On Improving the Robustness of Image Classification Model Using Fuzzing-Based Data Augmentation

Yuseung Lee, William Gyuho Suh, Guntitat Sawadwuthikul, Mehmet Hamza Erol, Baha Eren Yaldiz

***Abstract*—Deep neural networks (DNN) are to be notoriously brittle to small perturbations in their input data. These perturbations do not change the labeling of the dataset and are natural variations of the data points, so it is important to increase the robustness of the model. Recent studies showed that we can increase the robustness of DNN models by applying natural variations to the dataset. The choice of the data augmentation method is critical in the sense that the DNN learns from the selected dataset robustly. SENSEI [1] utilizes the genetic algorithm to generate the most suitable variant of the input data. We replicate it with their original approach and evaluated the performance of SENSEI using a common benchmark dataset: CIFAR-10 in terms of robust accuracy which is the accuracy of image classification when perturbed images are given input. In addition, we propose various genetic algorithms with three different selection methods and test their performances. In this paper, we conduct several analyses and experiments by implementing our SENSEI which is constructed from our own abstract framework AUGMENTATION. The results show that our SENSEI outperforms the standard model by achieving 83.0% robust accuracy which exceed 81.5% robust accuracy in the original paper.**

***Index Terms*—genetic algorithm, fuzzing, image classification, neural network, robustness, data augmentation.**

I. INTRODUCTION

WITH THE increasing demand for image classification techniques in cutting-edge technologies (e.g., autonomous driving [2], face detection [3]), there has been a noticeable level of improvement in the application of deep neural network (DNN) models in these fields [4]. Although DNN models have shown success in many image-related tasks, recent studies have brought up concerns about their effectiveness and robustness which can be very challenging even a fine amount of perturbation is added to images [5]. However, there are studies suggesting new techniques as well as toolboxes to help increase the robustness of machine learning models [5], [6], [7], [8].

Among the studies, we are interested in the various approaches for enhancing the robustness of DNN models in image classification and decided to work on the above paper [1]. The proposed technique applies the idea of Genetic Algorithm into data augmentation in order to prevent the model from being fragile to small perturbations in images. We have re-implemented the proposed data augmentation model (SENSEI) from the original paper by generalizing it into an abstract end-to-end model training framework and reproduced their experiments on the same image dataset, specifically CIFAR-10. In addition, we introduce two extensions of SENSEI and analyze possible improvements in terms of robustness and efficiency.

The paper contributes as a project assignment of CS454 AI Based Software Engineering, Fall 2021.

All authors have equal contribution.

II. ORIGINAL APPROACH

DNNs have demonstrated impressive success in a wide range of applications. However, they have also been shown to be quite brittle (i.e., not robust) to small changes in their input. The original paper [1] proposes a new augmentation framework, in particular a search-based selective data augmentation strategy, that aims to solve multiple performance-related problems of neural network models. These problems include: standard generalization, overfitting problems, robust generalization, and guaranteeing a model's persistence at perturbed samples. To improve the generalization of DNNs, the suggested idea in this paper is performing a basic form of data augmentation where, during training in each epoch, each training data is replaced by a variant created by randomly applying some sources of natural variations or noise (*optimal augmentation*). And at the same time, it prioritizes some challenging data points while ignoring the rest (*selective augmentation*). This framework, SENSEI, applies the two ideas aforementioned and formulates an objective function that needs to be optimized.

$$\min_{\theta} E_{(x,y)\sim D}\left[\max_{\delta\in S} L(\theta, x+\delta, y)\right] \tag{1}$$

The objective function aims to minimize the training loss — when the perturbed sample with the maximum loss is given. This way the inner maximization mimics a natural variation (e.g. rotated or translated media), that is most challenging for the model. And by the outer minimization, the model is trained to have a better prediction value. Interestingly, SENSEI emphasizes the inner maximization and implements it as a framework instead of tailoring the loss function. Instead of augmenting the data initially and training all data points together, SENSEI augments the data points in an iterative manner and trains on those until all data points become *pointwise robust*, meaning that the model can potentially predict any perturbation of a data point.

They have chosen the genetic algorithm as their method for data augmentation. Each series of transformations is represented as a *chromosome* that is used for genetic mutation and

crossover. Note once again that they only augment the samples that are not yet robust enough for the model, then apply the GA and finally choose the variation with the maximum loss. An example of a chromosome that enters the genetic algorithm in SENSEI would look as the following: $c = \{rotation : 1, translation : 2, shear : 0.15\}$. SENSEI applies its mutation and crossover operations onto such chromosomes that form in the current population in order to generate a new population. Then it selects the sample that has reduced the loss of DNN by the largest ratio in the generated population. The original paper learns the DNN models using SENSEI's method and measures the training loss and the robust accuracy.

To evaluate the performance of SENSEI, they used common benchmark datasets such as CIFAR-10, GTSRB, IMDB, and others, that are famous for image classification. Also, they used the state-of-the-art neural architectures to train specifically for each dataset, including ResNet [9] and Wide ResNet [10]. Six image transformation operations like rotation and translation are chosen and the hyperparameter space is restricted in order to make sure that the translated images are visually similar to naturally-perturbed ones. Despite this effort, their experimental setup was arguably biased since they were using a discrete search space.

III. AUGMENTATION: THE GENETIC DEEP LEARNING FRAMEWORK

For the implementation of our work, we first checked the existing libraries to add our extensions on top. However, the only existing code which is SENSEI's official implementation has a highly complex code structure that limits the extendability and readability of the code. On top of all those, it lacks the fundamental abstractions needed in applying the idea of the SENSEI that works on different deep learning models, datasets and augmentations.

Due to the reasons above, we decide to create a PyTorch [11] based framework, named AUGMENTATION[1], that addresses the limitations of SENSEI's official implementation. In our implementation, we support the easy integration of deep learning models, genetic algorithm models and custom datasets for training with genetic algorithm based data augmentation.

A. Code Structure

Our primary need for the code is to have a clear separation between the deep learning model, the genetic algorithm model, the dataset, and the augmentations to be applied. Therefore we decide to have the code structure as in Fig. 1.

1) main.py: The main script that parses the given config file and runs the experiment described on the config.

2) configs: The configurations of the experiment is described in the config file. The following are included in the config file:

- Experiment name
- Dataset to be used in the experiment
- Deep learning model and its parameters to be used in the experiment

[1]Code repository: https://github.com/beyaldiz/augmentation

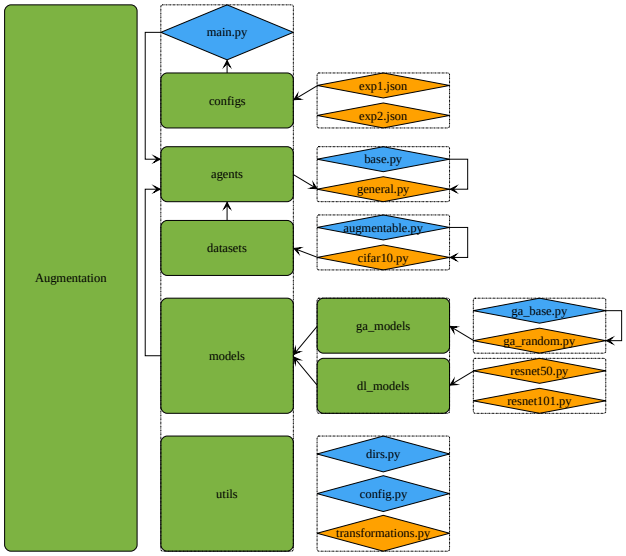


Fig. 1. Code directory. Folder levels are from left to right. Arrows represent dataflow of the code. Green corresponds to folders, blue corresponds to fixed source codes and orange corresponds to extendable source codes and configuration files. Arrows for the utils are omitted for clarity.

- Genetic algorithm model and its parameters to be used in the experiment
- Augmentations to be applied and their limits

3) agents: Agents handle the initialization of the models and datasets described in the config file. Also, they handle the training and logging related functions. All the custom agents should inherit base agent defined in "base.py".

4) datasets: Datasets are defined under this folder. All the dataset classes should inherit the "Augmentable" dataset class defined in "augmentable.py" to properly apply the augmentations created by the genetic algorithm models.

5) ga_models: Genetic algorithm models are defined under this folder. The custom genetic algorithm models should override the functions defined in "ga_base.py".

6) dl_models: Deep learning models are defined under this folder.

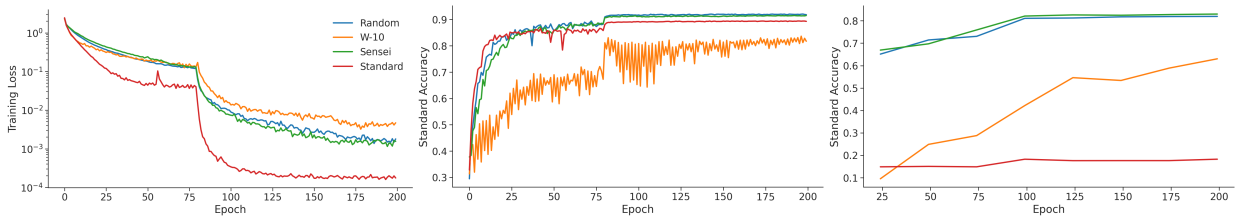
7) utils/transformations.py: All the augmentations are defined under this script as functions that takes the image as input.

B. Benefits of our Framework

The two key motivations of our implementation are extendibility and simplicity. Therefore, one can easily add a custom deep learning model and a genetic algorithm model to leverage the genetic algorithm guided augmentations on training. On top of that, one can easily define custom augmentations and simply apply them by adding it to the config file. Also, our implementation enables conducting many experiments efficiently by only creating multiple config files for each experiment.

IV. EXTENSION

According to the paper, SENSEI applies the elitism selection method for its GA implementation, as it states *"only best set*



(a) Training Loss (log) (b) Standard Accuracy (c) Robust Accuracy

Fig. 2. Result reproduction of [1]. We verify the implementation of our framework by observing the performance of ResNet-50 model [9] involving three different data-augmenting strategies: RANDOM (blue), W-10 (orange), and SENSEI (green). A plain model training without apply data augmentation (red) represents the baseline for all other strategies. RANDOM and SENSEI outperform other strategies and are equally efficient. All strategies are constructed from our framework AUGMENTATION which provides convenience for experimenting different genetic algorithms.

is passed as a current population for the next generation". In our work we introduce two spin-offs of SENSEI that utilize different GA selection methods. The first extension is SENSEI-RW, which applies the roulette-wheel selection instead of elitism. The second extension, SENSEI-T, applies the tournament selection by randomly choosing 5 chromosomes for each tournament and choosing the best chromosome among those five. When we re-implement the experiments from the original paper, we add these two extensions in order to understand whether different selection methods could bring an improvement to the original SENSEI in terms of robustness and effectiveness.

V. EXPERIMENTS

Using the newly implemented code for SENSEI, we conducted three experiments in order to see whether the performance of our re-implementation follows the trends shown in the original paper. In *Experiment 1* we verify whether the reproduction of SENSEI is successful by comparing its training loss and robust accuracy to that of the state-of-the-art augmentation methods. We set the hyperparameters of SENSEI as equal to that in the original paper's experiment. Then in *Experiment 2* we conduct the same experiment as *Experiment 1* but with a DL model with different parameters, in order to see if SENSEI would maintain its robustness when applying different neural network models. Finally, *Experiment 3* observes the performance of the extensions of SENSEI that we have derived by making modifications in two aspects of genetic algorithm. We first execute SENSEI-RW and SENSEI-T, which are derived by implementing new selection methods, and we also run SENSEI in two different types of search spaces: discrete and continuous.

A discrete search space, which is applied in the original study, divide the restricted range of transformation into equally-spaced values. For example, they restricted the range of rotation from $-30°$ to $30°$. The discrete search space used for the data augmentation considers only integral values in the range which consists of 61 possible values. In contrast, a continuous search space takes any value in the given range into consideration. This implies that the continuous search space offers a greater variety of choices than the discrete search space. However, to be fair to both settings, we measure the robust accuracy using the same set of perturbations that was used in the original paper. Only the the lower bound, the middle value, and the upper bound of the given range (e.g. $\{-30°, 0°, 30°\}$ for rotation) are used to test the model's robustness.

The data augmentation approaches used in our experiments include: RANDOM, W-10, SENSEI, SENSEI-RW, SENSEI-T and a *Standard* version with no augmentation involved is also used for comparing the robust accuracy. We choose CIFAR-10 as the target image dataset and train the selected DL model, ResNet-50, for 200 epochs. In this study, we limit the types of image transformations to only three, including rotation, translation, and shear. We run the following PyTorch-based experiments on NVIDIA GeForce RTX 3090 for approximately 4 days in total.

A. Experiment 1: Reproduction of SENSEI

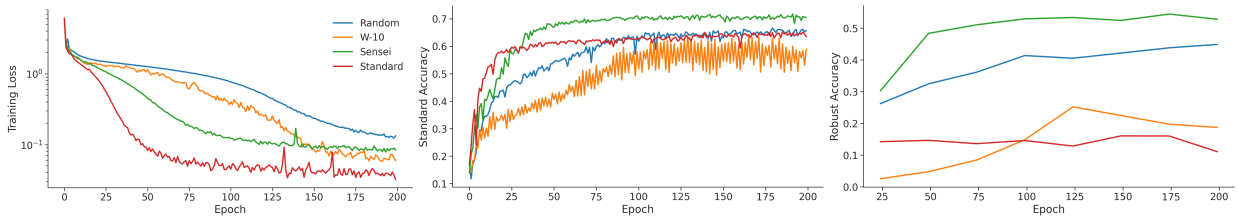
In our first experiment, we test our implementation of *Augmentation* by reproducing the data augmentation strategy SENSEI proposed by the original paper [1]. The results from Fig. 2(a) show that we successfullly obtain a similar trend of the training loss as the authors of [1] (Figure 4 in the original paper). Even so, the training loss of W-10 does not correlate with the original results as it outranks other strategies including SENSEI.

Our ResNet-50 model [9] achieves 83% using SENSEI and 82% using RANDOM augmentation strategy for the robust accuracy, which both results are relatively better than the original performance. Employing data augmentation technique with genetic algorithm applied obviously outperforms W-10 and standard training in terms of both standard and robust accuracy. On the contrary to the original paper, W-10 performs obviously worse than RANDOM in both the standard and robust accuracy. A summary of the training and the performance is showed in the Table I.

However, due to heavy computation of the genetic algorithm, SENSEI spent more than 28 hours to complete 200 epochs of training while RANDOM used only ~ 9 hours. Different from the original results, the performance gap between the two strategies is relatively less significant compared to the original results. Deploying SENSEI in practice might not be the most effective solution because of the time consumption.

B. Experiment 2: Robustness of the Framework

Despite the research question of the original paper that focused on the robustness of the deep neural network against



(a) Training Loss (log) (b) Standard Accuracy (c) Robust Accuracy

Fig. 3. Repetition of Experiment 1 with another implementation of ResNet-50. We utilize the API from an open-source library, torchvision, to instantiate any neural network model, including ResNet-50, that is provided in the model zoo of the library. The relative performance trend between each strategy is considerably similar to that of Experiment 1 except the training loss of W-10.

perturbed samples, we expect the same trend when we conduct the experiment with another neural network. We re-run Experiment 1 using ResNet-50 model which is provided by PyTorch [11] instead of the model implemented by the authors of [9].

Interestingly, the training losses are slightly different from the curves in Experiment 1. Illustrated in Fig. 3(a), RANDOM induces the most training loss compared to other data-augmenting strategies. This will be further discussed in the later section.

In terms of robust accuracy, there is no empirical change in the relative trend. SENSEI (53%) still performs slightly better than RANDOM (45%) and outputs a significantly greater accuracy than W-10 (19%). Standard training which does not involve data augmentation remains at the bottom with 11% robust accuracy. The standard accuracy in Experiment 2 correlates well with the results from Experiment 1 as shown in Fig. 3(b).

C. Experiment 3: Extension to Genetic Algorithm

In addition to the first two experiments, we extend the study by specifically adjusting the genetic algorithm. As mentioned in Section IV, we are interested in the effect of the selection method and propose two variants of SENSEI: SENSEI-T and SENSEI-RW. Moreover, we notice that the original implementation employed a discretized search space. We also conduct another set of executions to test the performance of SENSEIs in a continuous search space and compare them with the results from previous experiments.

1) Selection Method: We compare the performance of SENSEI and two new variants using PyTorch's ResNet-50 as the target model because of the less training time. Fig. 4 reveals that SENSEI-RW outperforms SENSEI in terms of standard and robust accuracy although the improvement is not groundbreaking (see the figures in Table I. SENSEI-T performs slightly worse than the first two strategies but still outperforms W-10 and standard training substantially.

2) Search Space Formulation: Another modification on the genetic algorithm by enlarging the search space does not show any remarkable improvement. As shown in Fig. 5, the relative performance trend between each strategy is also similar to that of the discretized search space. However, we notice from Table I that there is $\sim$ 1% increase in the robust accuracy on all SENSEI variants (54%, 54%, and 52% for SENSEI, SENSEI-RW, and SENSEI-T, respectively).

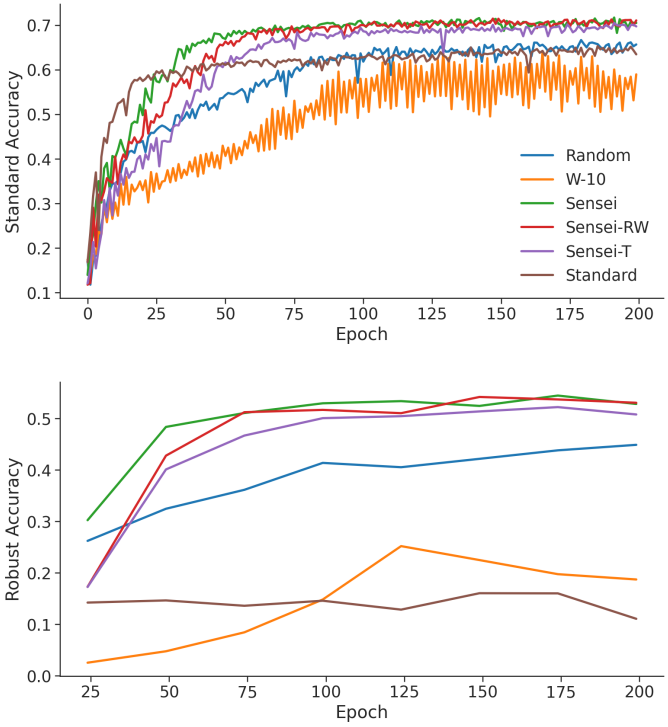


Fig. 4. Performance comparison using different data-augmenting strategies including two of our proposed variants: SENSEI-RW and SENSEI-T which modify the selection method of the genetic algorithm. The network used for the training is ResNet-50 that is provided by torchvision (the same setting as Experiment 2) which requires less training time.

VI. DISCUSSION

A. Reproduction Comparison

We have successfully implemented SENSEI using our proposed framework AUGMENTATION which is more modular and readable. We test its validity by replicating an experiment conducted in the original paper. In our Experiment 1, we train the original ResNet-50 model to classify CIFAR-10 dataset. The results mostly align with the paper well except two key differences: the performance gap between RANDOM and SENSEI and the abnormality of W-10.

According to Fig. 2(c), the performance gap between the two strategies is less significantly compared to the same gap in the original paper, decreasing from 8% to 1%. From this result, it can be seen that learning with RANDOM has a large

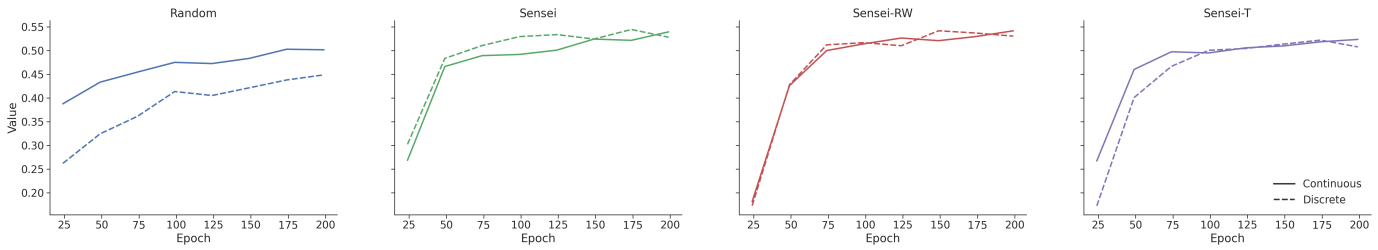


Fig. 5. The effect of search space. We compare the performance of the output model in terms of robust accuracy after training it with different configuration of the genetic algorithm. A continuous search space on the parameters (solid line) performs slightly better than a discretized search space (dashed line). ResNet-50 model provided by torchvision is used for the training.

TABLE I
FULL PERFORMANCE COMPARISON OF STANDARD ACCURACY, ROBUST ACCURACY, AND TRAINING TIME (IN HOURS) ACROSS DIFFERENT STRATEGIES AND MODEL IMPLEMENTATIONS

Model	Search Space	RANDOM			W-10			SENSEI			SENSEI-RW			SENSEI-T			Standard		
		SAcc	RAcc	Time	SAcc	RAcc	Time	SAcc	RAcc	Time	SAcc	RAcc	Time	SAcc	RAcc	Time	SAcc	RAcc	Time
ResNet-50 (original)	Discretized	**0.918**	0.819	9:41	0.817	0.630	25:13	0.915	**0.830**	**28:56**	-	-	-	-	-	-	0.893	0.182	6:23
ResNet-50 (torchvision)	Discretized	0.656	0.449	3:30	0.589	0.187	14:41	0.705	0.528	14:42	**0.710**	**0.530**	**15:17**	0.697	0.508	14:37	0.634	0.111	1:13
ResNet-50 (torchvision)	Continuous	0.700	0.502	3:36	0.645	0.322	14:28	0.703	0.539	14:48	**0.712**	**0.541**	**15:11**	0.701	0.523	14:57	0.664	0.173	1:15

deviation in the results. The results also vary depending on the hyperparameter of the initial experimental setup.

The behaviour of W-10 does not appear only in Experiment 1 but in other experiments as well. One may notice that it induces significant training loss compared to other strategies but it fails to improve both its standard and robust accuracy. Especially for the standard accuracy, W-10 reveals an abundance of oscillation in the curve and performs even worse than standard training that does not involve any data augmentation. One possible explanation is that W-10 attempts too hard inducing samples with high loss so that at the same time it slows the learning. However, another experiment should be conducted to prove this claim.

On the contrary, W-10 succeeds in improving the robust accuracy as it performs definitely better than the standard training. By introducing a great amount of training loss might lead towards the model's robustness, however, the perturbation should be selected carefully so that it aids the learning of the model instead of always choosing the one with the highest loss. One possible alternative is to apply the strategy of W-10 right before that batch is feed-forwarded to the model instead of choosing the perturbation of all batches and samples before each epoch starts.

B. Robustness of SENSEI

Apart from the model's robustness, the robustness of SENSEI seems to be questionable. According to the results in Table I and those from the original paper, there are inconsistency when the model is changed. For example, the performance of SENSEI drops substantially when we change only the implemenation of ResNet-50 from the original version to the one provided by PyTorch. We have not yet thoroughly looked into both implementation and identified the differences, still, the disparity of the performance between each implementation suggests that SENSEI needs to be fine-tuned when the model is changed. Interestingly, Gao has mentioned this *threat to validity* in Section 5.6 of [1]. They admitted that SENSEI may not be generalized for the other types of models (external validity). However, they have made a proper study on the effect of the parameters of the genetic algorithm, namely internal validity, such as altering the population size and the fitness function in Section 5. Their results guarantee the efficiency of SENSEI when the population size is at least 10. Some parameters might be dependent on the dataset as well as the model selection. Another set of experiment to validate SENSEI's robustness can be done on a vast set of model choices, as well as to find a proper solution or methodology to fine-tune this very technique.

C. Search Space Formulation

Despite the fact that using a continuous search space, which is larger, performs slightly better than using a discrete search space in terms of robust accuracy, our results from Table I show that the performance gap between SENSEIs and RANDOM diminishes when the search space expands. Although we have not conducted an additional experiment that focuses on the effect of search space size, genetic algorithm introduced in SENSEI might not be the best choice when the search space is exceptionally large. Modifications need to be made to SENSEI so that it works consistently well under the variety of the search space.

However, there is an expensive trade-off between enabling the genetic augmentation and the training time when the search space becomes very large. Although SENSEI has reached 83.0% robust accuracy using the original implementation of ResNet-50, the 200-epoch training lasted almost 29 hours. On the other hand, RANDOM achieves 81.9% robust accuracy under 10-hour time. Exhaustive training time, thus, becomes

another factor that needs to be considered when SENSEI is deployed in practical applications.

D. Future Plans

To improve the robustness of SENSEI, it is necessary to run more experiments using other datasets and model choices as well as different problems, for example, semantic segmentation. Also, it essentially needs to perform well on solving machine learning problems in other domains such as natural language processing and audio processing that heavily rely on deep neural network.

Secondly, the original paper has conducted the experiment with three types of transformations as well as with six types which are not included in this study. An experiment that involves a higher number of transformation types might provide us a bigger picture on the effect of search space size. Introducing non-linear image transformations are convincingly interesting as well.

VII. ACKNOWLEDGEMENT

This project cannot succeed without beneficial suggestions from Professor Shin Yoo and computing instance from School of Computing, Korea Advanced Institute of Science and Technology (KAIST). We would like to thank Saeyoon Oh who helped us request the instance from the department.

REFERENCES

[1] X. Gao, R. K. Saha, M. R. Prasad, and A. Roychoudhury, "Fuzz testing based data augmentation to improve robustness of deep neural networks," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, ser. ICSE '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1147–1158. [Online]. Available: https://doi.org/10.1145/3377811.3380415

[2] H. Fujiyoshi, T. Hirakawa, and T. Yamashita, "Deep learning-based image recognition for autonomous driving," *IATSS Research*, vol. 43, 12 2019.

[3] S. Balaban, "Deep learning and face recognition: the state of the art," *CoRR*, vol. abs/1902.03524, 2019. [Online]. Available: http://arxiv.org/abs/1902.03524

[4] D. Zoran, M. Chrzanowski, P. Huang, S. Gowal, A. Mott, and P. Kohli, "Towards robust image classification using sequential attention models," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2020, pp. 9480–9489. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPR42600.2020.00950

[5] L. Schott, J. Rauber, M. Bethge, and W. Brendel, "Towards the first adversarially robust neural network model on mnist," *arXiv preprint arXiv:1805.09190*, 2018.

[6] D. Zoran, M. Chrzanowski, P.-S. Huang, S. Gowal, A. Mott, and P. Kohl, "Towards robust image classification using sequential attention models," 2019.

[7] F. Tramer and D. Boneh, "Adversarial training and robustness for multiple perturbations," *arXiv preprint arXiv:1904.13000*, 2019.

[8] J. Rauber, W. Brendel, and M. Bethge, "Foolbox: A python toolbox to benchmark the robustness of machine learning models," *arXiv preprint arXiv:1707.04131*, 2017.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[10] S. Zagoruyko and N. Komodakis, "Wide residual networks," 2017.

[11] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf