RobustNet : Adversarial Training and Data Augmentation Techniques against ImageNetC Corruptions.

Karthik Thimmavajjula Narasimha
Georgia Institute of Technology
North Avenue Atlanta, GA 30332
karthik6@gatech.edu

Saptarshi Basu
Georgia Institute of Technology
North Avenue Atlanta, GA 30332
sbasu7@gatech.edu

Abstract

RobustNet provides perturbation based adversarial training (fooling images) and data augmentation (style transfer, AugMix, random crop and horizontal flipping) based approaches to improve corruption robustness of a CNN (ResNet-18). A 20 class subset of ImageNet is used for adversarial training and data augmentations and performance is tested using Mean Corruption Error (mCE) on the same 20 class ImageNet-C dataset. Pre-trained ResNet-18 fine-tuned on deliberately misclassified perturbation based fooling images when combined with clean data performed best with mCE of 0.76. Similar improvements were also obtained with ResNet-18 trained from scratch indicating its applicability outside of pre-trained models. Comparison of data augmentation techniques indicated that AugMix performed best (mCE = 0.96) followed by stylized ImageNet method (mCE = 1.01). A combination model trained from scratch on Stylized Images, ImageNet20 with AugMix resulted in mCE of 0.91. The fooling method approach demonstrated significant corruption robustness at a low computational cost.

1. Introduction

Image classification through CNN has been widely studied and has several state-of-the-art model architectures. However, CNN performance is compromised in domain transfer and out of distribution data present in real life applications which might differ from curated training datasets. This is in marked contrast to human vision systems which are very robust to such input perturbations and are not fooled by minute variations. This robustness to corruption could be based on human's ability to learn global features compared to CNN's affinity towards texture based cues [5]. Nevertheless, as deep learning based vision system find application in self-driving cars, robots, and industrial applications, it is critical to ensure that such systems are robust to common corruptions like image noise, digital modifications, weather related issues, and blurred images. This paper looks at different data augmentation techniques (random crop and flipping, AugMix [9], stylized images) and proposes a perturbation based adversarial training to improve the robustness of CNN's to common corruptions by deliberately creating misclassified or fooling images.

A comprehensive summary of existing work done on image corruption robustness is provided in Mummadi et al. [11] and Hendrycks et al. [7]. Some of the successful approaches are stylized images [5], feature space augmentation like Style Blending ([10] and [12]), data augmentations (AugMix [9], CutMix [17], RandAugment [4]), self-supervised learning [8] to learn shape features, and network architecture updates including Info Dropout [13] and ViT based models [3]. Many of these approaches are effective but suffer from high computational cost, long time to train, and may not be generalized across different corruption types.

Developing methods to improve corruption robustness of CNN models (ResNet-18 and others) would significantly improve real life performance and reliability of deep learning based vision systems in safety critical applications like self-driving car and robotic surgery. It is important to develop training methods with low computational cost that are not based on test corruptions to develop a robust model that can provide similar accuracy in unknown compromised settings.

1.1. Data Generation and Processing

A brief description of different datasets used in this paper are provided below in Table 1. The 20 classes of ImageNet [2] selected for this study, ImageNet20 [1]), are from the animal kingdom: goldfish, great white shark, hammerhead, kite, spotted salamander, loggerhead, common iguana, horned viper, harvestman, hornbill, wallaby, jellyfish, sea anemone, conch, chambered nautilus, Dungeness crab, crayfish, spoonbill, crane, and bustard.

DateSet	Definition/Details
IN20 or ImageNet20	20 class Subset of ImageNet dataset [2];1300 training image and 50 val image per class; Varying resolution
IN20-SIN20	IN20 dataset augmented with Stylized ImageNet dataset [5]
ImageNetFool	IN20 dataset replaced with fooling images generated with different learning rates
IN20-Fooling5/IN20-Fooling20/IN20-Fooling100	IN20 dataset replaced with fooling images; 1300 image per class; Number after fooling denotes learning rate of 5,20,100
CropAndFlip	Training Procedure utilizing Random Cropping and Random Horizontal Flipping as data pre-processing
AugMix	Training Procedure utilizing AugMix [9] as data pre-processing in addition to Random Cropping and Random Horizontal Flipping
ImageNetC20/ ImageNet-C	Test DataSet which includes the same 20 classes as IN20 but with training images corrupted using one of the 19 corruptions with severity levels ranging from 1 to 5 [7]

Table 1. DataSet Definitions.

2. Approach

A multi-pronged approach was undertaken to improve the robustness of the selected neural network (ResNet18) against common corruptions found in the ImageNet-C dataset [7]. The approaches were built on implementing, optimizing, and combining methods outlined in the literature (data augmentations like random crop, horizontal flipping, AugMix [9], and stylized images (SIN) [5] as well as proposing a novel adversarial training approach built on the imperceptible perturbation based mis-classification method Szegedy et al.[16]. (henceforth referred to as fooling images).

The corruption robustness problem was simulated by using ImageNet-C data [7] as the test set which was not exposed to the neural network model (ResNet-18) during training. This is key to ensure that the robustness of the model can be tested independently. Care was also taken to avoid data augmentation techniques that directly mimic the corruptions seen in the ImageNet-C data [7]. The present work focuses on enabling the chosen neural network ResNet-18 learn more robust features through a combination of data augmentations (image transformations and stylized images) and adversarial training (statistical corruptions).

ResNet-18 was selected for the problem as it is the smallest of the state-of-the art Residual Networks providing an excellent option for using a well performing model that is manageable for training on the ImageNet20 (subset dataset of 20 classes) [1]. Making these modifications were necessary due to the large number of experiments and hyperparameter tuning involved. The computational resources needed for this was very high precluding training on full ImageNet dataset with bigger ResNet architectures. Initially we ran into issues with loading and working with the entire dataset. We overcame this issue by carefully selection smaller batch sizes and optimizing the learning rate accordingly. Through these experiences we learnt a great deal about how to work with larger datasets and the compromise between entire dataset into memory leading due to RAM limitations vs. working with smaller batch sizes at the cost of slower but easily manageable training with limited computational power.

An initial baseline performance was set by training a pretrained ResNet-18 model on ImageNet20 [1] data and updating the final fully connected layer. The output was modified from 1000 classes to 20 classes to match the number of classes in ImageNet20. The validation set was 50 different clean image corresponding to the same 20 classes. The same validation set was used for all subsequent experiments but the training set was improved by combining data augmentations and adversarial training using fooling images 1. With the augmented datasets, the model was either trained from scratch or all the weights were fine tuned based on pre-trained model for some of the experiments for quick experimentation. The data trend for both models trained from scratch and pre-trained models are very similar.

Further, we propose a novel adversarial training approach based on the fooling image concept introduced by Szegedy et al.[16]. The authors of that paper allude to the potential of using these mis-classified images for adversarial training based on the principle of hard-negative mining. In the present work, we use the misclassified images to force the ResNet-18 network to learn more robust features. We generate different levels of perturbations by modifying the learning rate (lr = 5, 20, 100) used in the gradient ascent algorithm. For each image, a random class was selected as the target class and the pre-Softmax score was maximized for that class using gradient ascent algorithm using the baseline ResNet-18 model with an updated fully connected layer. The code was built upon what was submitted in the assignments with additional experiments on learning rate and max iterations. It was also modified to handle batch processing and generation of images from the ImageNet20 dataset [1].

Initially, training on just the misclassified images did not result in significantly improved robustness against cor-

ruptions. However, when combined with the clean ImageNet20 dataset, the corruption robustness was significantly improved. A combination of clean and hard-negative images might be the key to make the neural network learn more robust features to address the dichotomy. It must be highlighted here that the perturbations were not necessarily imperceptible at higher learning rates. The changes are relatively more significant than what was used in the work by [16] since our focus is not on creating imperceptible changes but rather generating adversarial samples. The advantages of this approach are manifold - ranging from ease of generating these images and shorter training time compared to stylized images and other data augmentation methods as well as significantly improved performance.

We used the codes provided by Geirhos et al. [5]to generate Stylized ImageNet Data from the ImageNet20 dataset (Figure 12). The data was generated by transferring a randomly picked paintings's style from painters by number dataset on Kaggle [14] to ImageNet20 training dataset. This essentially randomizes the texture of each training image and forces the network to learn features beyond localized textures. We refer to this dataset as SIN20 (StylizedImageNet20).

For AugMix transformation we used available pytorch implementation of the AugMix based on Hendrycks et al.'s work [9]. AugMix stochastically mixes common image transformations in conjunction with a consistency loss to generate an augmented dataset. Mixing different augmentations stochastically prevents the neural network from memorizing augmentations thereby generalizing better to the corruptions encountered in the test set. To ensure that the neural network is not trained on the corruptions in the dataset, we removed brightness, contrast, color and sharpness from the list of transformations that can be applied to the training dataset.

3. Experiments and Results

The robustness of the techniques were evaluated by first training ResNet18 network from scratch on the various augmented datasets by converting the images to tensors, resizing and normalizing (using based on best practices ,see provided code for details) by employing pytorch DataSet, DataLoader and transform classes. We employed softmax crossentropy loss for training the networks since this is a classificatin problem . To avoid overfitting, weight decay or L2 regularization was used in the SGD and Adam optimizer. Learning rate of 0.005 was used for fine tuning pre-trained models, while hyperparameter tuning was used when learning from scratch to find optimal learning rate (0.001) and L2 regularization coefficient (0.001).

The performance is then evaluated on the ImageNetC20 dataset (refer Table 1) using mean corruption error (mCE) as the metric [7].

mCE is evaluated as follows. Let $E_{s,c}^{Network}$ denote the top-1 error rate of a 'Network' on a particular corruption type 'c' with severity 's'. Then corruption error for that particular corruption is defined as

$$CE_c^{Network} = \frac{\sum_1^5 E_{s,c}^{Network}}{\sum_1^5 E_{s,c}^{BaselineNetwork}} \quad (1)$$

where $E_{s,c}^{BaselineNetwork}$ denotes the top-1 error rate on the same corruption 'c' with severity level 's' using a reference network (Baseline Network). In our work we use ResNet-18 network pretrained on 1000 class ImageNet dataset and the fully connected layer's weights finetuned on our ImageNet20 (IN20) dataset as the reference network.

The normalization is done to take into account the variation in difficulty associated with different corruptions as evident in the baseline performance shown in Fig. 1

The mean Corruption Error (mCE) is then computed as the mean of the corruption errors over all the different types of corruptions.

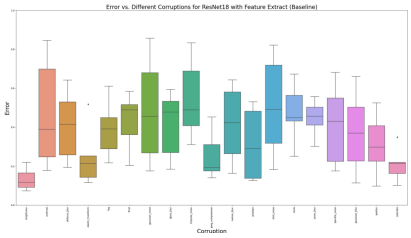

Figure 1. Raw error rate across different corruption types for all severity levels in baseline model

.

The mCE for the various techniques is shown in Figure 3 and are discussed in more detail below.

3.1. Data Augmentation

3.1.1 Effect of Random Crop and Horizontal Flip Transformations

ResNets trained on ImageNet data are typically trained [6] by adding Random Crop (where a random portion of the image of desired size is cropped from the training image) and Random Horizontal Flip (where image is randomly flipped horizontally). These operations are known to make the networks robust and generalize better to test datasets. As a sanity check for our code, we evaluated the mCE on ImageNetC20 dataset with and without these Crop and Flip Transformations. As shown in Figure 3, the mCE improves from 1.3 to 1.05 by adding the Crop and Flip Transormations serving as a sanity check for our code.

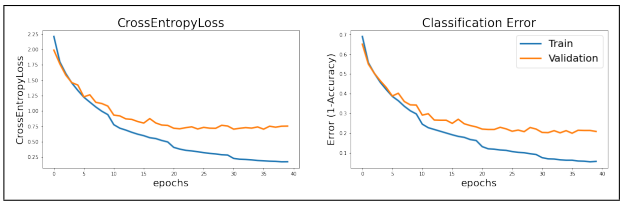

Figure 2. Typical Learning curves (Cross Entropy Loss and Classification Error vs. number of epochs) showing the training process

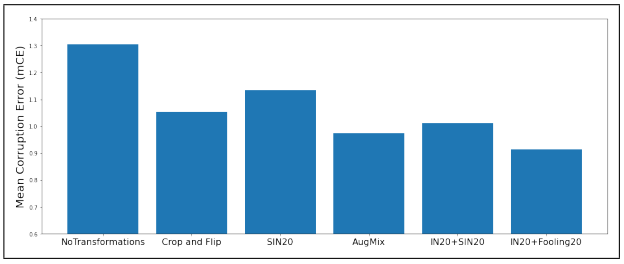

Figure 3. Mean Corruption Error (mCE) vs. different data transformation techniques explored in this work.

3.1.2 Effect of AugMix, Stylization and Adversarial training (Fooling Image)

The ResNet-18 networks were then trained on AugMix, SIN20 and Fooling20 datasets.The results of these training are summarized in Figure 3. The mCE reduced from 1.05 with CropAndFlip to 0.98 with AugMix, 1.01 with IN20+SIN20 and 0.91 with IN20+Fooling20 datasets. This represents a best case improvement of 14 percent compared to typically used CropAndFlip data preprocessing and 30 percent improvement compared to no transformations in data preprocessing.

Figure 4 shows the histograms of weights and biases of initial and final layers of the networks (layer1, layer4 and fully connected layers) trained with these techniques. Interestingly the distribution of network weights is very similar for initial layers and also for all the convolutional layers across the entire network. The main difference comes from the batch normalization and down sampling layers close to the fully connected layer. This suggests that the different data augmentation techniques are learning similar features in the initial layers. However, the compositions of how these features are used in layers closer to the output are different contributing to different robustness in the network. A complete understanding of what is contributing to this difference requires a much thorough examination of various layers and is beyond the scope of the current work.

To further analyze the similarities between the different layers of these networks, we computed the CKA (Centered Kernel Alignment) scores using torch_cka package in python [15]. The CKA plots for AugMix, IN20+SIN20, IN20+Fooling20 vs. CropAndFlip are summarized in Figure 5. These plots show that the different networks behave differently with respect to their similarity to the widely used CropAndFlip transformations. This suggests that the different techniques are improving the robustness of the network through different mechanisms and could perhaps be combined to further improve the robustness of the network.

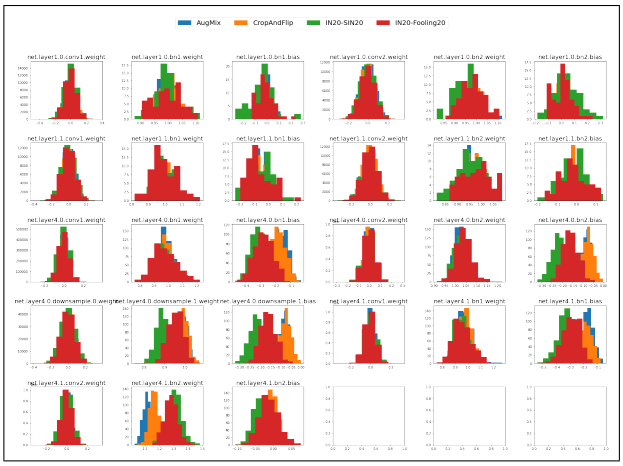

Figure 4. Histograms of Weights and Biases of first and last layers showing that the different between the networks trained using various techniques is probably arising from latter layers and is mostly confined to the batch normalization and downsampling layers

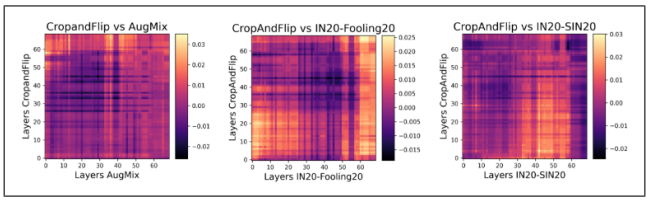

Figure 5. Centered Kernel Alignment Analysis comparing AugMix, IN20-Fooling20 and IN20-SIN20 trained neural networks to CropAndFlip trained neural networks

This is further evidenced from our result on training the network using IN20+SIN20 dataset by applying AugMix transformation. The mCE in this case improves from 1.01 for IN20+SIN20 and 0.98 for AugMix to 0.91 3. This presents us with myriad of opportunities to further investigate combining these various methods to further improve the robustness of the networks.

3.2. Further Investigations on Hard-Negative Adversarial Training

In this section we further investigate the effect of adding Fooling Images during training. An example of the misclassified images created with different learning rates are shown in Fig. 6 where the original (horned viper) image is misclassified as bustard (LR=5.0), spotted salamander (LR=20.0), and common iguana (LR = 100.0)). It is helpful to know

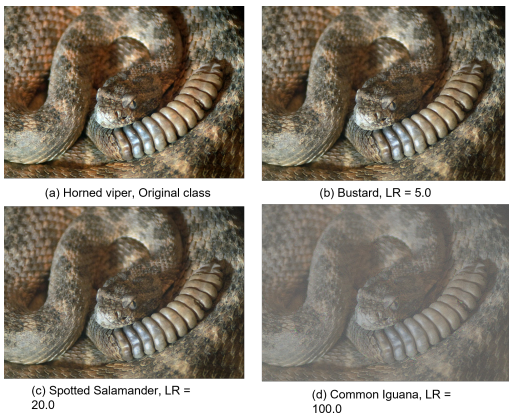

Figure 6. Example misclassified images (left to right): a) Original, b) LR = 5, c) LR = 20, and d) LR = 100

that although at very high learning rate of 100.0, the dpi reduction employed to ensure faster Fooling Image generation causes haziness in the images, this does not expose the network to corruptions in the test dataset since haziness is not a ImageNet-C corruption [7].

A number of combinations of the hard-negative adversarial training approach was utilized in this work. As a baseline, their performance on clean validation dataset and ImageNetC is shown in Fig. 7. A few acronyms are introduced in this image to define the different experiments: FT - FineTuned, O-Original, fc - fully connected, F-num - Fooling with learning rate given by the number, All - All fooling images with different learning rates (LR), C - Combined with clean ImageNet20 data. AugMix did not necessarily help in improving clean performance or robustness when the model weights are fine tuned from pre-trained model. Also, interesting to note that just fine tuning on adversarial examples had a poor performance. However, when combined with clean ImageNet20 dataset, significant improvement (24%) was observed in corruption robustness. The best MCE of 0.76 was obtained when all the fooling images were combined with clean data set. It is also interesting to observe that at lower fooling learning rate, the clean validation error dropped while at very high learning rate of 100.0, the clean error increased relative to baseline. In all cases, when fooling images are combined with clean ImageNet20 dataset, mCE dropped but the performance level is clearly influenced by the learning rate. From a limited set of experiments, the optimal is somewhere around 20.0 but more iterations would be needed to tune this parameter. Smaller learning rate resulted in imperceptible changes which helped in improving clean data performance as well as MCE. But increasing the learning rate to 20.0 resulted in higher corruption robustness or lower MCE. Further increasing the learning rate to 100.0, resulted in deteriorating performance. The level of perturbations seem to be key in ensuring that the neural network learn robust features as it tries to distinguish between a clean image and a similar image that it misclassifies due to optimized low probability high dimensional changes in the manifold Szegedy et al.[16].

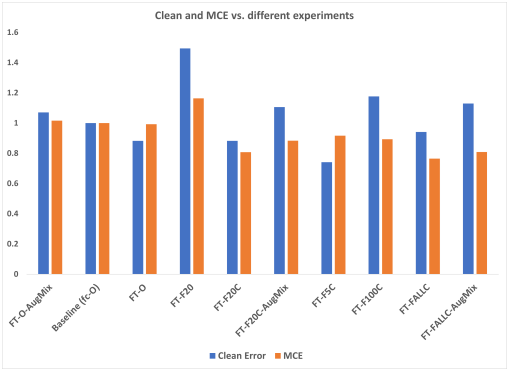

Figure 7. Clean validation error and mCE for different training experiments.

The model trained on all three sets of peturbed misclassified images (fooling images) along with clean ImageNet20 performed best. This is evident in Fig. 8 across different corruption types. Comparison with Fig. 1, shows that the adversarial-trained model lowers the overall error while also reducing the variation in error rate between different severity level. This demonstrates that the model is robust against more severe levels of corruption as well.

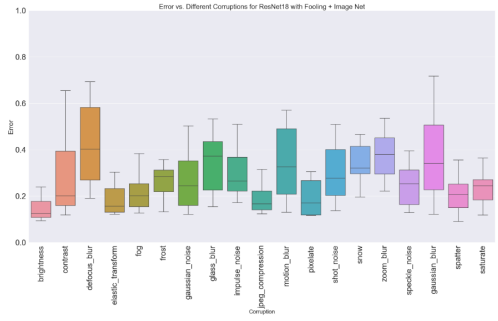

Figure 8. Raw error rate across different corruption types for best model

.

This is further substantiated by a more granular analysis of the impact of adversarially perturbed training on ResNet-18 performance as shown in Fig. 9. The plots show the error of the best model normalized with the error of baseline model across the five broad classes of ImageNetC corruptions - noise, blur, weather, digital, and extra [7]. This relative nuances are important but has to be interpreted in the context of mCE of 0.76 with the best model: ResNet-18

trained on all sets of ImageNetFool (5,20,and 100) combined with clean ImageNet20 data. Interestingly, the performance is poorer for brightness corruption at lower severity levels, which is generally one of the easier corruptions based on Fig. 1. It could perhaps be due to the lowering of dpi in the perturbed images. In case of digital corruptions, the improvement is more at higher severity levels than at lower severity levels.

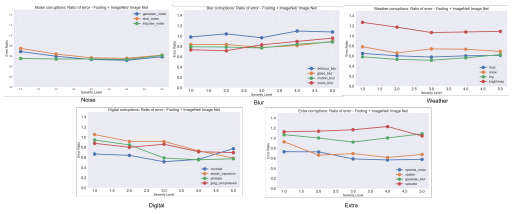

Figure 9. Error rate ratio across different corruption types

.

3.2.1 Adversarial Training: ResNet-18 Filter and Feature Map Analysis

To better understand the differences in training, a brief qualitative analysis of the ResNet-18 kernel weights and feature maps for few selected layers are presented here. Additional high resolution images are available in the supplementary folder. The weights in the initial convolution layer (layer 1) shows not significant difference in the kernel weights. However, at deeper layers (layer 4 Block 2 Conv2), subtle variations in weights in different channels are visible indicating that the model is focusing on different features at deeper levels. The initial layers might be focusing on local features while the abstract global features are learnt differently. There is a dampening effect of pre-training on exactly how these weights might differ. This is illustrated in Fig. 10.

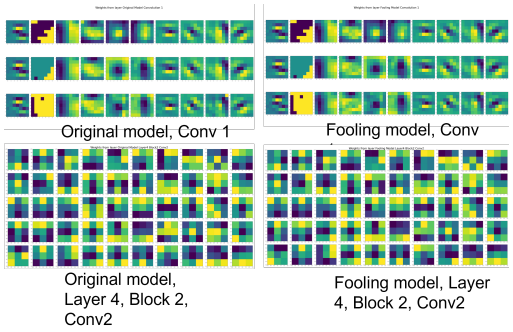

Figure 10. Kernel/Filter weights variability across original and fooling image trained models.

The initial layers of the neural network learn more local features while the deeper layers learn more abstract global features. The differences in feature maps in models trained using clean ImageNet data and those trained on fooling images (learning rate = 20.0) are shown in Fig. 11. The image corresponds to bustard bird with motion blur corruption of severity level 3. 64 equally spaced channels were selected for comparison as the number of channels increased at deeper layers. It is observed that at both initial and deeper convolution layers, there are differences in some of the channels between the models trained with clean ImageNet20 and ImageNetFooling datasets, respectively. In general, qualitatively, the bustard like features are visible in the feature maps of the model trained with fooling images.

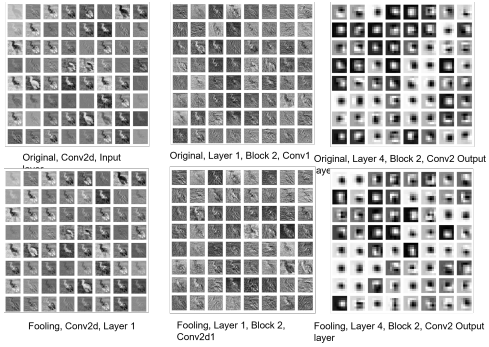

Figure 11. Kernel/Feature maps visualization across original and fooling image trained models.

4. Summary and Conclusions

In the present work, ResNet-18 was used as a representative CNN to test a number of data augmentation and adversarial training techniques to improve the robustness of the network against ImageNet-C corruptions. All the tested data augmentation techniques including AugMix, SIN20 and adversarial training using Fooling20 considerably improved the robustness of the network. Our custom approach for adversarial training using a perturbation based technique [16] performed the best with an mCE of 0.76. This improved performance was achieved at low computation cost compared to existing data augmentation or stylized image based approaches. The fooling image approach along with AugMix, SIN, and other data augmentations were implemented from scratch to demonstrate application beyond pre-trained models. In future, this work can be extended across a wider data set with more perturbed images and novel architecture developments like Information dropout and transformers to develop more robust neural networks that can generalize across domain and out of distribution datasets, typically observed in real life applications.

5. Work Division

The contribution of the different team members are summarized in Table 2.

There was excellent team work and collaboration between the two team members with equal contribution across all areas of the project. Even though the code implementation of specific parts were undertaken at an individual capacity, there were strong inputs and ideas shared on every aspect of the project including bug review, testing, and sharing of data.

References

[1] https://www.kaggle.com/datasets/shahnazari/imagenet20. 1, 2

[2] Eunbyung Park Addison Howard. Imagenet object localization challenge, 2018. 1, 2

[3] Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, and Andreas Veit. Understanding robustness of transformers for image classification, 2021. 1

[4] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space, 2019. 1

[5] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019. 1, 2, 3, 8

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 3

[7] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019. 1, 2, 3, 5

[8] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty, 2019. 1

[9] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty, 2019. 1, 2, 3

[10] Boyi Li, Felix Wu, Ser-Nam Lim, Serge Belongie, and Kilian Q. Weinberger. On feature normalization and data augmentation, 2020. 1

[11] Chaithanya Kumar Mummadi, Ranjitha Subramaniam, Robin Hutmacher, Julien Vitay, Volker Fischer, and Jan Hendrik Metzen. Does enhanced shape bias improve neural network robustness to common corruptions?, 2021. 1

[12] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap by reducing style bias, 2019. 1

[13] Baifeng Shi, Dinghuai Zhang, Qi Dai, Zhanxing Zhu, Yadong Mu, and Jingdong Wang. Informative dropout for robust representation learning: A shape-bias perspective, 2020. 1

[14] Wendy Kan small yellow duck. Painter by numbers, 2016. 3

[15] Anand Subramanian. torch_cka, 2021. 4

[16] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2013. 2, 3, 5, 6

[17] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features, 2019. 1

Student Name	Contributed Aspects	Details
Karthik Thimmavajjula Narasimha	Idea generation	Literature review and proposed ideas related to training and augmentations
	Data generation and processing	Generated SIN20 dataset
	Data Aug., Fooling training	Hyperparameter tuning and training/performance evaluation with AugMix, CropAndFlip, IN20, SIN20, IN20+SIN20 datasets
	Other contributions	Worked with training Fooling20, IN20+Fooling20 datasets from scratch
Saptarshi Basu	Idea generation	Literature review and proposed ideas related to training and augmentations
	Data generation and processing	Downloaded, cleaned, and processed ImageNetC and ImageNetFool datasets
	Fooling Image Experiments	Hyperparameter tuning and training/performance evaluation with fooling and combined datasets
	Other contributions	Worked with AugMix and L2 regularization to investigate overfitting

Table 2. Contributions of team members.

6. Appendix

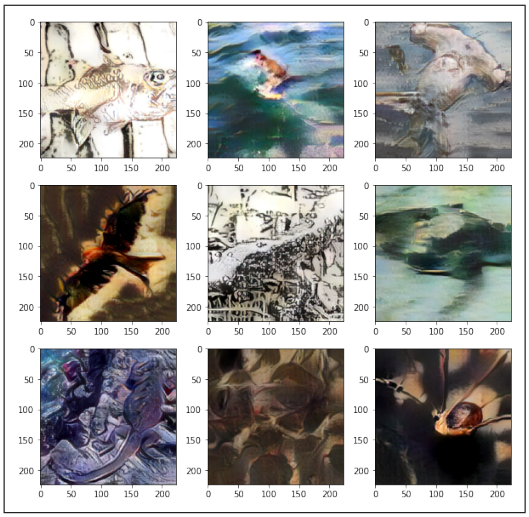

Figure 12. Typical Stylized Images used in the training set generated using the codes provided by [5]