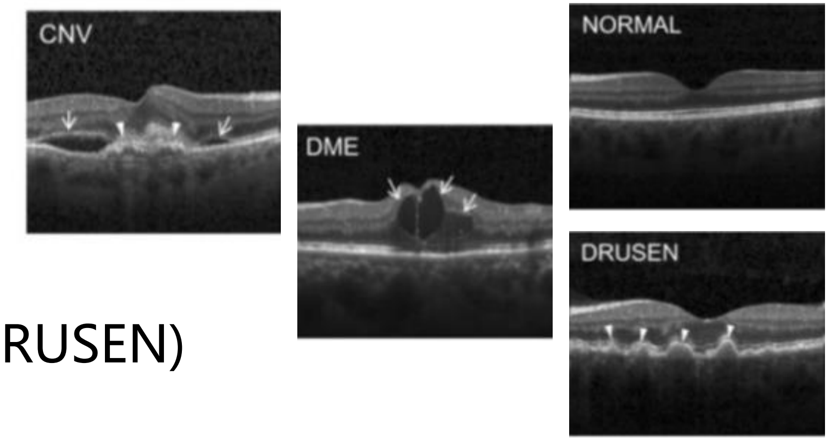
Classification of Retinal OCT Images Based on Deep Learning

Shiruo

Background

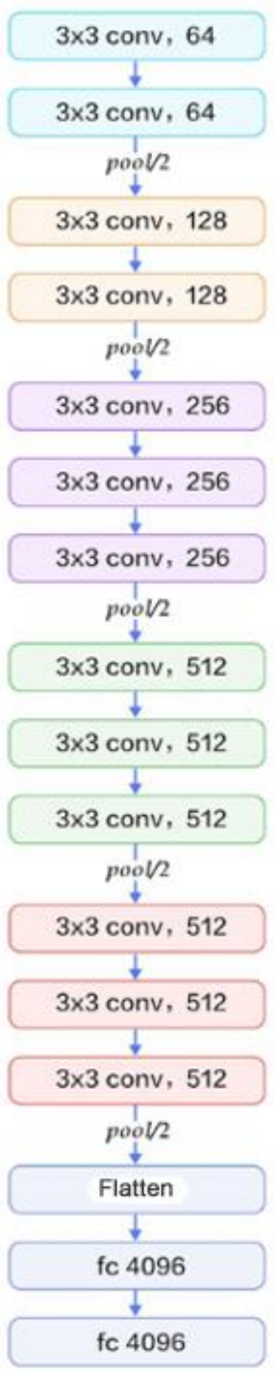
- Optical Coherence Tomography (OCT) technology is widely applied in ophthalmology to view the morphology of retina.

- Common eye diseases:
  choroidal neovascularization (CNV)
  Diabetic macular edema (DME)
  Multiple drusen present in early AMD (DRUSEN)



- Traditional ophthalmic diagnosing process is manual and lengthy. The process can be simplified and accelerated if patients' eye conditions can be told from the medical images by computer.

Model design, implementation and training



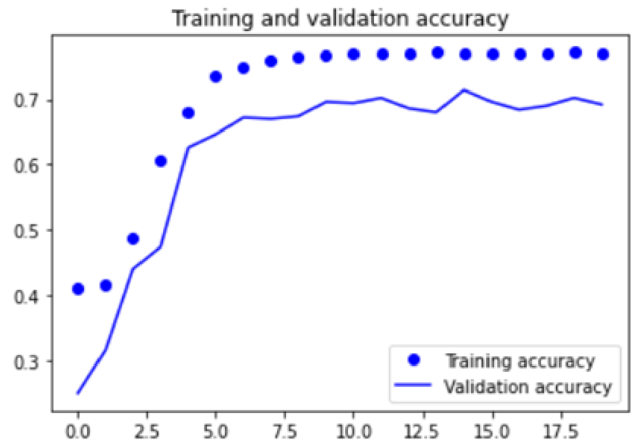
- **Dataset**

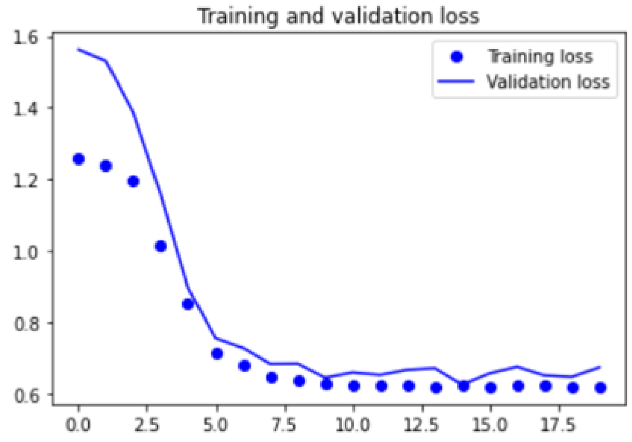
	CNV	DME	DRUSEN	NORMAL	total
Training	31190	10888	7776	26048	75902
Validation	125	125	125	125	500
Test	125	125	125	125	500

According to the data provider, images with more noise and lower quality were included in the training set to help generalization.

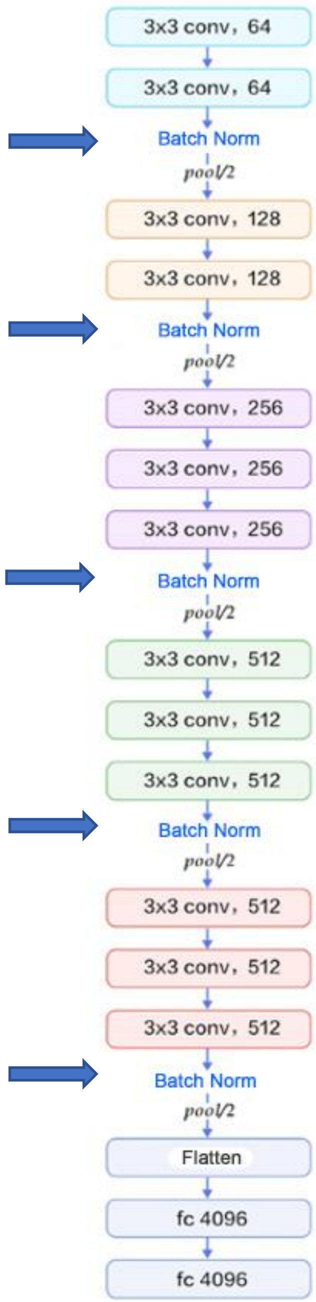
- **Training with pure VGG16 structure**

→ Unsatisfactory outcomes

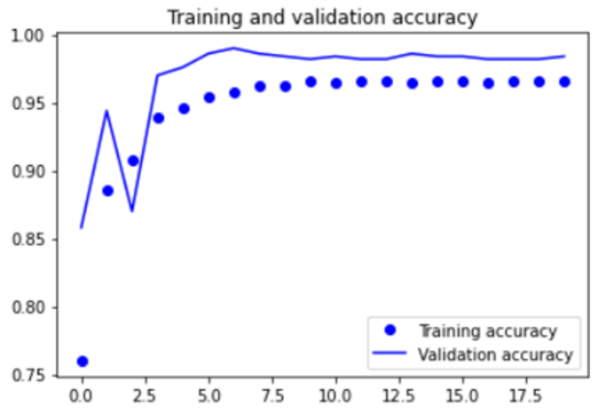


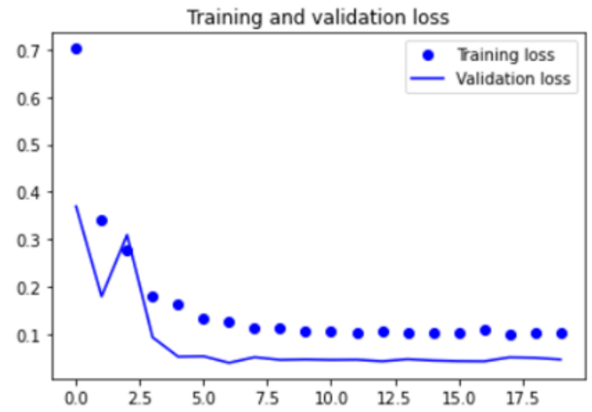


Model design, implementation and training



- **VGG16 structure with Batch Normalization layers inserted**





Batch Normalization (BN) was proposed in 2015 to reduce internal covariate shift during training and accelerate training process.

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Recent researches found that BN has a smoothening effect on the loss (it makes gradients of the loss more Lipschitz).

This enable us to use learning rates of boarder range and make the training process less sensitive to hyperparameter choices.

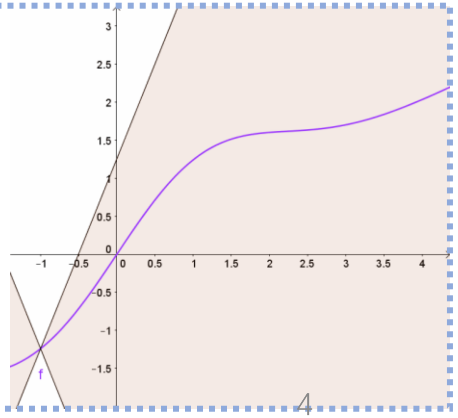
Recall that f is Lipschitz if

$$|f(x_1) - f(x_2)| \le L||x_1 - x_2||$$

for all x_1 and x_2

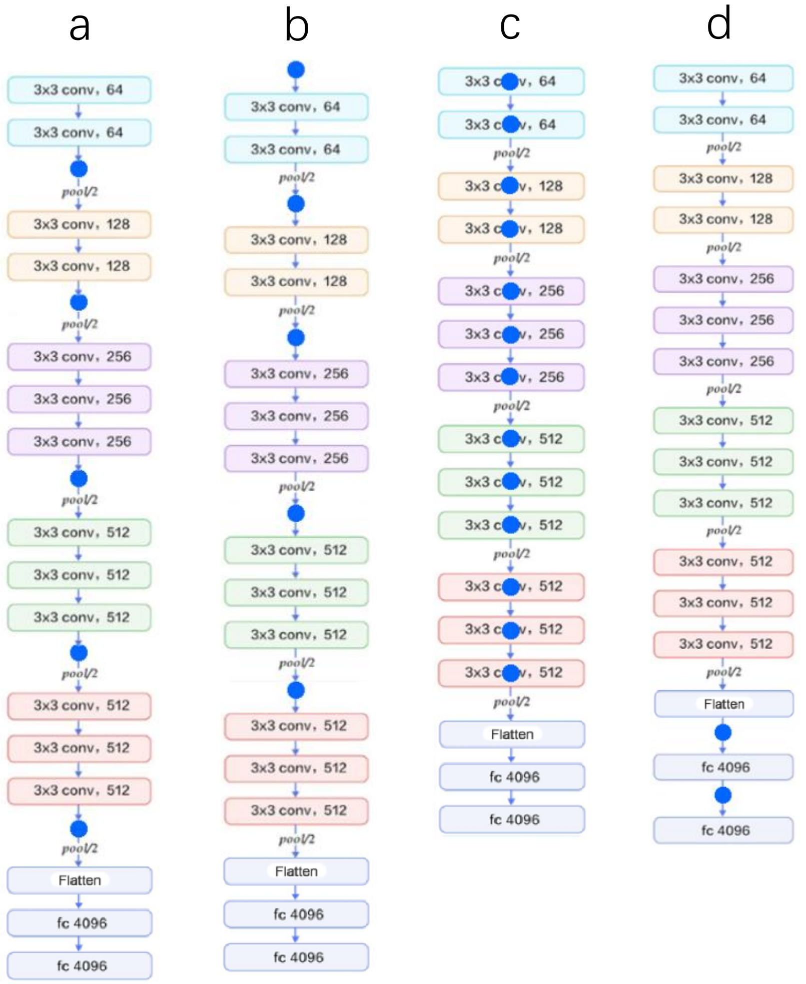
$$\rightarrow \quad f(x_1) \le f(x_2) + L||x_1 - x_2||$$

$$f(x_1) \ge f(x_2) - L||x_1 - x_2||$$

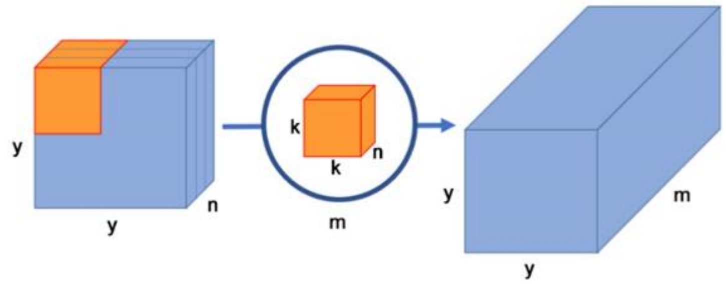


Does the location of BN matters?

	a	b	c	d
Val_acc (max)	0.9900	0.9920	0.9920	0.9900
Val_loss	0.0388	0.0360	0.0401	0.0340
Test_acc	0.9880 ☺	0.9800	0.9700	0.9840
Test_loss	0.0394	0.0744	0.0890	0.0351

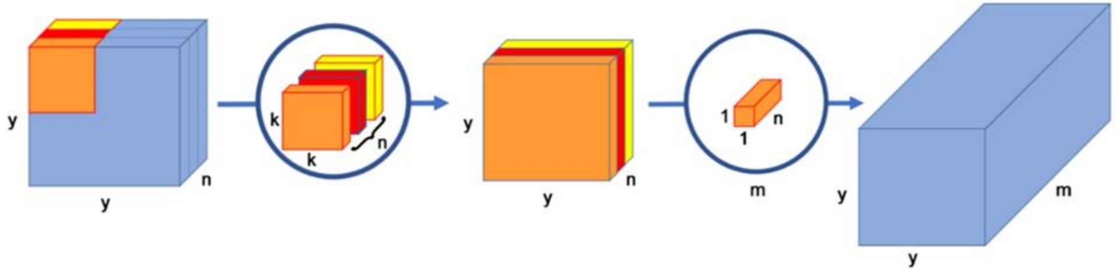


Normal Convolution vs. Depthwise Separable Convolution



Normal Convolution process
(with zero padding)

$$N1 = y^2k^2mn$$



Depthwise Separable Convolution process
(with zero padding)

$$N2 = y^2k^2n + y^2mn$$

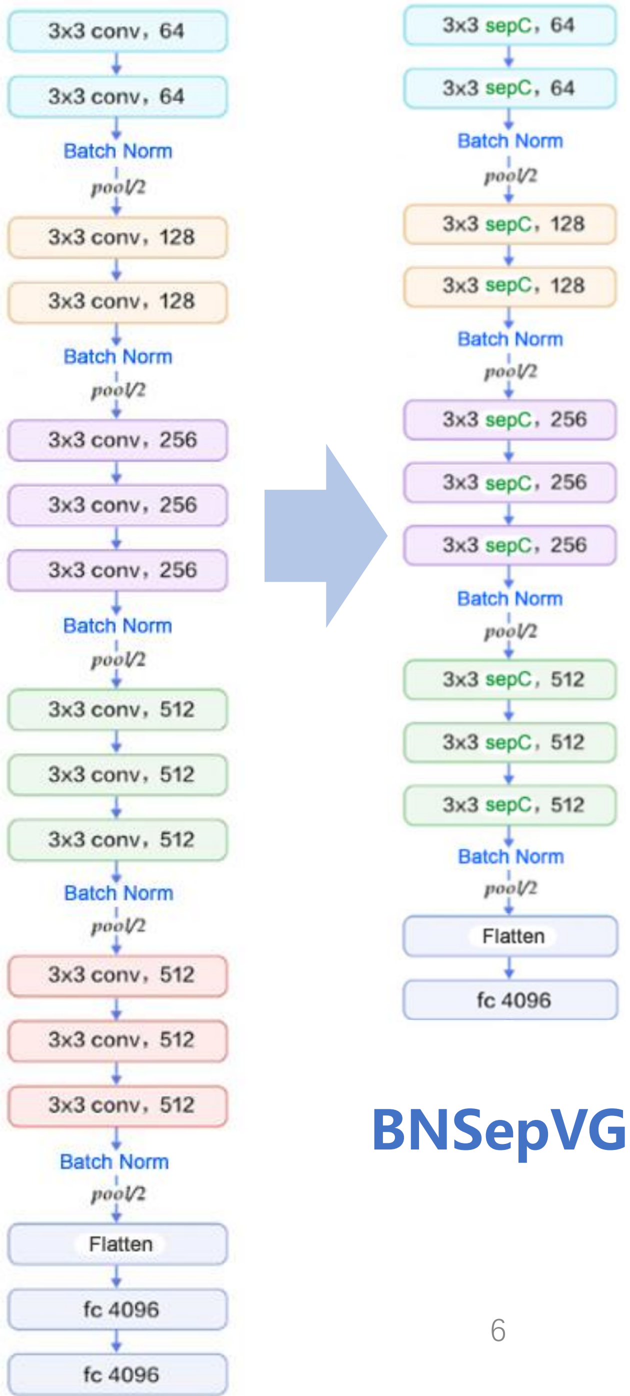
In deep learning, N2 ≪ N1

Make the model lighter!

- If the features extracted by the first few blocks is enough for classification?
- If dropouts can be diminished or removed?
- Replace the Normal Convolutions into Depthwise Separable Convolutions.

a. Previous model (VGG16 with BNs before MaxPooling layers)
b. Last conv block removed (default: dropout rate of 0.5)
c. Last conv block removed + dropout rate of 0.2
d. Last conv block removed + no dropout
e. Last conv block removed + dropout rate of 0.2 + separable convolution
f. Last conv block removed + no dropout + separable convolution

	a	b	c	d	e	f
Val_acc (max)	0.9900	0.9960	0.9920	0.9920	0.9900	0.9940
Val_loss	0.0388	0.0322	0.0304	0.0363	0.0183	0.0306
Test_acc	0.9880	0.9840	0.9880	0.9720	0.9840	0.9880 ☺
Test_loss	0.0394	0.0621	0.0562	0.0775	0.0575	0.0460



BNSepVGG

Analyses and Comparisons

1. Model trained in BNSepVGG structure
2. Model trained in VGG16 structure
3. Model trained in AlexNet structure
4. ImageNet pre-trained DenseNet201 model
5. ImageNet pre-trained ResNet50 model
6. ImageNet pre-trained InceptionV3 model
7. ImageNet pre-trained Xception model

	1	2	3	4	5	6	7
Val_acc (max)	0.9940	0.7141	0.8640	0.9420	0.6540	0.6760	0.8980
Val_loss	0.0306	0.6156	0.3495	0.1801	0.8296	0.8400	0.2457
Test_acc	0.9880	0.7120	0.8320	0.8160	0.6420	0.6760	0.8120
Test_loss	0.0460	0.6508	0.4465	0.5719	0.8517	0.8656	0.4662

1				2				3				4				5				6				7			
124	1	0	0	124	1	0	0	124	1	0	0	105	18	1	1	112	13	0	0	108	16	0	1	120	4	0	1
0	125	0	0	13	89	4	19	4	116	3	2	1	120	0	4	13	95	0	17	17	100	0	8	10	106	0	9
5	0	120	0	36	11	19	59	52	4	68	1	26	28	59	12	43	22	2	58	48	40	16	21	44	8	56	17
0	0	0	125	0	1	0	124	0	11	6	108	0	1	0	124	0	11	2	112	1	10	0	114	0	1	0	124

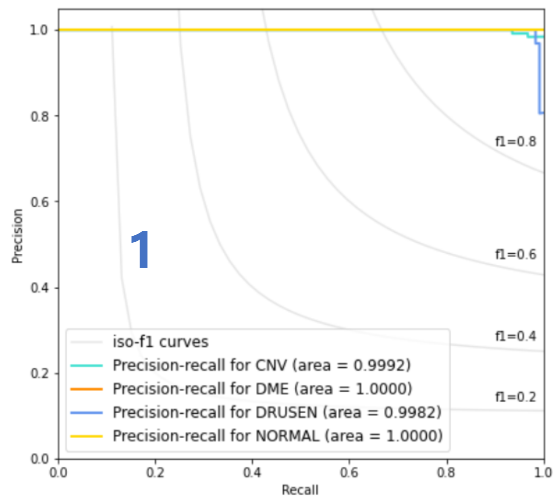
Analyses and Comparisons

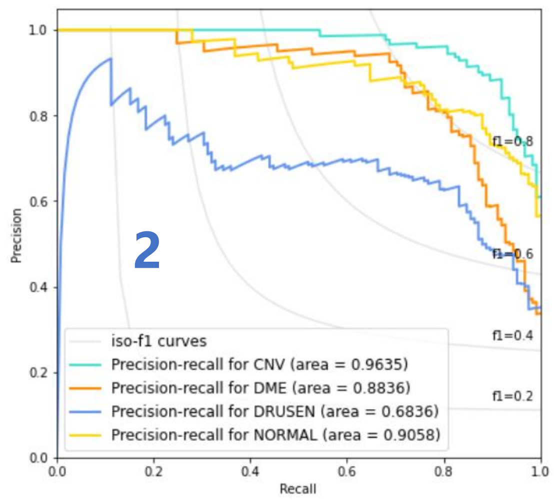
$$\text{precision} = \text{PPV} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

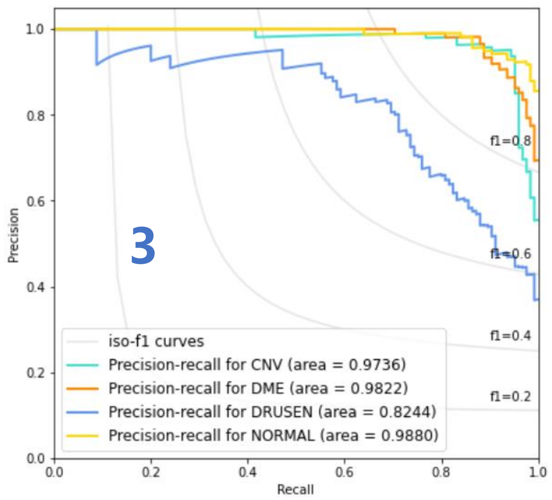
Precision-Recall (PR) Curve

$$\text{recall} = \text{sensitivity} = \text{TPR} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Nagative}}$$

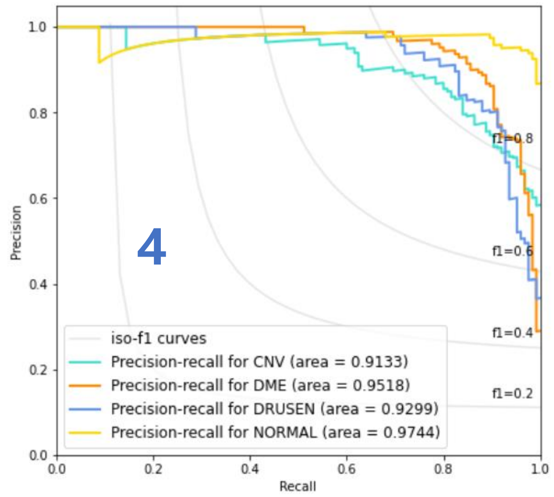
$$f_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

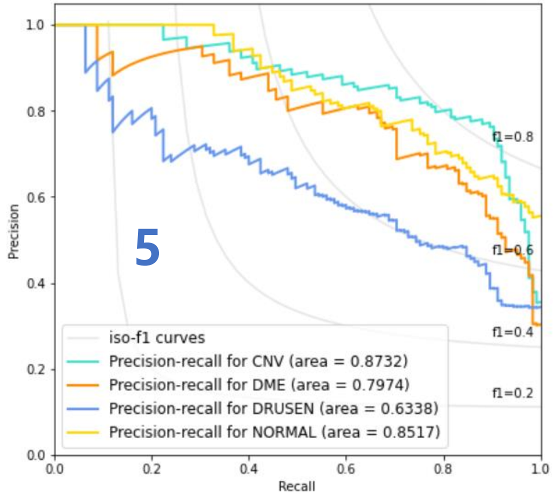


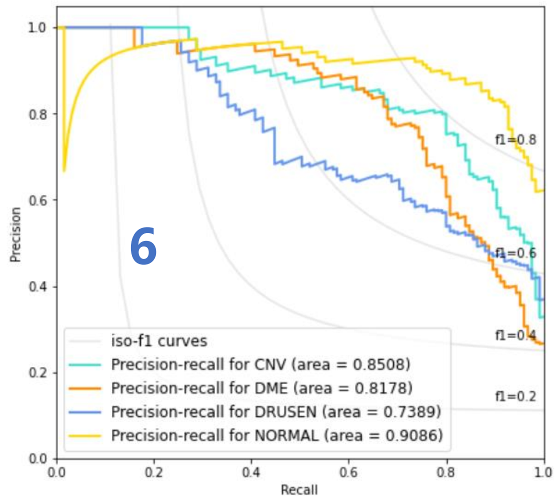


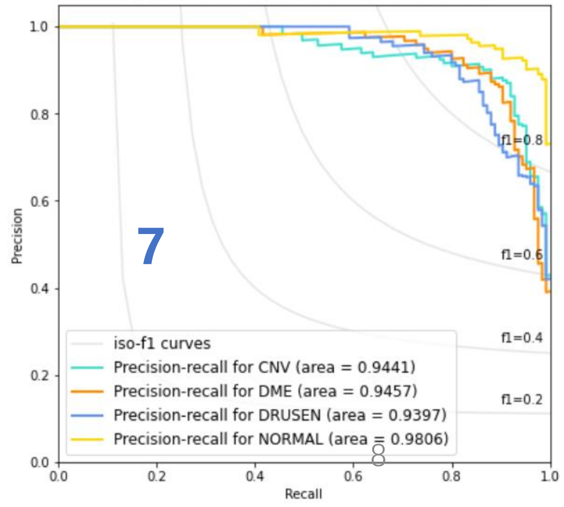


1. Model trained in BNSepVGG structure
2. Model trained in VGG16 structure
3. Model trained in AlexNet structure
4. ImageNet pre-trained DenseNet201 model
5. ImageNet pre-trained ResNet50 model
6. ImageNet pre-trained InceptionV3 model
7. ImageNet pre-trained Xception model









In real-life medical practice,

CNV, DME → Urgent

DRUSEN, NORMAL → Non-urgent

	1	2	3	4	5	6	7
sensitivity	1	0.908	0.98	0.976	0.932	0.964	0.96
specificity	0.98	0.808	0.732	0.78	0.696	0.604	0.788

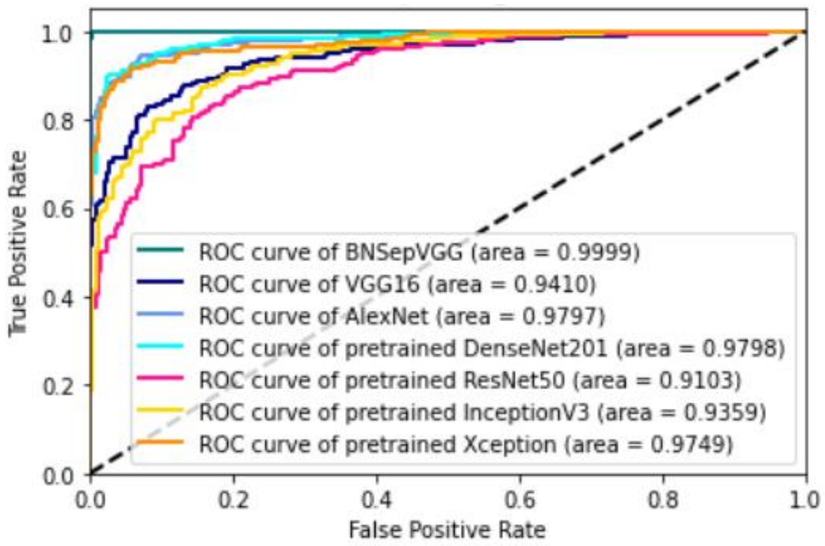
1. Model trained in BNSepVGG structure
2. Model trained in VGG16 structure
3. Model trained in AlexNet structure
4. ImageNet pre-trained DenseNet201 model
5. ImageNet pre-trained ResNet50 model
6. ImageNet pre-trained InceptionV3 model
7. ImageNet pre-trained Xception model

$$\text{FPR} = 1 - \text{specificity} = \frac{\text{False Positive}}{\text{True Negative} + \text{False Positive}}$$

$$\text{TPR} = \text{sensitivity} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

ROC curve

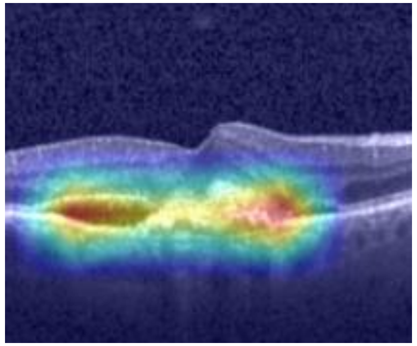
(Receiver Operating Characteristic)

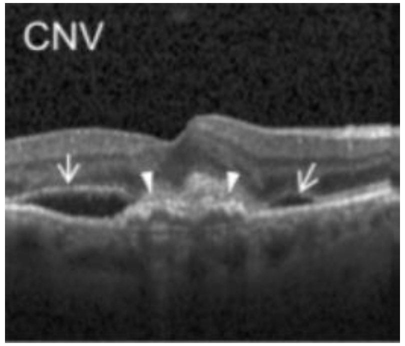


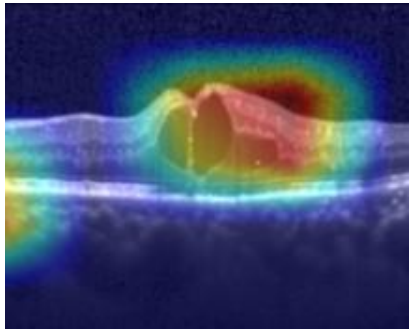
Grad-CAM Heatmaps

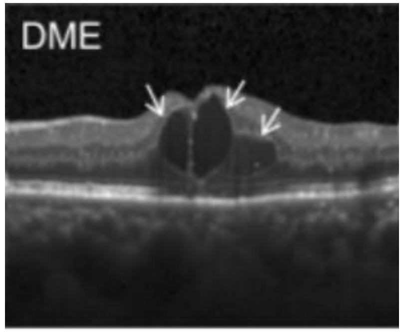
how intensely the input image activates the class

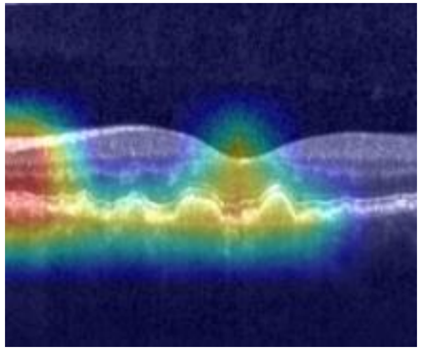
CAM generated by BNSepVGG model **Marked examples**

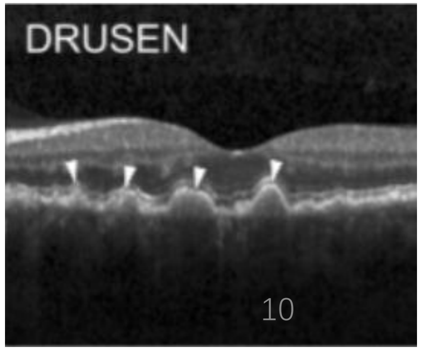












Thanks for your attention

--Shiruo