Deep Learning Approach to Diabetic Retinopathy Detection

|  |  |
| --- | --- |
| Keywords: | Deep learning, diabetic retinopathy, deep convolutional neural network, multi-target learning, ordinal regression, classification, SHAP, Kaggle, APTOS. |
| Abstract: | Diabetes is a disease that is strongly associated with both microvascular and macrovascular complications, including retinopathy, nephropathy, neuropathy (microvascular), ischemic heart disease, peripheral vascular disease, and cerebrovascular disease (macrovascular), resulting in organ and tissue damage.[4] Diabetic Retinopathy (DR) is an important cause of visual impairment among persons with diabetes. The prevalence of diabetes is higher in urban areas (11.2%, 95% CI: 10.6–11.8) than in rural areas (5.2%, 95% CI: 4.9–5.4; P < 0·0001),[5] but DR does not show this variation.[6] DR accounted for 1.07% of blindness and 1.25% of moderate to severe visual impairment (MSVI) in 2015.[7] There are no recent studies on the prevalence of DR across all the geographical divisions of India. This makes it difficult to identify where DR screening and treatment programs are most needed. Most of the available estimates of DR are from diabetes clinics, which is subject to bias, limiting their use in planning ophthalmic services for persons with diabetes in the general population. Prevalence of DR has been included in the Rapid Assessment of Avoidable Blindness (RAAB + DR) survey by the International Center for Eye Health (ICEH), London as a relatively rapid and affordable method for estimating the burden of diabetes and DR in the population aged ≥50 years. This will help to plan and prioritize diabetic eye care services. The objectives of the current study were to assess the prevalence of total, known, and new diabetes and to assess the prevalence of DR and sight-threatening DR (STDR) among people with diabetes. Other objectives included evaluation of coverage of DR examinations among people with known diabetes (the proportion of people with known diabetes who have had an eye exam ever and in the past year) and/or to know the levels of glycemic control (random blood sugar <200 mg/dl) among them. |

Blindness Detection Dataset (13000 images).

# INTRODUCTION

Diabetic retinopathy (DR) is one of the most threatening complications of diabetes in which damage occurs to the retina and causes blindness. It damages the blood vessels within the retinal tissue, causing them to leak fluid and distort vision. Along with diseases leading to blindness, such as cataracts and glaucoma, DR is one of the most frequent ailments, according to the US, UK, and Singapore statistics (NCHS, 2019; NCBI, 2018; SNEC, 2019).

DR progresses with four stages:

* *Mild non-proliferative retinopathy*, the earliest stage, where only microaneurysms can occur;
* *Moderate non-proliferative retinopathy*, a stage which can be described by losing the blood vessels’ ability of blood transportation due to their distortion and swelling with the progress of disease;
* *Severe non-proliferative retinopathy* results in deprived blood supply to the retina due to the increased blockage of more blood vessels, hence signaling the retina for the growing of fresh blood vessels;
* *Proliferative diabetic retinopathy* is the advanced stage, where the growth features secreted by the retina activate proliferation of the new blood vessels, growing along inside covering of retina in some vitreous gel, filling the eye.

Each stage has its characteristics and particular properties, so doctors possibly could not take some of them into account, and thus make an incorrect diagnosis. So this leads to the idea of creation of an automatic solution for DR detection.

At least 56% of new cases of this disease could be reduced with proper and timely treatment and monitoring of the eyes (Rohan T, 1989). However, the initial stage of this ailment has no warning signs, and it becomes a real challenge to detect it on the early start. Moreover, well-trained clinicians sometimes could not manually examine and evaluate the stage from diagnostic images of a patient’s fundus (according to Google’s research (Krause et al., 2017), see Figure 1). At the same time, doctors will most often agree when lesions are apparent. Furthermore, existing ways of diagnosing are quite inefficient due to their duration time, and the number of ophthalmologists included in patient’s problem solution. Such sources of disagreement cause wrong diagnoses and unstable ground-truth for automatic solutions, which were provided to help in the research stage.

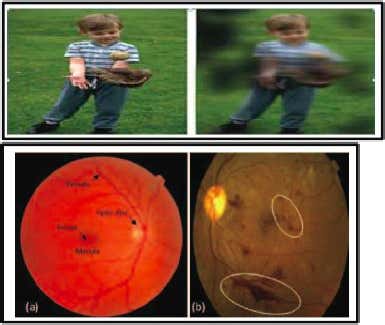


Figure 1: Google showed that ophtalmologists’ diagnoses differ for same fundus image. Best viewed in color.

Thus, algorithms for DR detection began to appear. The first algorithms were based on different classical algorithms from computer vision and setting thresholds (Michael D. Abrmoff and Quellec, 2010; Christopher E.Hann, 2009; Nathan Silberman and Subramanian, 2010). Nevertheless, in the past few years, deep learning approaches have proved their superiority over other algorithms in tasks of classification and object detection (Harry Pratt, 2016). In particular, convolutional neural networks (CNN) have been successfully applied in many adjacent subjects and for diagnosis of diabetic retinopathy itself (Shaohua Wan, 2018; Harry Pratt, 2016).

In 2019, APTOS (Asia Pacific TeleOphthalmology Society) and competition ML platform Kaggle challenged ML and DL researchers to develop a five-class DR automatic diagnosing solution (APTOS 2019 Blindness Detection Dataset). In this paper, we propose the transfer learning approach and an automatic method for detection of the stage of diabetic retinopathy by single photography of the human fundus. This approach is able to learn useful features even from a noisy and small dataset and could be used as a DR stages screening method in automatic solutions. Also, this method was ranked 54 of 2943 different methods on APTOS 2019 Blindness Detection Competition and achieved the quadratic weighted kappa score of 0.92546.

# LITERATURE SURVEY

Many research efforts have been devoted to the problem of early diabetic retinopathy detection. First of all, researchers were trying to use classical methods of computer vision and machine learning to provide a suitable solution to this problem. For instance, Priya et al. (Priya and Aruna, 2012) proposed a computer-vision-based approach for the detection of diabetic retinopathy stages using color fundus images. They tried to extract features from the raw image, using the image processing techniques, and fed them to the SVM for binary classification and achieved a sensitivity of 98%, specificity 96%, and accuracy of 97.6% on a testing set of 250 images. Also, other researchers tried to fit other models for multiclass classification, e.g., applying PCA to images and fitting decision trees, naive Bayes, or k-NN (Conde et al., 2012) with best results 73.4% of accuracy, and 68.4% for F-measure while using a dataset of 151 images with different resolutions.

Nandana Prabhu et al., [1] have proposed a system for Diabetic Retinopathy detection based on the presence of the feature that shows the symptoms of the disease. The system makes use of fundus images, the bright lesions on the retina and the exudates are extracted as they indicate the symptoms of the disease. Based on the features extracted various stages of the disease is detected using hierarchical classification

With the growing popularity of deep learningbased approaches, several methods that apply CNNs to this problem appeared. Pratt et al. (Harry Pratt, 2016) developed a network with CNN architecture and data augmentation, which can identify the intricate features involved in the classification task such as micro-aneurysms, exudate, and hemorrhages in the retina and consequently provide a diagnosis automatically and without user input. They achieved a sensitivity of 95% and an accuracy of 75% on 5,000 validation images. Also, there are other works on CNNs from other researchers (Carson Lam and Lindsey, 2018; Yung-Hui Li and Chung, 2019). It is useful to note that Asiri et al. reviewed a significant amount of methods and datasets available, highlighting their pros and cons (Asiri et al., 2018). Besides, they pointed out the challenges to be addressed in designing and learning about efficient and robust deeplearning algorithms for various problems in DR diagnosis and drew attention to directions for future research.

Yogesh Kumaran, Chandrashekar M. Patil [4] have presented a brief survey in detecting DR using different preprocessing and segmentation techniques as it is difficult to process the raw fundus images by machine learning algorithms. They have given brief view from the nutshell in order to facilitate others on recent advancement and research for their work. This can also help in the insight detection which is based on the work of researches in the field

# PROBLEMSTATEMENT

## Datasets

The image data used in this research was taken from several datasets. We used an open dataset from Kaggle Diabetic Retinopathy Detection Challenge 2015 (EyePACs, 2015) for pretraining our CNNs. This dataset is the largest available publicly. It consists of 35126 fundus photographs for left and right eyes of American citizens labeled with stages of diabetic retinopathy:

* No diabetic retinopathy (label 0)
* Mild diabetic retinopathy (label 1)
* Moderate diabetic retinopathy (label 2)
* Severe diabetic retinopathy (label 3)
* Proliferative diabetic retinopathy (label 4)

In addition, we used other smaller datasets: Indian Diabetic Retinopathy Image Dataset (IDRiD) (Sahasrabuddhe and Meriaudeau, 2018), from which we used 413 photographs of the fundus, and MESSIDOR (Methods to Evaluate Segmentation and Indexing Techniques in the field of Retinal Ophthalmology) (Decencire et al., 2014) dataset, from which we used 1200 fundus photographs. As the original MESSIDOR dataset has different grading from other datasets, we used the version that was relabeled to standard grading by a panel of ophthalmologists (Google Brain, 2018).

As the evaluation was performed on Kaggle APTOS 2019 Blindness Detection (APTOS2019) dataset (APTOS, 2019), we had access only to the training part of it. The full dataset consists of 18590 fundus photographs, which are divided into 3662 training, 1928 validation, and 13000 testing images by organizers of Kaggle competition. All datasets have similar distributions of classes; distribution for APTOS2019 is shown in Figure 2.

As different datasets have a similar distribution, we considered it as a fundamental property of this type of data. We did no modifications to the dataset distribution (undersampling, oversampling, etc.).

The smallest native size among all of the datasets is 640x480. Sample image from APTOS2019 is shown in Figure 3.

## Evaluation metric

In this research, we used quadratic weighted Cohen’s kappa score as our main metric. Kappa score measures the agreement between two ratings. The quadratic weighted kappa is calculated between the

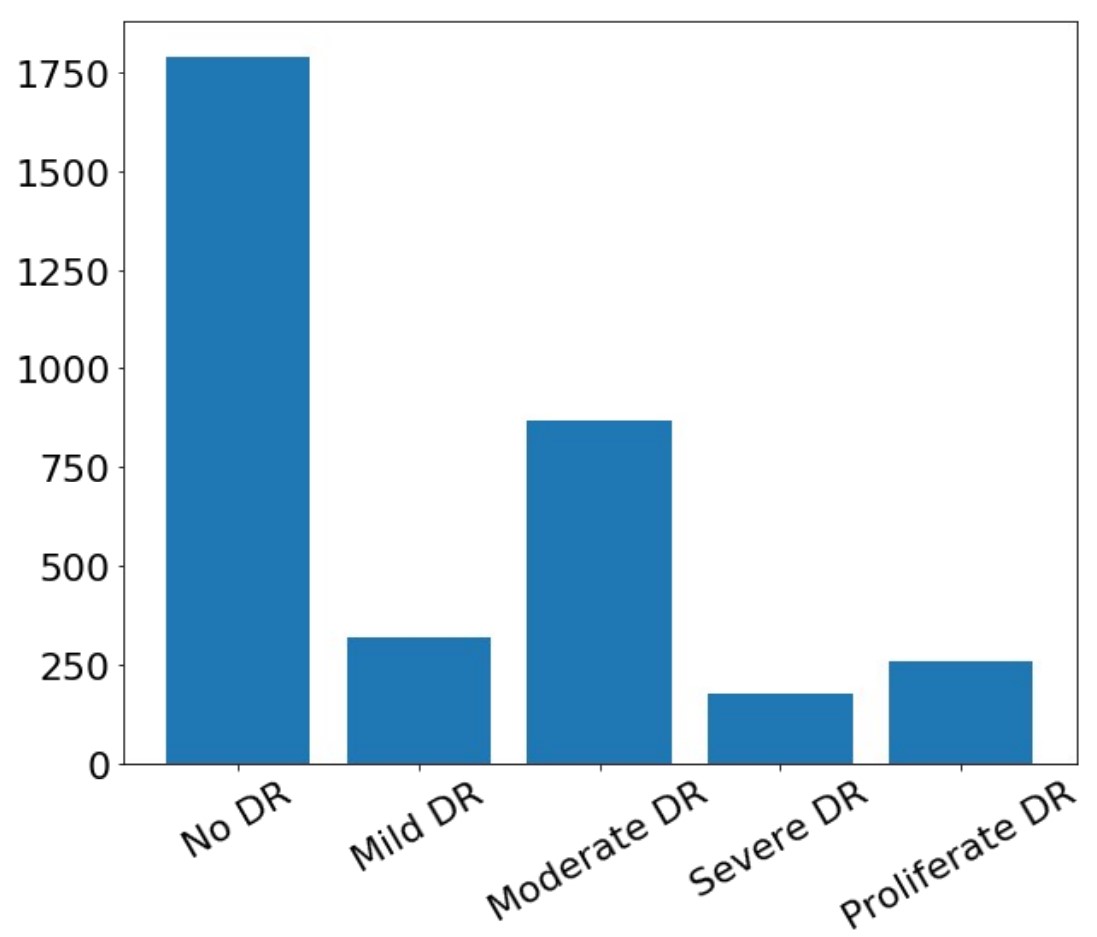


Figure 2: Classes distribution in APTOS2019 dataset.

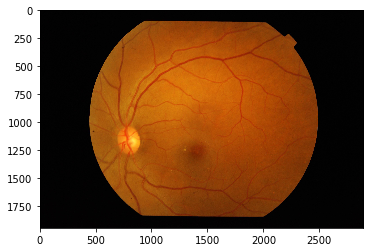
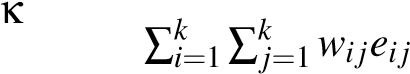


Figure 3: Sample of fundus photo from the dataset.

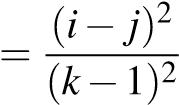
scores assigned by the human rater and the predicted scores. This metric varies from -1 (complete disagreement between raters) to 1 (complete agreement between raters). The definition of κ is:

*k*

*j*=1*wijoij*

= 1−*,* (1)

where *k* is the number of categories, *oij*, and *eij* are elements in the observed, and expected matrices respectively. *wij* is calculated as following:

*wij* *,* (2)

Due to Cohens Kappa properties, researchers must carefully interpret this ratio. For instance, if we consider two pairs of raters with the same percentage of an agreement, but different proportions of ratings, we should know, that it will drastically affect the Kappa ratio.

Another problem is the number of codes: as the number of codes grows, Kappa becomes higher. Also, Kappa may be low even though there are high levels of agreement, and even though individual ratings are accurate. All things mentioned above make Kappa a volatile ratio to analyze.

The main reason to use the Kappa ratio is that we do not have access to labels of validation and test datasets. Kappa value for these datasets is obtained by submitting our model and runner’s code to the checking system on the Kaggle site. Moreover, we do not have explicit access to images from the test dataset.

Along with the Kappa score, we calculate macro F1- score, accuracy, sensitivity, specificity on holdout dataset of 736 images taken from APTOS2019 training data.

# METHOD

The diabetic retinopathy detection problem can be viewed from several angles: as a classification problem, as a regression problem, and as an ordinal regression problem (Ananth and Kleinbaum, 1997).

A. Our method of approach – Brief introduction The first step is to input the images to the convolutional neural network (refer Fig.1). But we are not training the network straightaway with the input images. We need to do some preprocessing before training the model. The need for preprocessing is to improve the image data for enhancing the features of the image and to filter the random noises for better processing. Then we resize all the images downscale to level up images of different dimensions to a single dimension. After preprocessing phase is completed, the next step is to go for feature selection and feature extraction. Feature selection involves selecting a subset of features from the total image features in a way that this subset is itself enough to classify the images as one with Hemorrhage-Microaneurysms and as one without it which is normal. Extraction of features involves reducing the dimension of the selected subset of features. This is to reduce the size of input that is going to be fed for training the model. Feature extraction attempts to eliminate the redundant pixels or information from the selected subset of features so as to make it more suitable for further processing

## Preprocessing

1. Resizing the images The need for resizing is to convert all the varying dimensional (height and width) images to a single dimension. Because the training algorithm takes images which are of same height and width as input for better processing and thus reduces the processing time. Here in our dataset we have random dimensionality images such as 228x221, 998x630 and so on. Hence we resize the images to a height of 96 and width of 96 with depth as 3 that is three channels since we are using color images. 2. Noise Filtering Noise is nothing but the random variation in color or brightness of the image. Here, the noise present in our image would be speckle noise which is a kind of noise that occurs due to improper illumination of light while capturing the medical images. So to remove the noise, median filter is used. This filter works by taking a window of size 3x3 and replaces the pixel value by the median of the values residing in the window. One advantage of median filter is that it preserves the edges while removing the noise. D. Feature selection, Feature extraction, Model training and testing using CNN Before we pass our preprocessed image to the CNN, we need to convert the image to an array and map that array values in the range of 0 to 1. We set the epoch as 235 to attain a deep network. The initial learning rate is kept as 1e3 which is the default value for Adam optimizer and the batch size is 32. To train our model with more images since we have only 190 images for training, we generate more images from the existing dataset by passing parameters such as rotation range, width shift range, height shift range, shear range, zoom range and horizontal flip to the Image Data generator. 1) Convolutional neural network CNN comes under deep learning where it simulates the actual functioning of the brain neurons which capture and recognize objects. The CNN consists of three layers namely the convolution layer, the pooling layer and the fully connected layer. We are using Smallervggnet architecture of CNN which performs really well on image recognition and classification. The vgg network uses 3x3 convolution window followed by max pooling and finally end up with fully connected layer prior to a softmax classifier. 2) Convolution layer The convolution layer builds an image recognition classifier whose bias and weights are based on the concept of gradient descent. That means the activation map the layer builds should be able to classify the image properly at the same time it should be of less error. After constructing the activation map of size 3x3, it slides the window over the entire image matrix which is called convolving and replaces the image matrix with the new values obtained from convolving. The output is passed to the pooling layer. 3) Pooling layer The pooling layer reduces the spatial volume of the output and we use Max pooling which means we take a pool of size 3x3 from the image matrix (output of 1st layer) and finds the highest number in the pool and replace the entire pool with that highest number. Thus by this way, we reduce the spatial volume. For first two passes we keep the pool size as 3x3 and gradually decreasing it to 2x2 in the next pass. 4) Fully connected layer The fully connected input layer takes the output of the previous layers and flattens them to a single vector value. Then it applies weights to the features to classify the labels. At last it calculates probability for each of the labels. The label with high probability will be matched for the given input image. Thus by this way, we can get our detection and classification done. 5) Dropout Dropout states how many nodes of the current layer need to be cut off or disconnected from the next layer. This is to ensure the redundancy exists naturally in the model so that no one specific node predicts the certain class of the image. 6) Training the model On the whole, we first update the input shape of image as channel first. Then for Smallervggnet we pass parameters such as the convolution window size as 3x3, the number of filters it should built out of it as 32,for the next pass its 64 and for the next pass its 128 and the padding value is marked 'same' to preserve the dimension of the output as same as the input size. Next, the activation function used here ReLu, the Rectified Linear Unit, takes values which are either positive or zero. It also increases the non-linearity of the image. Then, we do batch normalization followed by max pooling of pool size 3x3. At last, the dropout is marked as 0.25 for convolution and pooling layer and it is 0.5 for fully connected layer. Dropout helps to control overfitting. Finally we end up with softmax classifier to return our predicted probabilities for each class label. 7) Testing the model The same process starting from preprocessing till the last layer of CNN is followed again for the test images. Now an operation is performed with the output of the testing image values from the fully connected layer with the values of all filters that was constructed previously in the training part to calculate the probability. The highest probability value is taken as the correct match. Finally we segment the two features from the abnormal images

## Data augmentation

We used online augmentations, at least one augmentation was applied to the training image before inputting to the CNN. We used following augmentations from Albumentations (A. Buslaev and Kalinin, 2018) library: optical distortion, grid distortion, piecewise affine transform, horizontal flip, vertical flip, random rotation, random shift, random scale, a shift of RGB values, random brightness and contrast, additive Gaussian noise, blur, sharpening, embossing, random gamma, and cutout (Devries and Taylor, 2017).

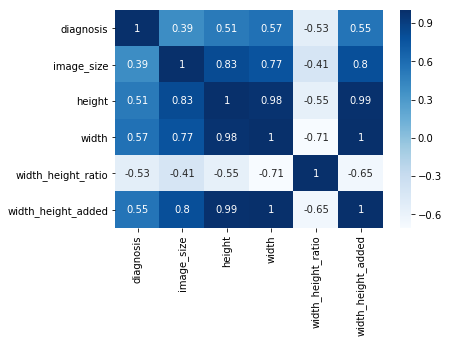


Figure 4: Spurious correlations between meta-features and diagnosis.

## Network architecture

We aim to classify each fundus photograph accurately. We build our neural networks using conventional deep CNN architecture, which has a feature extractor and smaller decoder for a specific task (head).

However, training the encoder from scratch is difficult, especially given the small amount of training data. Thus, we use an Imagenet-pretrained CNNs as initialization for encoder (Iglovikov and Shvets, 2018).

We propose the multi-task learning approach to detect diabetic retinopathy. We use three decoders. Each is trained to solve its task based on features extracted with CNN backbone:

* classification head,
* regression head,
* ordinal regression head.

Here, classification head outputs a one-hot encoded vector, where the presence of each stage is represented as 1. Regression head outputs real number in the range [0*,*4*.*5), which is then rounded to an integer that represents the disease stage. For the ordinal regression head, we use the approach described in (Cheng, 2007). Briefly, if the data point falls into category *k*, it automatically falls into all categories from 0 to *k*−1. So, this head aims to predict all categories up to the target. The final prediction is obtained by fitting a linear regression model to outputs of three heads. Neural network structure is shown in Figure 5.

We train all heads and the feature extractor jointly in order to reduce training time. We keep the linear regression model frozen until the post-training stage.

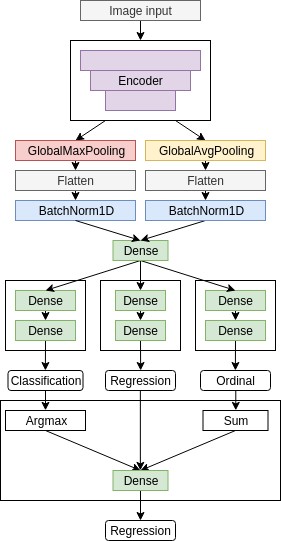


Figure 5: Three-head CNN structure.

## Training process

We use a multi-stage training process with different settings and datasets in every stage.

### Pretraining

We found out that labeling schemes are inconsistent between datasets, so we decided to use the largest one (2015 data) to pretrain our CNNs. Using transfer learning is possible because the natural features of the diabetic retinopathy are consistent between different people and do not depend on the dataset.

In addition, different datasets are collected on different equipment. Incorporation of this knowledge into the model increases its ability to generalize and elevates the importance of natural features by reducing sensitivity to instrument noise.

We initialize feature extractor with weights from Imagenet-pretrained CNN. Heads are initialized with random weights (He et al., 2015). We train a model for 20 epochs on 2015 data with minibatch-SGD and cosine-annealing learning rate schedule (Loshchilov and Hutter, 2016).

Every head is minimizing its loss function: crossentropy for classification head, binary cross-entropy for ordinal regression head, and mean absolute error for regression head.

After pretraining, we use encoder weights as initialization for subsequent stages. In our experiments, we observed the consistent improvement of metrics when we substituted weights of heads with random initialization before the main training, so we discard trained heads.

### Main training

The main training is performed on 2019 data, IDRID, and MESSIDOR combined. Starting with weights obtained in the pretraining stage, we performed 5-fold cross-validation and evaluated models on the holdout set.

At this stage, we change loss functions for heads: Focal Loss (Lin et al., 2017) for classification head, binary Focal Loss (Lin et al., 2017) for ordinal regression head and mean-squared error for regression head.

We trained each fold for 75 epochs using Rectified Adam optimizer (Liyuan Liu, 2019), with cosine annealing learning rate schedule. To save pretrained weights while new heads are in a random state, we disabled training (froze) of the encoder for five epochs while training heads only.

During the main training, we monitor separability in feature space generated by the encoder. We generate 2-dimensional embeddings with T-SNE (van der Maaten and Hinton, 2008) and visualize them in the validation phase for manual control of training performance. Figure 6 shows T-SNE of embeddings labeled with ground truth data and predicted classes. From the picture, it can be seen that images with no signs of DR are separable with a large margin from other images that have any sign of DR. Additionally, stages of DR come sequentially in embedding space, which corresponds to semantics in real diagnoses.

### Post-training

In the post-training stage, we only fit the linear regression model to outputs of different heads.

We found it essential to keep it from updating during previous stages because otherwise, it converges to the suboptimal local minima with weights of two heads close to zero. These coefficients prevent gra-

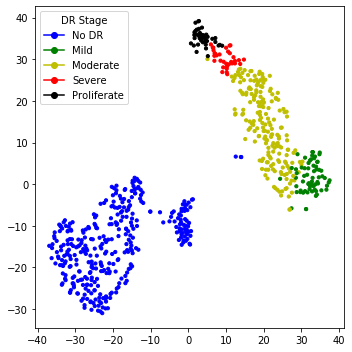
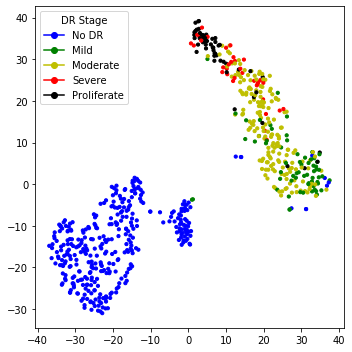


Figure 6: Feature embeddings with T-SNE. Ground truth (top) and predicted (bottom) classes. Best viewed in color.

dients of updating corresponding heads’ weights and further discourage network of converging.

Initial weights for every head were set to 1*/*3 and then trained for five epochs to minimize mean squared error function.

Difference between prediction distributions of regression head and linear regression outputs is show on Figure 7.

### Regularization

At training time, we regularize our models for better robustness. We use conventional methods, e.g., weight decay (Krogh and Hertz, 1992) and dropout. Also, we penalize the network for overconfident predictions by using label smoothing (Szegedy et al., 2016).

Additionally to label smoothing for classifica-

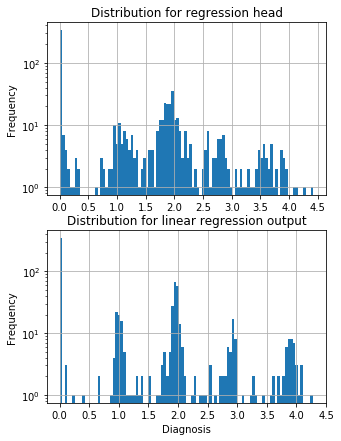


Figure 7: Output distributions for regression head and combination of heads

tion and ordinal regression heads, we propose label smoothing scheme for linear regression head. It can be used if it is known that underlying targets are discrete. We add random uniform noise to discrete targets:

*Ts* = *T* +∆

∆∼*U*(*a,b*)

Where *Ts* is smoothed target label, *T* is the original label, and *U* is the uniform distribution. In this case,  and *Ti Ti*+1 are neighbouring discrete target labels.

Applying this smoothing scheme, we could reduce the importance of wrong labeling.

### Ensembling

For final scoring, we ensembled models with 3 encoder architectures at different resolution that scored best on the holdout dataset : EfficientNet-B4 (380x380), EfficientNet-B5 (456x456) (Tan and Le, 2019), SE-ResNeXt50 (380x380 and 512x512) (Hu et al., 2017).

Our best performing solution is an ensemble of 20 models (4 architectures x 5 folds) with test-time augmentations (horizontal flip, vertical flip, transpose, rotate, zoom). Overall, this scheme generated 200 predictions per one fundus image. These predictions were averaged with a 0.25-trimmed mean to eliminate outliers from possibly overfitted models. A trimmed mean is used to filter out outliers to reduce variance.

We used Catalyst framework (Kolesnikov, 2018) based on PyTorch (Paszke et al., 2017) with GPU support. Evaluation of the whole ensemble was performed on Nvidia P100 GPU in 9 hours, processing 2.5 seconds per image.

# RESULTS

As experimental results, we provide two tables with metrics, which were mentioned in the Evaluation paragraph. The first table is about results that we have got from local validation without TTA (Table 1), and the second is with TTA (Table 2).

Our test stage was split into two parts: local testing and Kaggle testing. As we found locally, the ensembling method is the best one, and we evaluated it on Kaggle validation and test datasets.

On a local dataset of 736 images, ensembling with TTA performed slightly worse than without it. Ensemble with TTA performed better on the testing dataset of 13000 images as it has a better ability to generalize on unseen images.

Ensembles scored 0.818462/0.924746 validation/test QWK score for a trimmed mean ensemble without TTA and 0.826567/0.925466 QWK score for trimmed mean ensemble with TTA.

Additionally, we evaluated binary classification (DR/No DR) to check the best model’s quality as a screening method (see Tables 1 and 2, last row)

The ensemble with TTA showed its stability in the final scoring, keeping consistent rank (58 and 54 of

2943) on validation and testing datasets, respectively.

# INTERPRETATION

In medical applications, it is important to be able to interpret models’ predictions. As a good performance of the validation dataset can be a measure to select the best-trained model for production, it is insufficient for real-life use of this model.

By using SHAP (Shapley Additive exPlanations) (Lundberg and Lee, 2017), it is possible to visualize features that contribute to the assessment of the disease stage. SHAP unites several previous methods and represents the only possible consistent and locally accurate additive feature attribution method based.

Using SHAP allows ensuring that the model learns useful features during training, as well as uses correct features at inference time. Furthermore, in uncertain cases, visualization of salient features can assist the physician to focus on regions of interest where features are the most noticeable.

In Figure 8, we show an example visualization of SHAP values for one of the models from the ensemble. Red color denotes features that increase the output value for a given class, and blue color denotes features that decrease the output value for a given class. Overall intensity of the features denotes the saliency of the given region for the classification process.

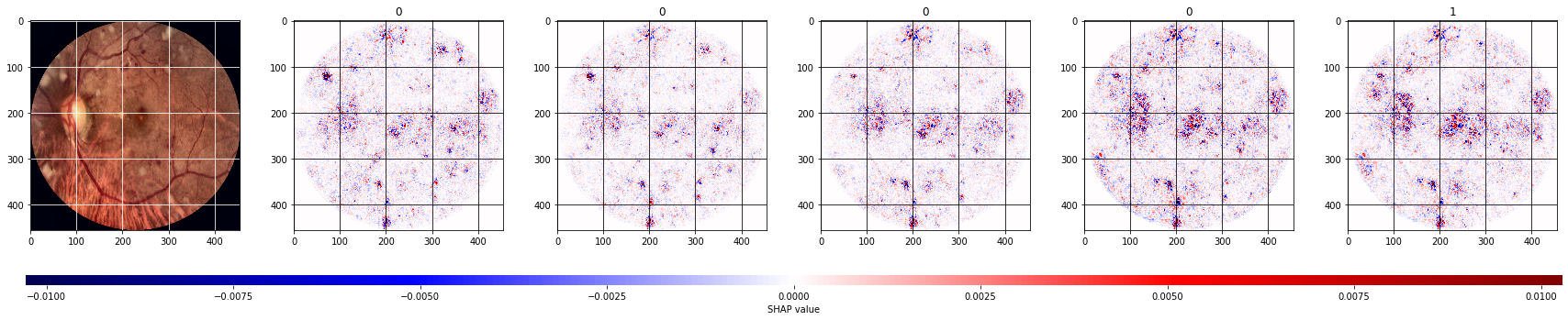
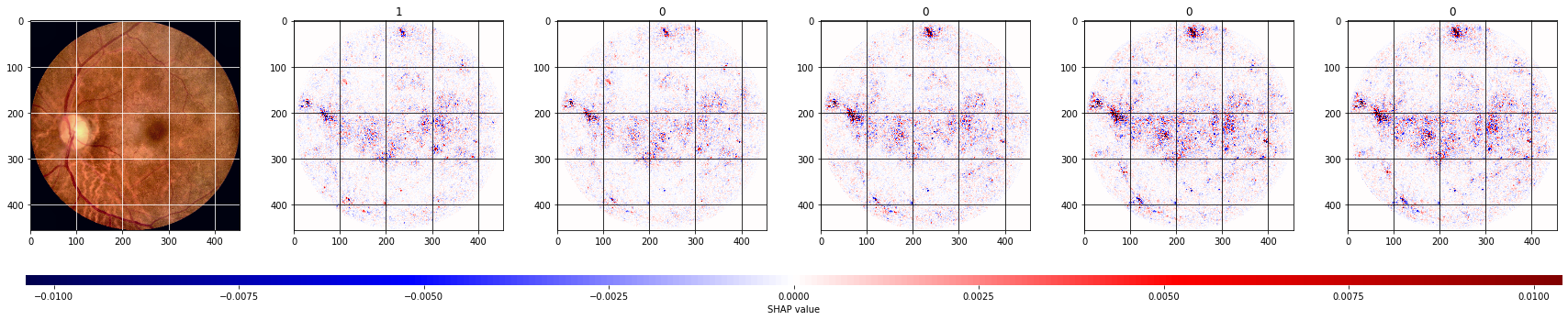


Figure 8: Shap analysis of sample images. Best viewed in color.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Model | QWK | Macro F1 | Accuracy | Sensitivity | Specificity | | EfficientNet-B4 | 0.965 | 0.811 | 0.903 | 0.812 | 0.976 | | EfficientNet-B5 | 0.963 | 0.815 | 0.907 | 0.807 | 0.977 | | SE-ResNeXt50 (512x512) | 0.969 | 0.854 | 0.924 | 0.871 | 0.982 | | SE-ResNeXt50 (380x380) | 0.960 | 0.788 | 0.892 | 0.785 | 0.974 | | Ensemble (mean) | 0.968 | 0.840 | 0.921 | 0.8448 | 0.981 | | Ensemble (trimmed mean) | 0.971 | 0.862 | 0.929 | 0.860 | 0.983 | | Ensemble (trimmed mean, binary classification) | 0.981 | 0.989 | 0.986 | 0.991 | 0.991 |   Table 1: Results of experiments and metrics tracked, without using TTA.   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Model | QWK | Macro F1 | Accuracy | Sensitivity | Specificity | | EfficientNet-B4 | 0.966 | 0.806 | 0.902 | 0.809 | 0.977 | | EfficientNet-B5 | 0.963 | 0.812 | 0.902 | 0.807 | 0.976 | | SE-ResNeXt50 (512x512) | 0.971 | 0.853 | 0.928 | 0.868 | 0.983 | | SE-ResNeXt50 (380x380) | 0.962 | 0.799 | 0.899 | 0.798 | 0.976 | | Ensemble (mean) | 0.968 | 0.827 | 0.917 | 0.828 | 0.980 | | Ensemble (trimmed mean) | 0.969 | 0.840 | 0.919 | 0.840 | 0.981 | | Ensemble (trimmed mean, binary classification) | 0.986 | 0.993 | 0.993 | 0.993 | 0.993 |   Table 2: Results of experiments and metrics tracked, with using TTA. |

# CONCLUSION

In this paper, we proposed the multistage transfer learning approach and an automatic method for detection of the stage of diabetic retinopathy by single photography of the human fundus. We have used an ensemble of 3 CNN architectures (EfficientNet-B4, EfficientNet-B5, SE- ResNeXt50) and made transfer learning for our final solution. The experimental results show that the proposed method achieves high and stable results even with unstable metric. The main advantage of this method is that it increases generalization and reduces variance by using an ensemble of the networks, pretrained on a large dataset, and finetuned on the target dataset. The future work can extend this method with the calculation of SHAP for the whole ensemble, not only for a particular network, and with more accurate hyperparameter optimization. Besides, we can do experiments using pretrained encoders on other connected to eye ailments tasks. Also, it is possible to investigate meta-learning (Nichol et al., 2018) with these models, but realized that it requires the separate in-depth research.