Report for Course Deep Learning 2024

Abu Taher
University of Oulu
90570-Oulu, Finland
ataher24@student.oulu.fi

Md Rabiul Hasan
University of Oulu
90570-Oulu, Finland
mdhasa24@student.oulu.fi

1. Approach

In this section, you should aim to give readers a clear understanding of the nature and context of your project or task. If a visual overview diagram can help illustrate your methodology or approach, it can be a valuable addition to enhance comprehension.

1.1. First step

Here you can start to introduce the approach in detail.

1.2. Second step

1.3. More information about approach

1.4. Mathematics

Please number all of your sections and displayed equations as in these examples:

$$E = m \cdot c^2 \tag{1}$$

and

$$v = a \cdot t. \tag{2}$$

2. Experiment and Discussion

In this section, you should introduce the datasets, training setups, the results, visualization, and analysis.

2.1. Dataset

In this study, we utilized two distinct datasets for diabetic retinopathy detection. The first dataset, DeepDRiD (Diabetic Retinopathy Image Dataset), is a comprehensive resource designed to facilitate the diagnosis and grading of diabetic retinopathy [3]. It covers multiple stages: no apparent retinopathy, mild, moderate, severe, proliferative diabetic retinopathy (PDR), and cases where fundus images from both eyes are inadequate for assessment. The dataset contains 2,000 images, allocated as 1,200 for training, 400 for validation, and 400 for testing, collected from patients with diverse backgrounds. The images were captured under varying conditions, such as different camera angles and lighting environments. Each patient contributed four images—two from the left eye and two from the right eye—along with a detailed list of relevant features.

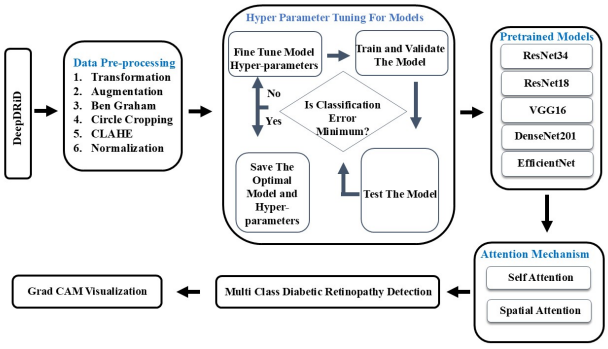

Figure 1. Diabetic Retinopathy Detection on DeepDRiD dataset

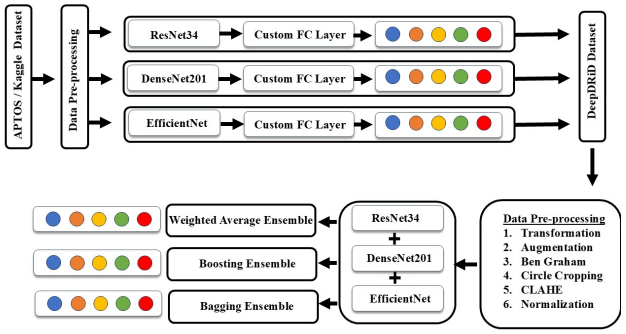

Figure 2. Diabetic Retinopathy Detection using Ensemble Learning

The second dataset, APTOS-2019 Blindness Detection, includes 3,662 samples obtained from participants in rural regions of India. These fundus photographs were taken under diverse environmental conditions over an extended timeframe. The dataset was curated by Aravind Eye Hospital, where a team of trained medical professionals labeled the images according to the International Clinical Diabetic Retinopathy Disease Severity Scale (ICDRSS).

2.2. Pre-trained Models with Transfer Learning

2.2.1 VGG16

The VGG (Visual Geometry Group) model is a widely recognized deep CNN architecture utilized for image recognition tasks. It comes in two main variants: VGG16 and VGG19, which consist of 16 and 19 layers, respectively. The architecture was initially trained on the ImageNet dataset, which contains over 14 million images spanning 1,000 distinct categories. Due to its remarkable performance metrics, the pre-trained VGG16 model is extensively employed by researchers for image analysis across various applications. [2]

2.2.2 DenseNet201

The term "DenseNet" refers to a group of convolutional neural networks (CNNs) distinguished by their use of dense connections between layers. Variants within this family include DenseNet201 and DenseNet169. For instance, DenseNet201, when applied to process images with dimensions of 128×128 pixels, utilizes dense blocks that establish direct connections among all 201 layers. This design enables a seamless flow of information, allowing each layer to draw features from preceding layers while simultaneously passing its feature maps to subsequent layers, all while maintaining the network's feed-forward structure.

2.2.3 EfficientNet

The EfficientNet family includes eight models, from B0 to B7, offering improved accuracy with minimal increase in parameters. Unlike traditional CNNs, EfficientNet uses the Swish activation function instead of ReLU. Its core building block, MBConv, introduced in MobileNetV2 , is more extensively utilized here due to a higher FLOPS budget. MBConv expands and compresses channels with direct connections between bottlenecks, reducing computation through depthwise separable convolutions by nearly k2k$\hat{2}$k2, where kkk is the kernel size . [1]

2.2.4 ResNet34

ResNet-34 is a convolutional neural network with 34 layers that is essential in a wide range of computer vision applications. Known as Residual Networks, ResNet has become a fundamental component in the design of neural networks. Originally presented in 2015, it has since evolved into a key tool in both academic research and commercial uses. It is particularly valuable in fields like image recognition and medical research, where its deep learning features hold promise for diagnosing and treating various diseases.

2.3. Training

2.3.1 Task A

In this part, we utilized pre-trained Convolutional Neural Networks (CNNs) — ResNet38, VGG16, DenseNet201, and EfficientNet — to fine-tune their performance on the DeepDRiD dataset. Transfer learning was applied to these models using their ImageNet weights, and the fully connected layers were tailored for five-class diabetic retinopathy classification. To enhance model performance, we experimented with batch sizes of 24, learning rates of 0.001, and data augmentation techniques, as detailed in Table 1.

The highest Cohen Kappa scores were achieved using the optimal hyperparameters and augmentation strategies summarized in Tables 1 and 2. Specifically, the best Kappa scores for ResNet38, VGG16, DenseNet201, and EfficientNet were 0.8170, 0.8547, 0.8127, and 0.8009, respectively.

Furthermore, Grad-CAM was employed to visualize the regions of importance identified by the models during classification. This method utilizes the gradients flowing into the final convolutional layer to produce a coarse localization map, highlighting critical areas in the input images that influence the predictions. The resulting heat maps provide insight into the model's focus during the decision-making process, as illustrated in Figure 1.

Table 1. Augmentation technique applied on DeepDRiD dataset.

Technique	Settings
Resize	(224, 224)
Random Horizontal Flip	p=0.5
Color Jitter	brightness=0.2, contrast=0.2, saturation=0.2, hue=0.1
Random Rotation	degrees=30
Random Affine	degrees=0, scale=(0.8, 1.2), translate=(0.1, 0.1)
Random Erasing	p=0.5, scale=(0.02, 0.1), ratio=(0.3, 3.3)
Normalize	mean=[0.5, 0.5, 0.5], std=[0.5, 0.5, 0.5]

2.3.2 Task B

In this part, we used five pre-trained models, namely VGG16, ResNet18, ResNet34, DenseNet121, and EfficientNet-B0. For image preprocessing, we applied CLAHE, Ben Graham transformation, image sharpening, Gaussian blur, random cropping, random rotation, random vertical flipping, random horizontal flipping, and color jitter.

The architecture of the VGG16 model is shown in Figure []. We trained this model using five preprocess-

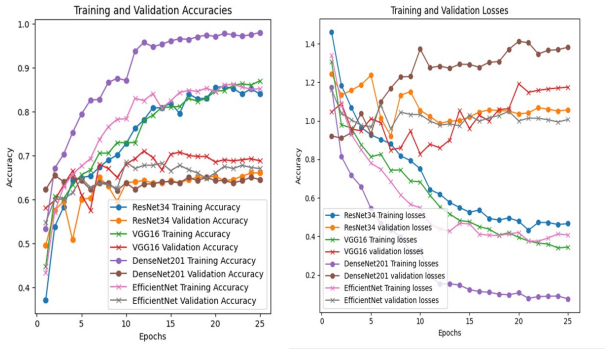

Figure 3. Training metrics of ResNet34, VGG16, DenseNet201, EfficientNet on DeepDRiD dataset

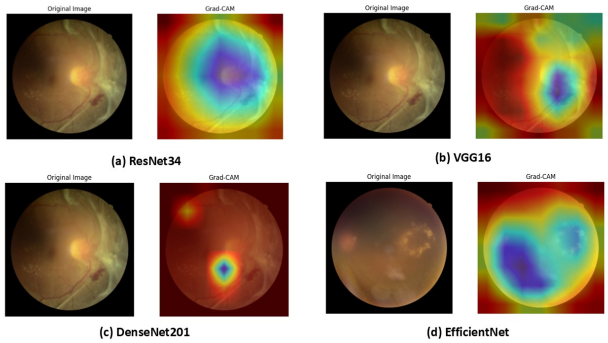

Figure 4. Grad-CAM visualization of ResNet34, VGG16, DenseNet201, EfficientNet on DeepDRiD dataset.

ing methods. Initially, we used the default preprocessing method provided in the template code. Then, we combined CLAHE and sharpening in one method, referred to as custom_transform_train1. Next, we used Ben Graham transformation and sharpening in another method, called custom_transform_train2. In custom_transform_train3, we applied both CLAHE and Ben Graham transformation. In custom_transform_train4, we combined CLAHE with the default method.

For custom_transform_train1, custom_transform_train2, and custom_transform_train3, the results were unsatisfactory due to class imbalance. The model performed well for the first class but poorly for the last class. However, when trained with the default augmentation method, the model performed significantly better. The training and validation accuracies and losses are shown in the figure below.

The architecture of ResNet18 is shown in Figure []. We loaded the pre-trained weights into the model and trained it on the DeepDRiD dataset. For this model, we used a learning rate of 0.001 and a dropout rate of 0.35. We trained the model using all five augmentation methods. Among them, the most effective method was the [] method.

ResNet34, DenseNet121, and EfficientNet-B0 were

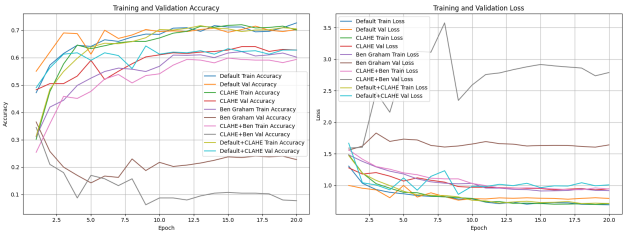

Figure 5. Training metrics of VGG16 using different augmentation techniques

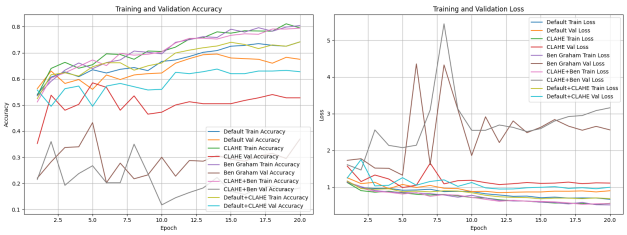

Figure 6. Training metrics of Resnet18 using different augmentation techniques

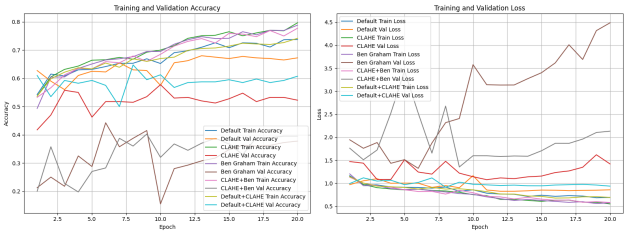

Figure 7. Training metrics of Resnet34 using different augmentation techniques

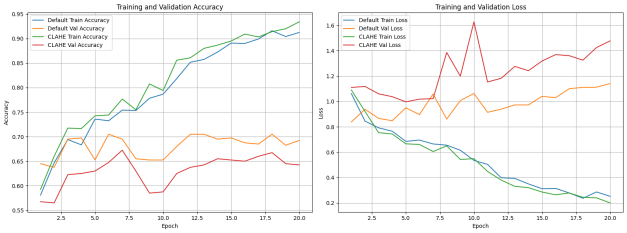

Figure 8. Training metrics of Efficientnet using different augmentation techniques

trained in a similar manner using all the augmentation techniques.

Oversampling: There is a class imbalance in the dataset. We used an oversampling method to balance the dataset. The number of samples per class is shown in Figure []. After oversampling, all classes had the same number of samples. While implementing the oversampling method, we sampled at the patient level rather than the individual im-

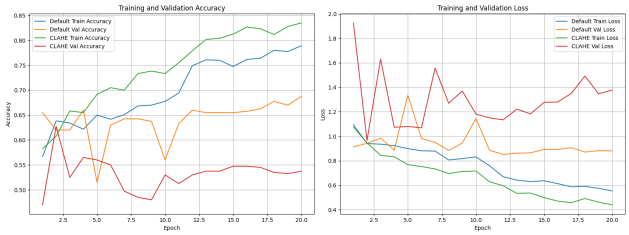

Figure 9. Training metrics of Densenet121 using different augmentation techniques

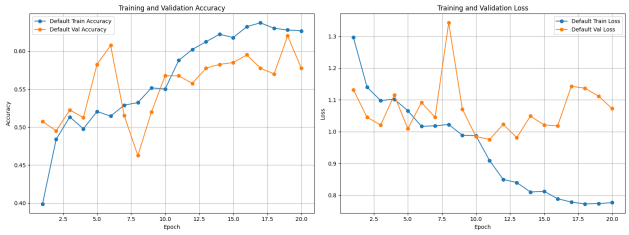

Figure 10. VGG16 training matrices—trained on oversampled data

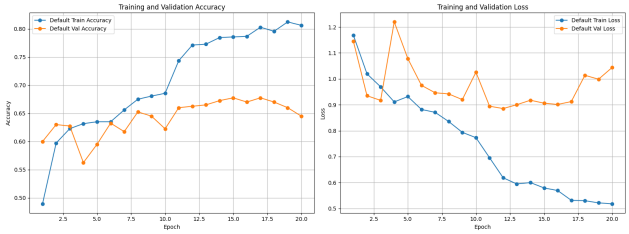

Figure 11. Resnet18 training matrices—trained on oversampled data

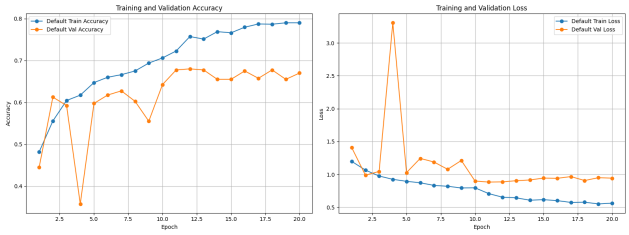

Figure 12. Resnet34 training matrices—trained on oversampled data

age level.

When we trained the five models using the oversampled dataset, we obtained the following results: [].

2.3.3 Task C

In this part, we have applied self-attention and spatial attention mechanisms to ResNet34, VGG16, DenseNet201,

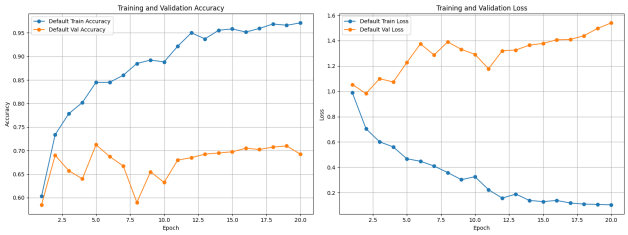

Figure 13. EfficientNet training matrices—trained on oversampled data

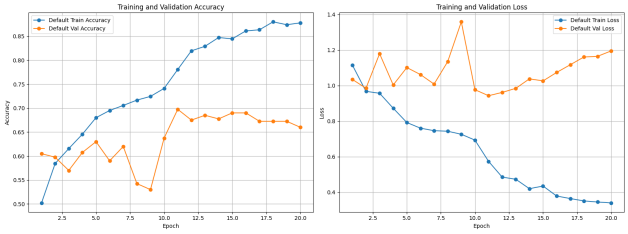

Figure 14. DenseNet training matrices—trained on oversampled data

and EfficientNet. Self-attention captures global dependencies, while spatial attention highlights key regions, enhancing feature representation, and improving the models' overall performance. For DenseNet201 by applying spatial attention mechanism has improved it overall best kappa from 0.8127 to 0.8494. For DenseNet201, the spatial attention mechanism notably increased the kappa score from 0.8127 to 0.8494. However, for the other models, the attention mechanisms resulted in only marginal changes in kappa scores.

2.3.4 Task D

For this part, we used three pre-trained models: ResNet34, DenseNet121, and EfficientNet. We implemented three ensemble techniques: weighted average, bagging, and boosting.

Weighted Average: In this method, we assigned arbitrary weights to each model during the testing phase. The predictions generated by each model were multiplied by their respective weights and summed up to produce the final prediction. This method improved the testing accuracy of the models when tested on Kaggle.

Bagging: In this method, we used bootstrapping to create multiple datasets and trained each model separately. During the aggregation phase, we used a majority voting method to generate the final predictions.

Boosting: For this method, the models were trained sequentially, with weights assigned to emphasize the data with higher errors. At the prediction stage, we applied a majority voting system for the final predictions.

2.4. References

When referenced in the text, enclose the citation number in square brackets, for example [**?**].

3. Conclusion

Summary of your report.

References

[1] Ümit Atila, Murat Uçar, Kemal Akyol, and Emine Uçar. Plant leaf disease classification using efficientnet deep learning model. *Ecological Informatics*, 61:101182, 2021. 2

[2] Rabiul Hasan, Shah Muhammad Azmat Ullah, Avizit Nandi, and Abu Taher. Improving pneumonia diagnosis: A deep transfer learning cnn ensemble approach for accurate chest x-ray image analysis. In *2023 International Conference on Information and Communication Technology for Sustainable Development (ICICT4SD)*, pages 109–113. IEEE, 2023. 2

[3] Ruhan Liu, Xiangning Wang, Qiang Wu, Ling Dai, Xi Fang, Tao Yan, Jaemin Son, Shiqi Tang, Jiang Li, Zijian Gao, et al. Deepdrid: Diabetic retinopathy—grading and image quality estimation challenge. *Patterns*, 3(6), 2022. 1