CSE PROJECT II@LIONS

Robust Training and Interpretability for Fundus Imaging

Author:
Jiahua Wu
SCIPER No. 293807

Supervisors:
Igor Krawczuk & Thomas Sanchez

January 10, 2020

ROBUST TRAINING AND INTERPRETABILITY FOR FUNDUS IMAGING

ABSTRACT

Machine learning (ML) has seen its growing application in medicine whereas its lack of robustness and interpretability halts the field from completely accepting it. To address this problem, we investigate the possibility of adversarial training as an effective way to enhance these two properties. Specifically, the experiments are conducted on a fundus imaging dataset dedicated for diabetic retinopathy detection to examine the practicability of the proposed method regarding on image-based diagnosis.

1 Introduction

ML has been creating a paradigm shift in medicine in recent years, from basic research (drug discovery and development) to clinical applications(diagnosis and prognosis of diseases such as diabetic retinopathy)[Edi19]. However, as revealed in [Sze+14], [Big+17], the ML models are vulnerable to adversarial attacks, where a human-imperceptible manipulation in the input can result in a different predictions or in medical terms a misdiagnosis, which could be utilized for fraudulent interests. This was recently demonstrated by a group of researchers who showed that a carefully calculated perturbation on an image of a benign skin mole that is imperceptible to the human eye can be misclassified as a malignant mole, with 100% confidence [Fin+19]. Additionally, the lack of interpretability is another obstacle that prevent ML from being fully trusted. Due to the black-box nature of decision-making process of the deep neural networks, the diagnosis given by the models are not always based on features used by medical professionals. (pathological tumors, inflamed tissues, etc.), making the decision process incomprehensible to humans.

Fortunately, literature on adversarial attacks [KGB16], [GSS15], [WJM17], [Mad+17] shows that (restricted to their choice of models in experiments) some widely used deep neural networks can be rendered resistant to the attacks if they are trained on corresponding adversarial examples. Particularly, in [Mad+17], Madry et al. claim the projected gradient (PGD) method with random starting point to be the strongest attack utilizing the local first order information (gradients) about the network. In [WJM17], Brendel et al. propose a decision boundary attack (DBA) method that can generate effective adversarial examples solely based on the final decision of the model. They are respectively representatives of white-box methods where we can manipulate the model to calculate gradients and black-box methods where we only have access to the output of the model.

Recently, researchers have demonstrated that the existence of adversarial examples results from the fact that the neural network makes use of some "non robust features" (subtle signals or patterns present in the inputs that are meaningless to humans) for classification [Ily+19]. Inspired by this observation, we suppose that through training on adversarial examples, the use of such non robust features will be penalized and the attention of the model should be shifted to the robust features that are equally utilized by humans, which makes the model more interpretable. We wish to verify this hypothesis by training some popular networks for image classification on adversarial examples and employ saliency map methods to visualize the importance the network attaches to each pixel.

2 Baseline Experiments

2.1 Dataset

The image dataset used in our work, which is obtained from the website of Kaggle diabetic retinopathy competition provided by EyePACS, contains 35126 high resolution fundus photographs taken under various imaging conditions. These fundus photographs have been labeled by a trained clinician with a scale of 0 to 4 based on the severity of DR. As a result, it is a typical five-class classification task where the model needs to correctly predict the severity of DR.

Besides, it is noteworthy that these images are taken by different types of cameras under diverse conditions. Some of the images are inverted while some are incomplete due to the variant microscope imaging procedure. Also, noise in both images and labels is unavoidable in the raw data set, which puts forward a high demand to the robustness of our classification system.

Last but not least, as shown in table 1, the dataset is highly imbalanced. Consequently, a "lazy" model which gives 0 as prediction all the time can achieve an accuracy of around 72%. To show the capability of the model in classifying minor classes, we include balanced accuracy in our performance metrics.

Label	Number of samples	Percentage
0	25810	73.5%
1	2443	6.9%
2	5292	15.1%
3	873	2.5%
4	708	2.0%

Table 1: *Distribution of original dataset*

2.2 Image Preprocessing and Data Augmentation

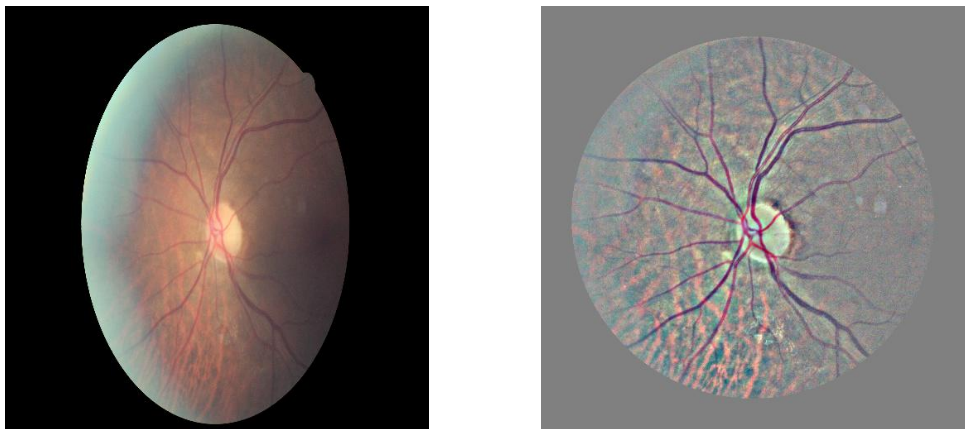
Image Augmentation. As mentioned above, the raw fundus images in the dataset have large variations in terms of brightness, contrast and the colour of eyeballs. Such irregularities constitute noise to the model and thus prevent it from effectively learning meaningful features. In order to standardise these images and reduce artifacts produced by irrelevant factors, we refer to the preprocessing scheme proposed by competition winner Graham. More precisely, we subtract each pixel value of images by the weighted means of its surrounding pixel values, and add it by 50% grayscale. This operation is similar to the 'high pass' processing in the PhotoShop software, which makes the blood vessels as well as the lesion areas in fundus images more explicit. Then, the fundus area in images will be clipped to 95% of the original size by covering a mask with a transparent circle on the center in order to remove the boundary effects arising from the last operation. Afterwards, pixel values of images are normalised to [-1 , 1]. Effects of the preprocessing procedure are demonstrated in figure 1.

Image Resize. The original images in the dataset are of different sizes around 3k×2k. Loading a dataset with images of such resolution exceeds the capacity of the gpu memory and the the inconsistency of the resolution makes it impossible to train a model using the raw images. Consequently, we resize them to 256×256. Initially, the resizing is done "on the fly", which means that the images are modified upon being read for training. This brought much repeated and unnecessary work and slowed down the training. For the sake of improvement, we resize the images and save them as a new dataset and directly train the model on it. In this way the training is sped up by around 73 times.

Data Augmentation. In order to enhance diversity of the dataset and therefore alleviate the problem of overfitting, we further apply various transformations to the images fed to the model during training, which includes:

- Randomly flip the images of left eye and right eye horizontally. Since human eyes are structurally mirror-symmetric, the fundus images can be flipped horizontally without affecting the label they correspond to.
- Randomly perform geometric transformation on images, including randomly scaling to 90%-110% of their size, translating by $\pm 0.05\%$ of the total width, rotating from -30 to 10 degrees and shearing from -10 to 10 degrees.
- Randomly change the brightness to 85%-115%, decreasing or improving the contrast in the range of 85%-115%.

All the steps of augmentation listed above are performed with a probability of 50%.



(a) *original image* **(b)** *processed image*

Figure 1: *Comparison between a typical original image and a typical preprocessed image. it can be found that fundus region is standardized to be a circular area, containing the optic disk, macula and main vessels. Furthermore, the background color of retina is faded, while the venules, hard exudates and hemorrhages are emphasised.*

2.3 Model Architecture

Model Architecture. For this typical image classification task, we employ there popular network structure families: resnet[He+15], efficientnet[TL19] and InceptionV3[Sze+15]. For the resnet family, resnet18 and resnet50 are used while for the efficientnet family we use efficientnet b0 - b3. Table2 shows the number of parameters of different networks. We also make use of transfer learning which consists in replacing the classification layer with a 5-unit one and in loading pretrained weights of the mentioned networks on Imagenet before the start of training.

Model	Number of Params
resnet18	11179077(∼11M)
resnet50	23518277(∼23M)
efficientnet b0	4013953(∼4M)
efficientnet b1	6519589(∼6M)
efficientnet b2	7708039(∼7M)
efficientnet b3	10703917(∼10M)
InceptionV3	24357354(∼24M)

Table 2: *Number of parameters of different models*

2.4 Evaluation Methods and Results

Train/Valid Split. To assess the performance of models tested, the original dataset is split into training set and validation set with a ratio of 7:3 and the resulting datasets are used throughout the whole series of experiments. Since the split is random, the resulting datasets approximately preserve the distribution of the original dataset(table3).

Performance Metrics. In order to assess the performance of models, we employ three metrics:

- **Usual Accuracy**. Defined as the fraction of right predictions, it is formally written as:

$$Accuracy = \frac{n_{correct\,predictions}}{n_{predictions}} \tag{1}$$

- **Balanced Accuracy**. Since the dataset is highly imbalanced, we introduce balanced accuracy to avoid inflated performance estimates on imbalanced datasets. It is the usual accuracy where each sample is weighted according to the inverse prevalence of its true class. More explicitly:
  If y_i is the true value of the i-th sample, and $w_i$is the corresponding sample weight, then we adjust the sample

weight to:

$$\hat{w}_i = \frac{w_i}{\sum_j 1\left(y_j = y_i\right) w_j} \tag{2}$$

where $1(x)$ is the indicator function. Given predicted $\hat{y}_i$ for sample i, the balanced accuracy is thus defined as:

$$\text{balanced-accuracy}\left(y, \hat{y}, w\right) = \frac{1}{\sum \hat{w}_i} \sum_i 1\left(\hat{y}_i = y_i\right) \hat{w}_i \tag{3}$$

In our case all classes are equally weighed and therefore $\hat{w}_i$ is reduced to $\frac{1}{n_i}$ where n_i is the number of samples of class i.

- **Cohen Kappa Score**. It is the metric adopted by the initial Kaggle competition and we introduce it here to compare the performance of our model with that of Kaggle competitors. It is used to measures the agreement between two raters (in our case, the model and the clinicians who estimate the DR level) and it typically varies from 0 (random agreement) to 1 (complete agreement). Mathematically, it is defined by:

$$\kappa = 1 - \frac{\sum_{i,j} w_{i,j} O_{i,j}}{\sum_{i,j} w_{i,j} E_{i,j}} \tag{4}$$

where i and j represent the labels (in our case vary from 0 to 4), O is the confusion matrix and w and E are defined by:

$$w_{i,j} = \frac{(i-j)^2}{(N-1)^2}, \quad E_{i,j} = n_i \times n_j \tag{5}$$

where n_i represents the number of samples of class i.

Training Setting. Training epochs are all set to be 20. Training is performed on 3 Tesla K40c gpus and the batch size is adjusted such that it is the greatest that does not exceed the memory capacity of the gpus. As for implementations, the balanced accuracy and cohen kappa score are calculated using corresponding sklearn functionalities and the deep learning library employed is Pytorch.

We test the model on the validation set after each epoch. The results shown in table4 are the ones at the best epoch(early stopping). Several observations can be made:

- In general, models trained and tested on images with resolution 512×512 has a better performance than those trained on images with resolution 256×256 or 299×299. This outcome can be explained by the fact that images with higher resolution contain more details and features.
- The efficientnet family, which is the fruit of an efficient scaling method for neural networks and achieves a better performance on multiple popular datasets than resnet as reported in [TL19], still outperforms the resnet family in our dataset with much less number of parameters.
- efficientnets with image size of 512×512 achieves a cohen kappa score comparable to the top teams in the kaggle competition leaderboard (for reference, the score of 10^{th} team is 0.814.). Credits should be given to the preprocessing and the data augmentation applied to images since their application helps to greatly improve the prediction accuracy (by around 20%, from 66.3% to 83.6%) and the balanced accuracy (by more than 10%, from 40.7% to 52.1%) compared to directly feed in the raw images.
- The implementation of efficientnet in Pytorch is not memory efficient thus the allowed batch size is much smaller than it should be, which results in a longer training time. This inefficiency of the implementation is also reflected on the inference time of the model: at the validation stage, resnet18 (4min27s for 11M parameters) is much faster than efficientnet b0 (6min42s for 4M parameters).

Considering training time and performance, we select efficient b0 as the baseline model and use image size of 512×512 for the following experiments.

Label	Number of samples	Percentage
0	18134	73.8%
1	1710	6.9%
2	3659	14.8%
3	603	2.5%
4	480	2.0%

(a)

Label	Number of samples	Percentage
0	7673	72.9%
1	733	6.9%
2	1632	15.5%
3	270	2.5%
4	228	2.2%

(b)

Table 3: *Distribution of training dataset(a) and validation dataset(b)*

	Image Size	Batch Size	Accuracy	Balanced Accuracy	Cohen Kappa Score	Training Time(20 epochs)
resnet18	256×256	128	81.2%	47.6%	0.701	2h40min
resnet18	512×512	64	83.6%	52.1%	0.760	8h26min
resnet50	256×256	64	81.9%	51.9%	0.725	5h59min
resnet50	512×512	64	82.6%	46.7%	0.725	19h25min
efficientnetb0	256×256	192	81.5%	44.0%	0.696	2h29min
efficientnetb0	512×512	16	**85.2%**	58.6%	0.805	14h59min
efficientnetb1	256×256	192	81.7%	45.5%	0.706	2h29min
efficientnetb1	512×512	16	85.0%	58.7%	0.805	19h15min
efficientnetb2	256×256	96	82.3%	54.9%	0.739	2h20min
efficientnetb2	512×512	16	**85.2%**	**61.0%**	**0.811**	19h49min
efficientnetb3	256×256	96	82.8%	53.9%	0.746	5h50min
efficientnetb3	512×512	24	85.1%	58.5%	0.803	10h1min
InceptionV3	299×299	64	81.6%	49.7%	0.706	7h12min

Table 4: *Performance of different models on validation set measured by usual accuracy, balanced accuracy and cohen kappa score*

3 Dataset Balancing

The model obtained in section2 is trained on an imbalanced dataset. Therefore, the distribution of the prediction is also imbalanced and significantly leans to the major class (i.e. class with label 0). This effect can be seen in figure2(a) which is the confusion matrix of efficient b0 tested on the validation set. In practice, this implies that patients with with DR would be wrongly diagnosed as Non-DR, which would cause a delay of treatment and aggravate the illness. From the point of view of robustness, the decision boundary of the dominant class is considerably large and accordingly the one of the minor class is tiny and brittle: a small perturbation to the images with minor label leads to a change in prediction, which makes the model non robust.

In order to alleviate this model learning bias, we extensively experiment with two families of dataset-balancing methods: Modify loss to give more emphasis on the minority classes and rebalance the class prior distributions in a pre-processing procedure. In detail,

- Loss Modification
  - **Focal Loss**[Lin+18]. It is initially invented for model training of object detection where in the training dataset the background objects are overrepresented and thus cause the dataset to be imbalanced. It is given (in binary case) as:
$$\mathrm{FL}\left(p_{\mathrm{t}}\right)=-\left(1-p_{\mathrm{t}}\right)^{\gamma} \log \left(p_{\mathrm{t}}\right) \tag{6}$$
    where
$$p_{\mathrm{t}}=\begin{cases} p & \text{if } y=1 \\ 1-p & \text{otherwise} \end{cases} \tag{7}$$

In the above $y \in \{\pm 1\}$ is the ground-truth class and $p \in [0,1]$ is the model's estimated probability for the class with label y = 1. Recall that cross entropy loss is defined as $\mathrm{CE}(p,y) = \mathrm{CE}(p_{\mathrm{t}}) = -\log(p_{\mathrm{t}})$, focal loss is no more than adding a factor $(1-p_{\mathrm{t}})^{\gamma}$ to the standard cross entropy criterion. Setting $\gamma > 0$ reduces the relative loss for well-classified examples ($p_{\mathrm{t}} > 0.5$), putting more focus on hard, misclassified examples. According to confusion matrix of the baseline model (figure2(a)), the examples of overrepresented class(level 0) are the best-classified ones and thereby this loss smoothly reduces the relative loss of them and raises importance of the minor classes.

– **Class balanced loss**[Cui+19]. The loss of images is reweighed by inverse of the effective number of samples. The effective number of samples is defined as:

$$E_n = \frac{1-\beta^n}{1-\beta}, \quad \beta = \frac{N-1}{N} \tag{8}$$

where n is the number of samples of a class. The resulting loss is therefore given (in binary case) as:

$$\mathrm{CB}(p_{\mathrm{t}}) = \frac{1}{E_n}\mathrm{CE}(p_{\mathrm{t}}) = -\frac{1-\beta}{1-\beta^n}\log(p_{\mathrm{t}}) \tag{9}$$

When β tends to 1, this loss tends to the inverse frequency loss and consequently, by modifying β we are able to smoothly adjust the weights of each class while preserving the distributional properties of the dataset.

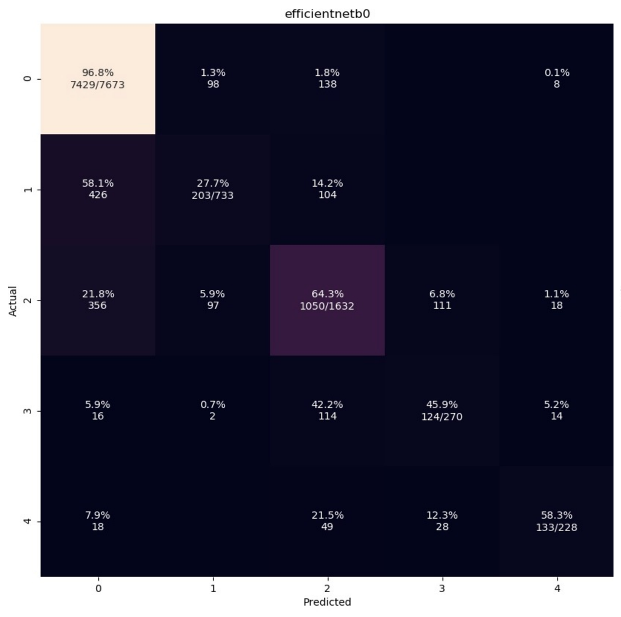
– **Inverse Frequency Loss**. The loss of images is reweighed by the inverse of the frequency of the class they belong to and thus more importance will be attached to minor classes.

- Dataset Balancing
  – **Oversampling**. Add repeated minor class samples until all classes have the same number of samples. The resulting dataset in our case counts 103240(25810*5) samples in total. As no new information is added during this process, the model easily suffers from overfitting due to excessively visiting the repeated samples.
  – **SMOTE[chawla_smote_2002]**. SMOTE stands for Synthetic Minority Over-Sampling Technique. It consists in populating the minor classes by adding synthetic samples. Synthetic samples are generated by the linear combination of samples with their own k-nearest neighbours that belong to the same class.

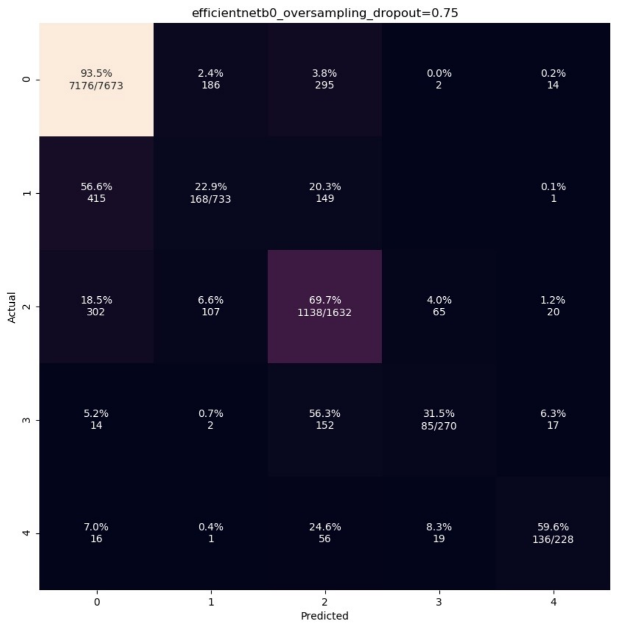
3.1 Results

The results of various methods are listed in table5. The setting is the same as section2, the model used is efficientnet b0 and its performance on the original imbalanced dataset is considered to be the baseline. Some observations are made:

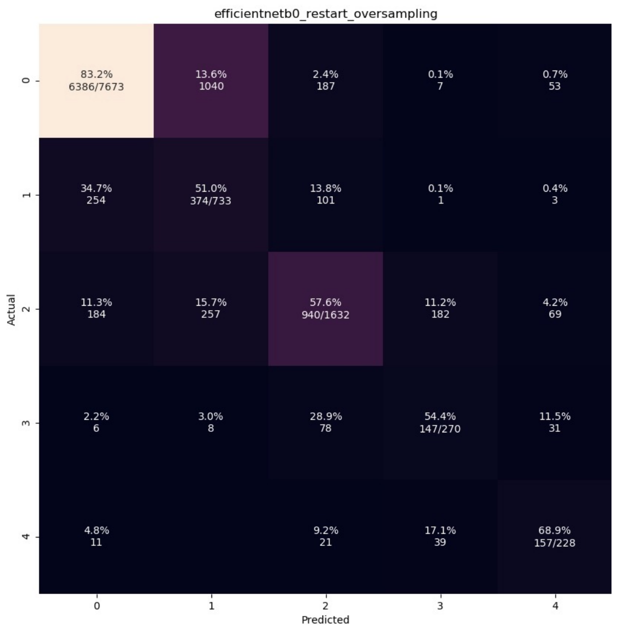
- In spite of data augmentation, training on an oversampled dataset exhibits model overfitting probably owing to repeatedly visiting duplicated samples. Overfitting causes a sharp drop in model performance. Although the problem of overfitting can be mitigated through dropout, judging from the confusion matrix (figure2(b)), the resulting model has a distribution of decision similar to the one trained on imbalanced dataset but with a lower accuracy in predicting the major class. In addition, due to drastic increase of number of samples (from 20k to 100k), the training time climbs considerably.
- If one starts the training from a pre-trained weight obtained from the above baseline experiments and applies oversampling, the model still overfits but at the first few epochs its decision distribution becomes more balanced(figure2(c)) at the expense of prediction accuracy of the major class and the overall accuracy stays at an acceptable level. The shown confusion matrix is that of the model obtained after 1 epoch of training.
- Model performance significantly drops after being trained using SMOTE. This deterioration can be explained by the fact that SMOTE creates new samples using linear combination, which is not meaningful for images whose sample space is far more complex than a linear space and whose relation with labels is by no means linear. Such oversimplification leads to a massive mislabeling for newly generated samples and it comes as no surprise that the resulting model has a poor performance.
- On the contrary, training with a reweighed loss function helps to balance decision distribution without overfitting problem and the training is notably faster than oversampling. Overall, focal loss with γ=2 and class balanced loss with β=0.99 would be good choices among all tested loss functions since they yield the highest balanced accuracy with merely a minor loss in usual accuracy(less than 0.1% for the former and 0.4% for the latter). According to the confusion matrix they result in (figure3(b, c)), they successfully force the model to attach more importance to minor classes, which results in a better accuracy in predicting certain minor classes. However, they have little influence on balancing the decision distribution and it can still be found that the model tends to give 0 as prediction.



(a)

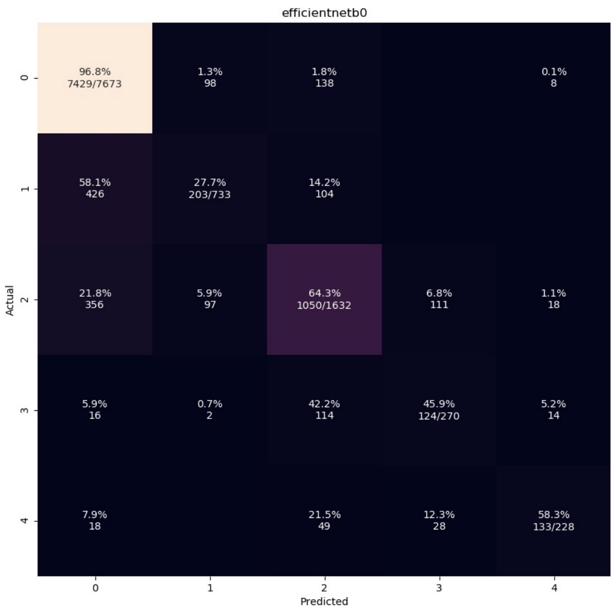


(b)

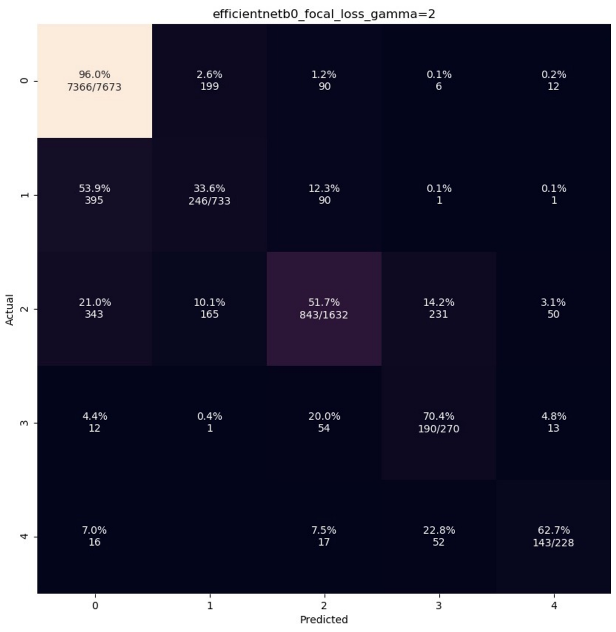


(c)

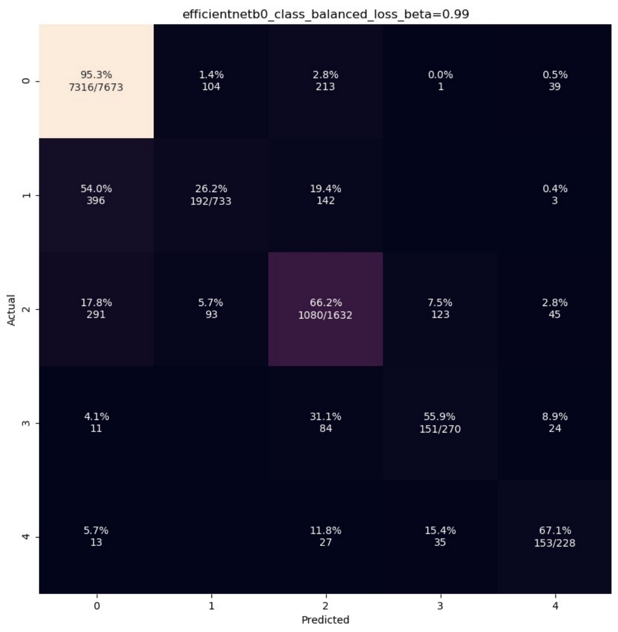
Figure 2: *Comparison of confusion matrices of efficient b0 trained on imbalanced dataset(a), on oversampled balanced dataset with dropout rate=0.9(b), on oversampled balanced dataset starting from pre-trained weights(c). It can be found from (a) that most of the samples of class "0" and "2" are classified correctly, as the number of these samples are basically clustered on the diagonal line. This is due to the fact that these two classes are the most prevalent in the dataset. However, there are relatively more samples of class "1" misclassified to be class "0" probably because the early stages of DR are more difficult to be correctly distinguished and the class-imbalanced training data (where most images are labeled with "0"), to a certain extend, mislead the prediction. The classification result of images with label '3' and '4' is not satisfactory either, presumably due to the lack of training samples. Same prediction distribution pattern can be found in (b). However, in (c), when the pretrained model is trained on oversampled dataset for one epoch, the prediction distribution becomes more balanced compared to (a) since much less images are assigned to level 0 and accuracy of the minor classes notably improves (10% for level "3" and level "4", 24% for level "1") .*



(a)



(b)



(c)

Figure 3: *Comparison of confusion matrices of efficient b0 trained on imbalanced dataset(a), on imbalanced dataset using focal loss with γ=2(b) and on imbalanced dataset with β=0.99(c). Compared to (a), the prediction accuracy of minor classes (especially "3" and "4") in (b) and (c) improves, which proves that the modification of loss function drives the model to focus more on minor classes. However, reweighing also does harm to the model performance on major classes. In both (b) and (c) the accuracy on level 0 slightly drops. In particular in (b) we spot a drop of accuracy for level 2 and in (c) a decrease of accuracy for level 1 can be found. Overall, loss function reweighing has no obvious effect on prediction distribution as it stays skewed towards level 0.*

	Batch Size	Accuracy	Balanced Accuracy	Cohen Kappa Score	Training Time (20 epochs)
crossentropy(baseline)	16	85.2%	58.6%	0.805	14h59min
focal loss@γ=1	48	84.7%	58.7%	0.800	6h02min
focal loss@γ=2	48	84.8%	**62.9%**	0.814	6h08min
focal loss@γ=3	48	84.3%	60.2%	**0.815**	6h12min
focal loss@γ=4	48	84.2%	59.3%	0.803	5h50min
class balanced loss@β=0.9	48	84.8%	59.7%	0.809	6h05min
class balanced loss@β=0.99	48	**85.2%**	62.1%	0.802	6h09min
class balanced loss@β=0.999	48	84.8%	60.5%	0.807	6h20min
inverse frequency loss	16	79.8%	59.6%	0.764	14h30min
oversampling+dropout=0	16	70.4%	39.9%	0.612	46h39min
oversampling+dropout=0.75	48	82.6%	55.5%	0.775	21h37min
oversampling+dropout=0.9	48	78.0%	57.8%	0.780	20h05min
oversampling+restart	48	76.0%	63.0%	0.765	5h4min (5 epochs)
SMOTE	16	21.11%	50.38%	0.364	57h37min

Table 5: *Performance of efficient b0 under different balancing methods. For SMOTE, the sample generation time is included in training time.*

4 Training with Corrupted Images

In view of equipping the model with robustness against low-quality or even corrupted images, we train the model on a dataset of images that undergo different types of corruptions(light leakage, motion blur, under expose and over expose) at different levels (from 1 to 4, the higher the level, the worse the images). The effects of corruptions are demonstrated in figure4. The corruption processing is Zeiss-proprietary.

The training is proceeded using the same training-validation split as the above sections. The baseline model is efficientnet b0, the images are of size 512×512, the batch size is set to be 48 and dropout is not used.

Although the underlying corruptions can be served as a new data augmentation and be applied newly to each image at each epoch, due to the lengthy image processing which takes around 50 seconds per image, we process the images of the (oversampled) original dataset and save them as new datasets. No other data augmentation is added. The corrupted datasets are then directly loaded for training without any further operation. During the generation of the corrupted datasets, four types of corruptions are randomly applied with random parameters that conform to their randomly chosen level in the level range.

Test on original validation set. The results on the original imbalanced dataset and oversampled balanced dataset obtained after 20 epochs and validated on the untouched imbalanced validation set are respectively summarised in table6 and table7. Compared with model trained on imbalanced dataset, we notice a drop in performance of the model because of the loss of information due to corruption. Nonetheless, when acting on an oversampled balanced dataset, the corruptions play a role of an enhanced data augmentation that mitigates the problem of overfitting.

Test on corrupted validation set. To check if training on a corrupted dataset can bestow resistance to corruption on the model, we further test the models on corrupted imbalanced validation dataset whose corruption level range is 1 – 4. The results are listed in table8.

As revealed in the results, training on a corrupted dataset allows the model to gain robustness against the corruptions of the same types.

Resistance against adversarial attack. To investigate the ability of the corrupted-trained model to withstand an adversarial attack, we record the success rate of an attack at different "box size" ϵ of the l_∞ PGD attack[Mad+17]. PGD attack is in fact projected gradient method on the negative loss function with respect to the input image. Mathematically, the iterative scheme is defined as:

$$x^{t+1} = \Pi_{x+\mathcal{S}}\left(x^t + \alpha\,\mathrm{sgn}\left(\nabla_x L(\theta, x, y)\right)\right) \tag{10}$$

where $x + S$ is the admissible set on which the resulting adversary is projected. Therefore, S determines how much the adversary can deviate from the original image. Here we define S as a l_∞ ball with size ϵ. As a result, the higher ϵ is, the higher deviation is allowed and thus the stronger the attack is. In other words, we can control the intensity of the attack by controlling ϵ.

The success rate of an attack is defined by the number of the change of predictions after the attack over the total number of samples:

$$success\, rate = \frac{number\, of\, prediction\, changes}{total\, number\, of\, predictions} \tag{11}$$

At the same ϵ, if the success rate of an attack is higher, the model is considered to be weaker against the attack.

The experiment is performed on 240 samples randomly chosen from the original validation set and the number of steps of PGD attack is set to be 5.

Results are presented in figure5 and for the sake of clarity, values of ϵ shown are all multiplied by 255. As expected, the success rate of the attack increases with ϵ and it is much higher on corruption-trained models than on baseline model at all tested ϵ, which implies that training on corrupted dataset does not seem to help the model gain resistance against gradient-based adversarial attacks. On the contrary, the corruption-trained model becomes more vulnerable in face of such attacks. This behaviour might be due to the fact that corrupted images still contain some non-robust features while lose some robust features of the original images. Consequently, compared with baseline, the model learns some non-robust features of the corrupted images and at the same time have no access to certain robust features of the original dataset, which results in a worse performance on adversarial samples generated from the original dataset. It would be better if the model is trained on a mixed dataset where both original and corrupted images are present or include the corruption into the data augmentation routine.

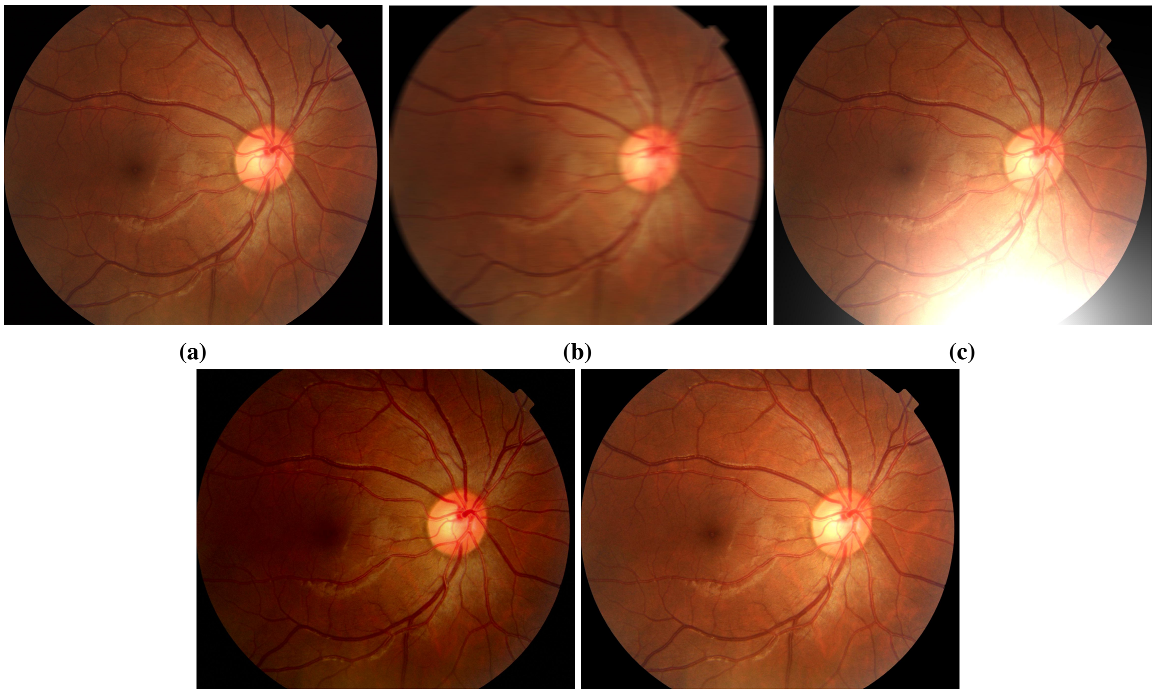


Figure 4: *Different kinds of corruptions on a test image(a): motion blur(b), light leakage(c), under expose(d) and over expose(e).*

Corruption Level Range	Accuracy	Balanced Accuracy	Cohen Kappa Score
baseline	85.2%	58.6%	0.805
1	81.2%	47.5%	0.701
1 – 2	81.7%	49.7%	0.725
1 – 3	82.6%	50.0%	0.718
1 – 4	81.7%	51.5%	0.731

Table 6: *Performance of efficient b0 trained on imbalanced dataset at different level ranges*

Training Corruption Level Range	Accuracy	Balanced Accuracy	Cohen Kappa Score
baseline	70.4%	39.9%	0.612
1	80.6%	47.7%	0.705
1 – 2	81.2%	47.1%	0.703
1 – 3	80.7%	48.1%	0.707
1 – 4	80.3%	46.6%	0.695

Table 7: *Performance of efficient b0 trained on balanced dataset at different level ranges*

Training Corruption Level Range	Accuracy	Balanced Accuracy	Cohen Kappa Score	Accuracy Drop
baseline	76.4%	39.1%	0.501	8.8%
1	77.8%	41.7%	0.575	3.4%
1 – 2	78.5%	44.6%	0.619	3.2%
1 – 3	77.6%	39.3%	0.586	5.0%
1 – 4	79.1%	40.3%	0.598	2.6%

Table 8: *Performance of corrupted-trained efficient b0s on corrupted(1 – 4) imbalanced validation dataset*

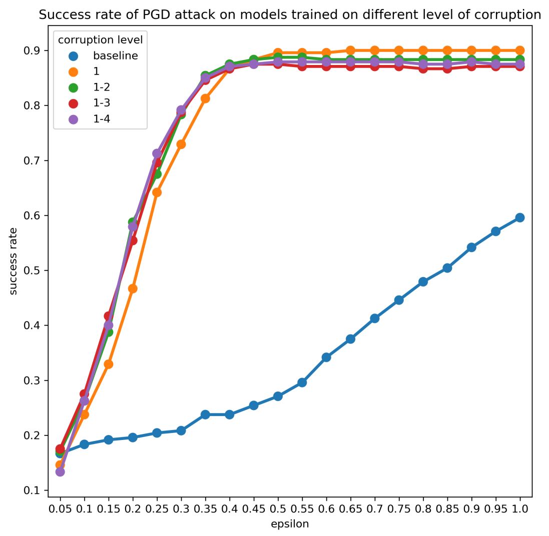


Figure 5: *Success rate of PGD attack at different ϵ for different models*

5 Adversarial Training

Further experiments of section4 show that the success rate of the PGD attack on the baseline model raises to more than 80% when ϵ is set to be more than 2.5. Consequently, we adopt 2.5 as the box size and 5 as the number of iteration steps in our adversarial training to ensure that adversarial samples can be effectively generated while avoid too much model performance loss on benign dataset.

The training scheme is as follows: for each batch, following the forward pass of the original samples and the calculation of loss, corresponding adversarial samples are generated and fed to the network and the adversarial loss is calculated. These two losses are equally contributed to the final loss for the backpropagation.

Results for performance of the model on the normal dataset and estimation of resistance against PGD attack are respectively presented in table9 and figure6. Table9 shows that the adversarially trained model undergoes an overall drop in performance on original dataset while figure6 shows that adversarial training with PGD attack helps the model to gain robustness against the same type of attack. However, it remains to check if the model is equally resistant to other types of attack such as decision boundary attack.

Furthermore, we use integrated gradients(IG)[STY17] to study feature importance for this network. The number of steps in the Riemman approximation of the integral chosen is 100 and the pixel values of baseline input are randomly generated from a uniform distribution between 0 and 1. The visualisation of IG of both baseline model and adversarially trained model evaluated on two images(one with level 0 and one with level 4) from the original dataset are shown in figure7 and figure8. Integrated gradients are aggregated along the colour channel and are overlaid on the faded actual image with positive attributions along the green channel and negative attributions along the red channel. As depicted in the figure7, the model trained with adversarial samples focuses on some structural features like vessels while the baseline model includes some random parts of the image into its decision process. Similarly in figure8, the adversarially trained model predicts the fundus to be DR at level 4 based on the damage of the vessels. These visualisations are in agreement with our hypothesis that adversarial training penalises use of non-robust features and therefore drives the model to focus more on robust features.

Training Corruption Level Range	Accuracy	Balanced Accuracy	Cohen Kappa Score
baseline	85.2%	58.6%	0.805
adv-trained@ϵ=2.5	80.0%	46.4%	0.668

Table 9: *Performance of adversarially trained model on original validation dataset*

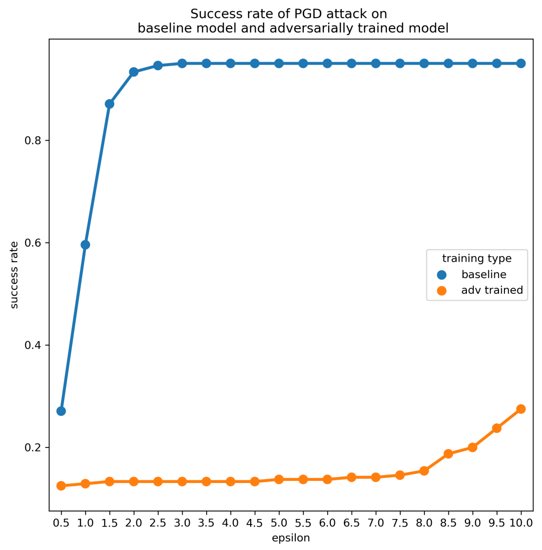


Figure 6: *Success rate of PGD attack at different ϵ for baseline model and adversarially trained model*

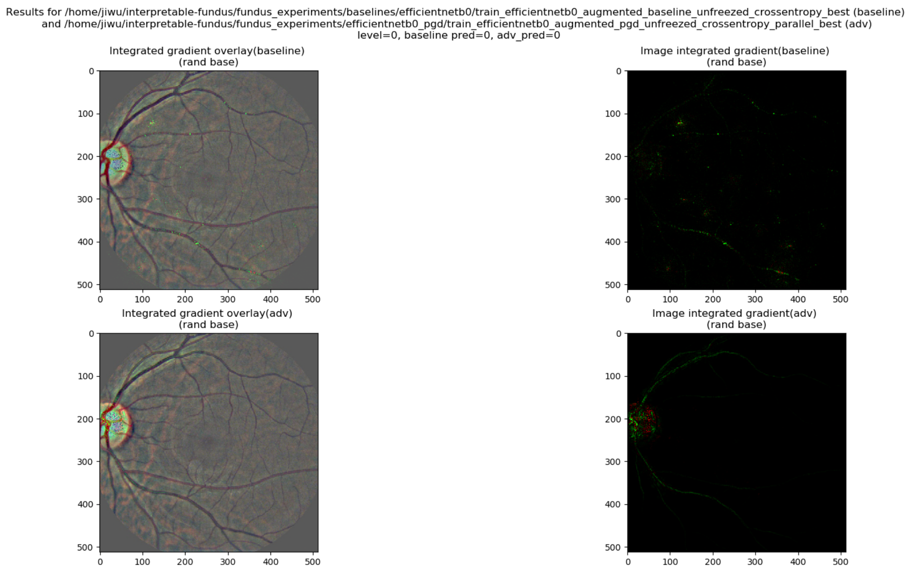

Results for /home/jiwu/interpretable-fundus/fundus_experiments/baselines/efficientnetb0/train_efficientnetb0_augmented_baseline_unfreezed_crossentropy_best (baseline)
and /home/jiwu/interpretable-fundus/fundus_experiments/efficientnetb0_pgd/train_efficientnetb0_augmented_pgd_unfreezed_crossentropy_parallel_best (adv)
level=0, baseline pred=0, adv_pred=0
Integrated gradient overlay(baseline) (rand base)
Image integrated gradient(baseline) (rand base)
Integrated gradient overlay(adv) (rand base)
Image integrated gradient(adv) (rand base)

Figure 7: *Integrated gradients of PGD attack-trained model and baseline model evaluated on an image of level 0 (healthy) of the original dataset. The IG distribution of baseline model is noisy, implying that it depends its prediction partially on some random areas whilst that of the adversarially trained model is clean and coincides with vessels, showing that the model makes prediction based on these meaningful features*

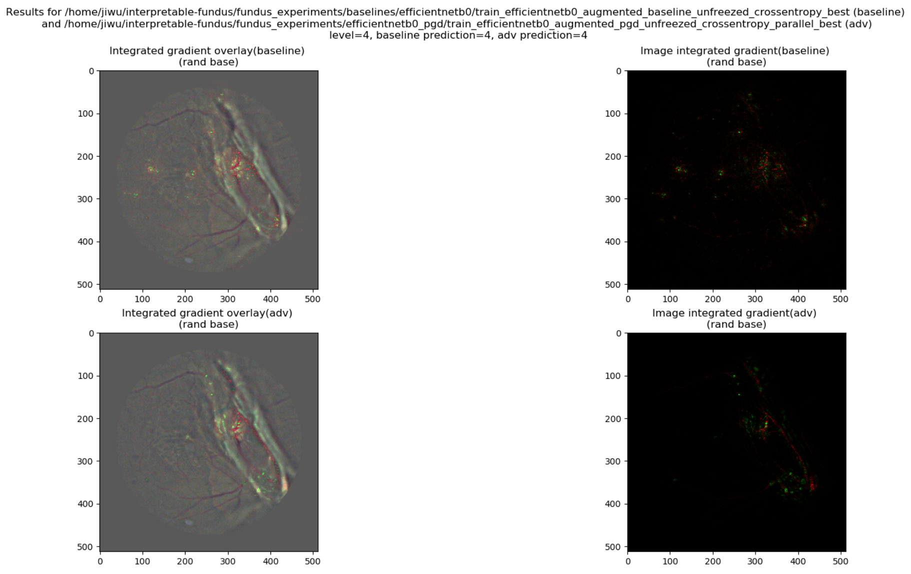

Results for /home/jiwu/interpretable-fundus/fundus_experiments/baselines/efficientnetb0/train_efficientnetb0_augmented_baseline_unfreezed_crossentropy_best (baseline)
and /home/jiwu/interpretable-fundus/fundus_experiments/efficientnetb0_pgd/train_efficientnetb0_augmented_pgd_unfreezed_crossentropy_parallel_best (adv)
level=4, baseline prediction=4, adv prediction=4
Integrated gradient overlay(baseline) (rand base)
Image integrated gradient(baseline) (rand base)
Integrated gradient overlay(adv) (rand base)
Image integrated gradient(adv) (rand base)

Figure 8: *Integrated gradients of PGD attack-trained model and baseline model evaluated on an image of level 4 (suffer from serious DR) of the original dataset.*

6 Conclusion

In this work, we investigate the possibility of enhancing model robustness and interpretability on a fundus dataset via different methods. Particularly, we confirm at least in our case that neural network can be rendered resistant to adversarial attacks by including adversarial samples into training and we provide evidence that adversarial training is possible to guide the model to pay more attention to meaningful features and thereby improve the interpretability of the model. In addition, we find that training on a corrupted dataset equips models with a certain degree of robustness against the same types of corruptions whilst it is not able to help models become more resistant to adversarial attacks nor become more interpretable.

Nonetheless, due to the lack of medical expertise, we are not able to identify genuine robust features which are used by professional clinicians when deciding levels of DR and therefore cannot tell if the model really learns them. Moreover, this work can still be improved and extended in many ways. As for adversarial training, one might use it as a data augmentation technique which could restore the problem of overfitting in an oversampled dataset. Besides PGD attack, other adversarial attacks could also be employed and tested against.

References

[Edi19] Nature Editorial. "Ascent of machine learning in medicine". In: *Nature Materials* 18.5 (May 2019), pp. 407–407. ISSN: 1476-1122, 1476-4660. DOI: `10.1038/s41563-019-0360-1`. URL: `http://www.nature.com/articles/s41563-019-0360-1` (visited on 10/27/2019).

[Sze+14] Christian Szegedy et al. "Intriguing properties of neural networks". In: *International Conference on Learning Representations*. 2014. URL: `http://arxiv.org/abs/1312.6199`.

[Big+17] Battista Biggio et al. "Evasion Attacks against Machine Learning at Test Time". In: *CoRR* abs/1708.06131 (2017). arXiv: `1708.06131`. URL: `http://arxiv.org/abs/1708.06131`.

[Fin+19] Samuel G. Finlayson et al. "Adversarial attacks on medical machine learning". In: *Science* 363.6433 (Mar. 22, 2019), pp. 1287–1289. ISSN: 0036-8075, 1095-9203. DOI: `10.1126/science.aaw4399`. URL: `https://science.sciencemag.org/content/363/6433/1287` (visited on 10/27/2019).

[KGB16] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. "Adversarial Machine Learning at Scale". In: *arXiv:1611.01236 [cs, stat]* (Nov. 3, 2016). arXiv: `1611.01236`. URL: `http://arxiv.org/abs/1611.01236` (visited on 09/29/2019).

[GSS15] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and Harnessing Adversarial Examples". In: *International Conference on Learning Representations*. 2015. URL: `http://arxiv.org/abs/1412.6572`.

[WJM17] Brendel Wieland, Rauber Jonas, and Bethge Matthias. "Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models". In: *arXiv:1712.04248 [cs, stat]* (Dec. 12, 2017). arXiv: `1712.04248`. URL: `http://arxiv.org/abs/1712.04248` (visited on 09/29/2019).

[Mad+17] Aleksander Madry et al. "Towards Deep Learning Models Resistant to Adversarial Attacks". In: *arXiv:1706.06083 [cs, stat]* (June 19, 2017). arXiv: `1706.06083`. URL: `http://arxiv.org/abs/1706.06083` (visited on 10/16/2019).

[Ily+19] Andrew Ilyas et al. "Adversarial Examples Are Not Bugs, They Are Features". In: *arXiv:1905.02175 [cs, stat]* (May 6, 2019). arXiv: `1905.02175`. URL: `http://arxiv.org/abs/1905.02175` (visited on 09/21/2019).

[He+15] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *arXiv:1512.03385 [cs]* (Dec. 10, 2015). arXiv: `1512.03385`. URL: `http://arxiv.org/abs/1512.03385` (visited on 01/08/2020).

[TL19] Mingxing Tan and Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: *arXiv:1905.11946 [cs, stat]* (Nov. 22, 2019). arXiv: `1905.11946`. URL: `http://arxiv.org/abs/1905.11946` (visited on 01/08/2020).

[Sze+15] Christian Szegedy et al. "Rethinking the Inception Architecture for Computer Vision". In: *arXiv:1512.00567 [cs]* (Dec. 1, 2015). arXiv: `1512.00567`. URL: `http://arxiv.org/abs/1512.00567` (visited on 09/29/2019).

[Lin+18] Tsung-Yi Lin et al. "Focal Loss for Dense Object Detection". In: *arXiv:1708.02002 [cs]* (Feb. 7, 2018). arXiv: `1708.02002`. URL: `http://arxiv.org/abs/1708.02002` (visited on 01/08/2020).

[Cui+19] Yin Cui et al. "Class-Balanced Loss Based on Effective Number of Samples". In: *arXiv:1901.05555 [cs]* (Jan. 16, 2019). arXiv: `1901.05555`. URL: `http://arxiv.org/abs/1901.05555` (visited on 01/08/2020).

[STY17] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. "Axiomatic Attribution for Deep Networks". In: *arXiv:1703.01365 [cs]* (Mar. 3, 2017). arXiv: `1703.01365`. URL: `http://arxiv.org/abs/1703.01365` (visited on 09/21/2019).