Verifying Machine Learning based Image Classifiers using Metamorphic Testing: Appendices

Anonymous Author(s)

ABSTRACT

This document provides further material to the ISSTA 2018 submission. Appendix A contains plots of test loss values of orginal (non-buggy) code of different deep learning architectures on different data-sets. Appendix B contains plots of test loss for all the mutants for MR-1 & MR-2 of the ResNet application. Appendix C gives further experimental details on the execution of the ML applications with the MRs.

KEYWORDS

Verifying Machine Learning, Metamorphic Testing of Deep Learning Classifiers

ACM Reference Format:

Anonymous Author(s). 2018. Verifying Machine Learning based Image Classifiers using Metamorphic Testing: Appendices. In *Proceedings of ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2018).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.475/123_4

1 APPENDIX A

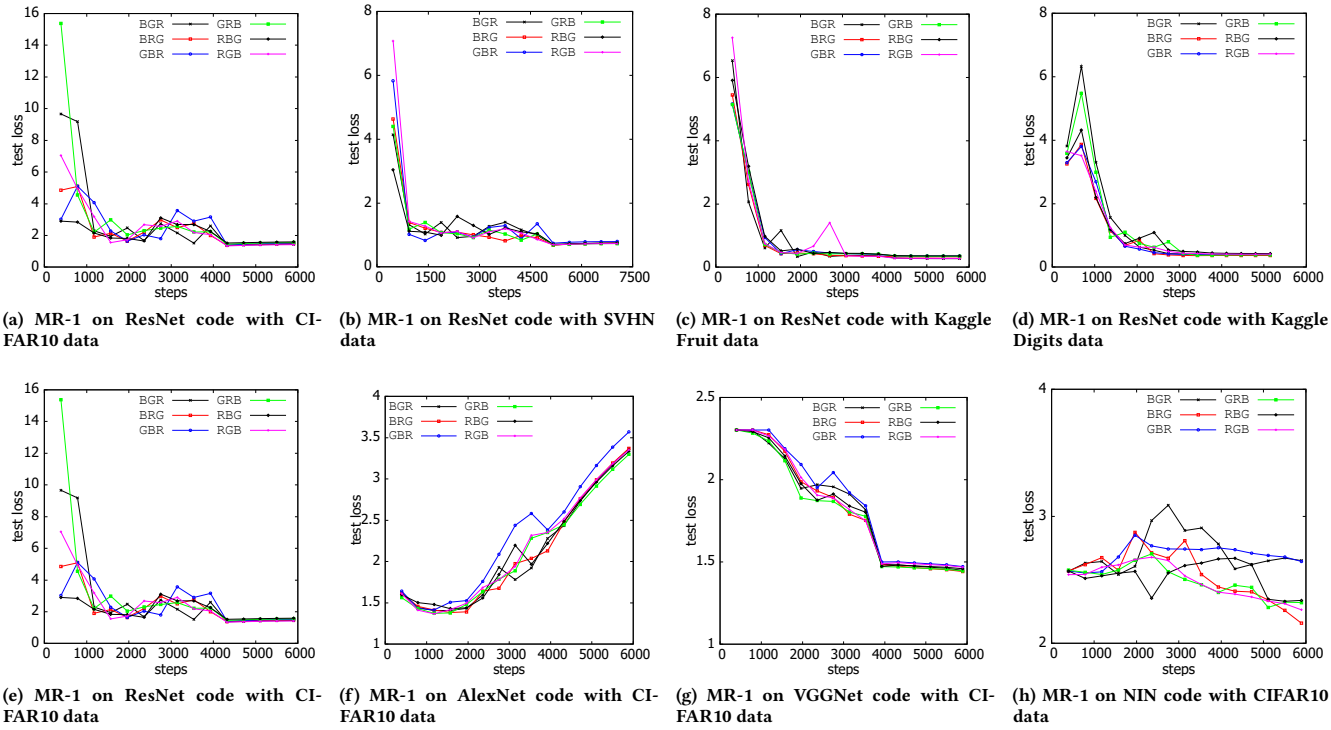
In this section, we will provide the results of the MR-1 (permuting the RGB order) and MR-2 (permuting the CONV Order on different data-sets and different architectures. These results are the output from original code (non-buggy). The combinations tried are shown in Table 1. The plot show that the expectation of having small variance in the test loss, due to the changes made in the input data as per MR-1 and MR-2, appears to hold. Particularly, the small variance (as measured by the maximum of the standard deviation of the loss) is not a specific property of ResNet architecture (because we see small variance across different architectures) and is not a specific property of the CIFAR-10 data-set (because we see small variance across different data-sets).

Deep Learning Architecture	Data-set	σ_{max} in test loss due to MR-1 (permute RGB)	σ_{max} in test loss due to MR-2 (permute CONV order)
ResNet	Cifar10	4.8	3.6
ResNet	SVHN [6]	1.4	3.3
ResNet	Kaggle Fruits [7]	0.8	2.1
ResNet	Kaggle digits [2]	1.1	0.9
AlexNet [4]	Cifar10	0.3	0.3
VGGNet [8]	Cifar10	0.1	0.1
NIN [5]	Cifar10	0.2	0.2

Table 1: Experiments conducted to validate MR-1 & MR-2

ISSTA 2018, 16-22 July, 2018, Amsterdam, The Netherlands

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4



(a) MR-1 on ResNet code with CIFAR10 data

(b) MR-1 on ResNet code with SVHN data

(c) MR-1 on ResNet code with Kaggle Fruit data

(d) MR-1 on ResNet code with Kaggle Digits data

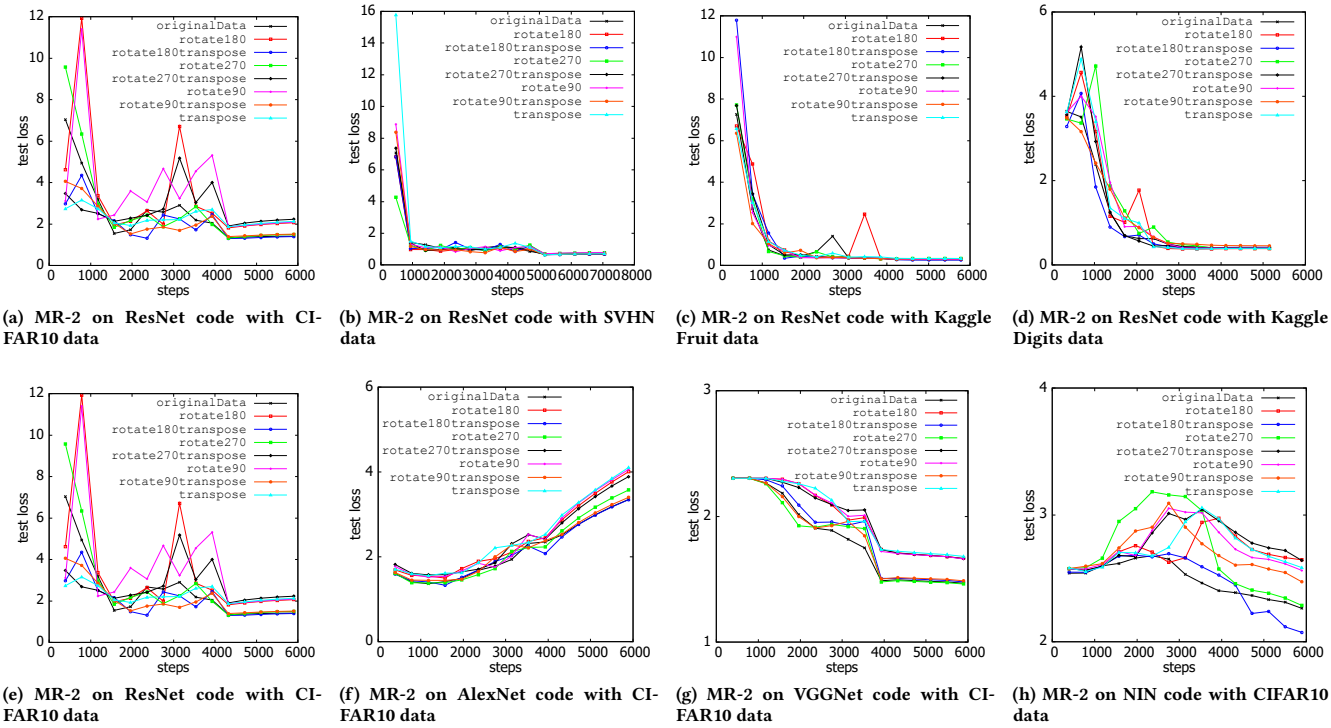
(e) MR-1 on ResNet code with CIFAR10 data

(f) MR-1 on AlexNet code with CIFAR10 data

(g) MR-1 on VGGNet code with CIFAR10 data

(h) MR-1 on NIN code with CIFAR10 data

Figure 1: Test loss from original code (non-buggy) due to changes MR-1: Permutation of RGB channels. Top four graphs for different data-sets on ResNet. Bottom four graphs for different architectures on CIFAR-10.



(a) MR-2 on ResNet code with CIFAR10 data

(b) MR-2 on ResNet code with SVHN data

(c) MR-2 on ResNet code with Kaggle Fruit data

(d) MR-2 on ResNet code with Kaggle Digits data

(e) MR-2 on ResNet code with CIFAR10 data

(f) MR-2 on AlexNet code with CIFAR10 data

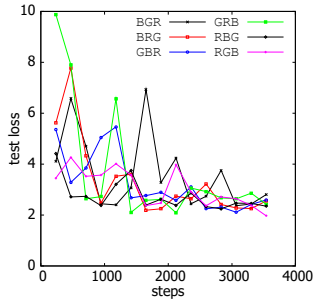
(g) MR-2 on VGGNet code with CIFAR10 data

(h) MR-2 on NIN code with CIFAR10 data

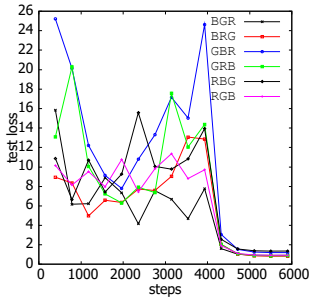
Figure 2: Validation loss from original code (non-buggy) due to changes of MR-2: Permutation of CONV order. Top four graphs for different data-sets on ResNet. Bottom four graphs for different architectures on CIFAR-10.

2 APPENDIX B

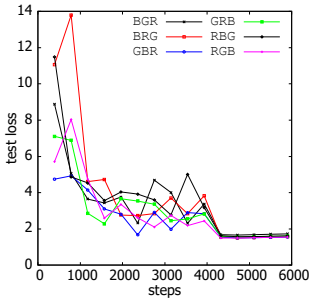
In this section, we will provide the plots of the test loss for all of the mutants when executed against MR-1 & MR-2. In some of the plots, we can see clear outliers and these outliers form the basis for catching the mutant. Quantitatively, we calcuate the maximum of the standard deviation of the test loss among the different variants of the MR. For example, in MR-1, at each step, 6 data-points of the test loss are available (each data-point corresponding to BGR, BRG, GBR, GRB, RBG, RGB). We calcuate the standard deviation (σ) among these six variants and use the maximum of the standard deviation across all steps as the metric to measure the results of MR-1 against a mutant. If this maximum is greater than a threshold, we say the mutant is killed. The threshold used for MR-1 & MR-2 is 9, however, we found that the threshold is fairly robust in the sense it makes little difference in the number of mutants captured.



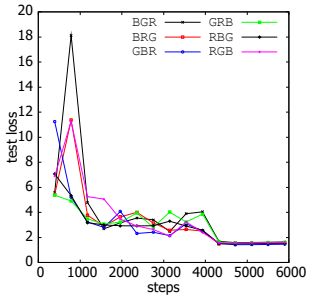
(a) MR-1 on mutant c9 with CIFAR10 data



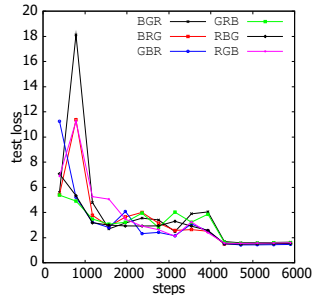
(b) MR-1 on mutant c29 with CIFAR10 data



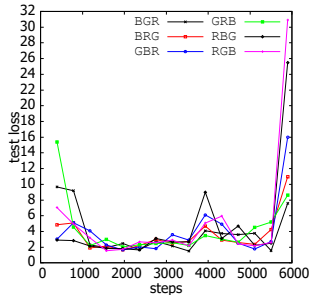
(c) MR-1 on mutant c30 with CIFAR10 data



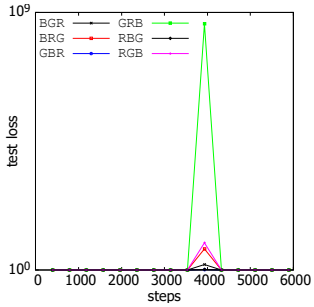
(d) MR-1 on mutant c31 with CIFAR10 data



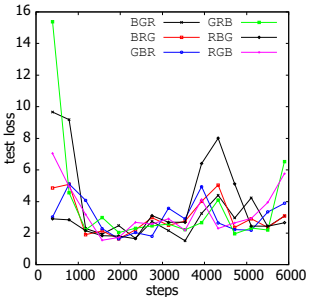
(e) MR-1 on mutant c32 with CIFAR10 data



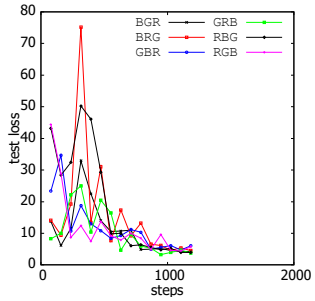
(f) MR-1 on mutant c43 with CIFAR10 data



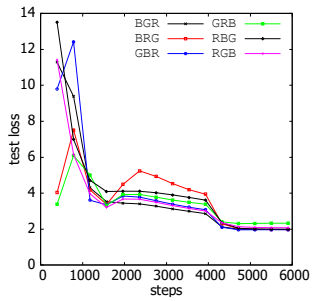
(g) MR-1 on mutant c44 with CIFAR10 data



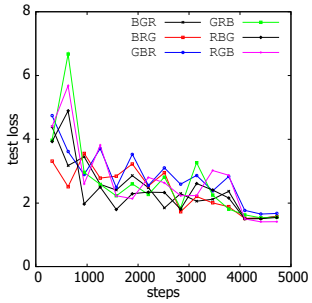
(h) MR-1 on mutant c45 with CIFAR10 data



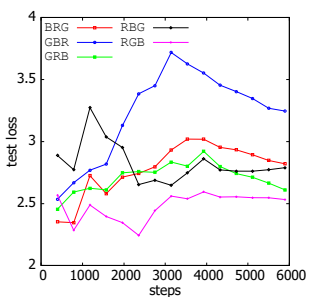
(i) MR-1 on mutant c49 with CIFAR10 data



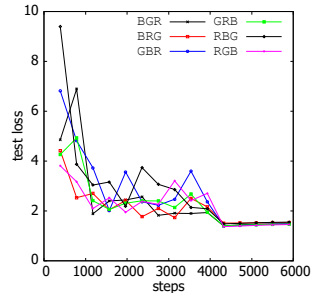
(j) MR-1 on mutant c50 with CIFAR10 data



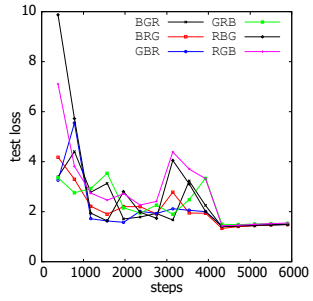
(k) MR-1 on mutant c116 with CIFAR10 data



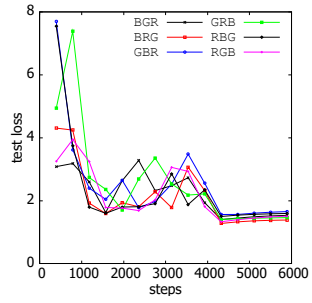
(l) MR-1 on mutant c221 with CIFAR10 data



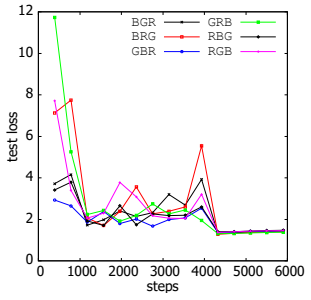
(m) MR-1 on mutant r6 with CIFAR10 data



(n) MR-1 on mutant r48 with CIFAR10 data

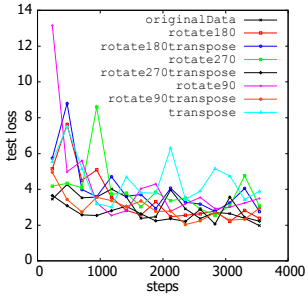


(o) MR-1 on mutant r49 with CIFAR10 data

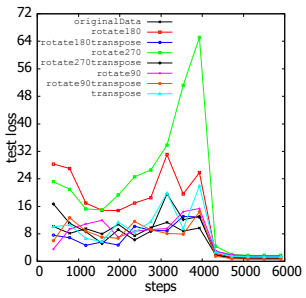


(p) MR-1 on mutant r67 with CIFAR10 data

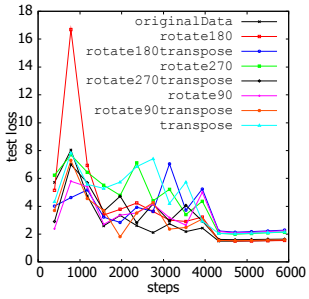
Figure 3: Test loss from MR-1: Permutation of RGB order for the mutants on ResNet Application.



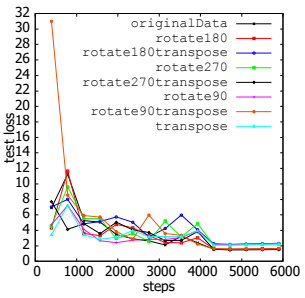
(a) MR-2 on mutant c9 with CIFAR10 data



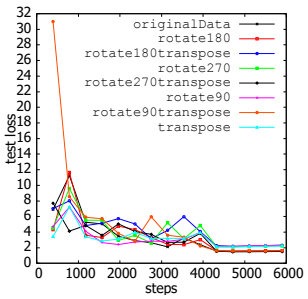
(b) MR-2 on mutant c29 with CIFAR10 data



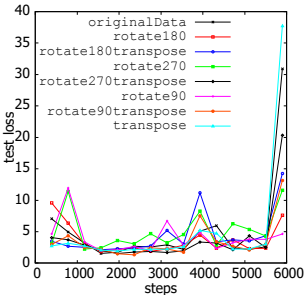
(c) MR-2 on mutant c30 with CIFAR10 data



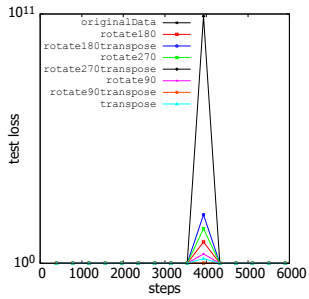
(d) MR-2 on mutant c31 with CIFAR10 data



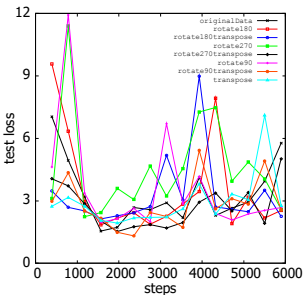
(e) MR-2 on mutant c32 with CIFAR10 data



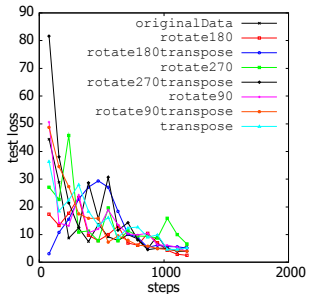
(f) MR-2 on mutant c43 with CIFAR10 data



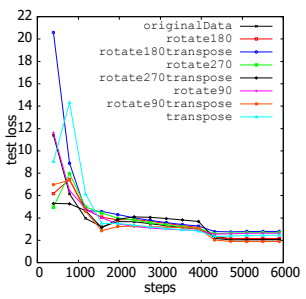
(g) MR-2 on mutant c44 with CIFAR10 data



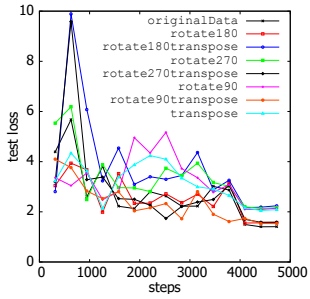
(h) MR-2 on mutant c45 with CIFAR10 data



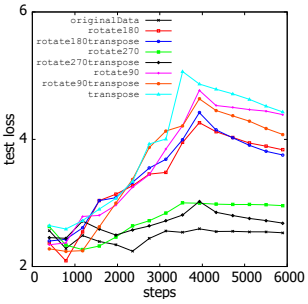
(i) MR-2 on mutant c49 with CIFAR10 data



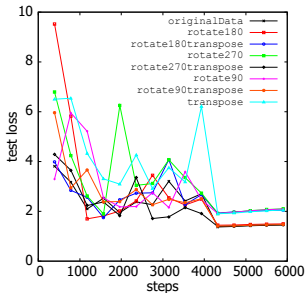
(j) MR-2 on mutant c50 with CIFAR10 data



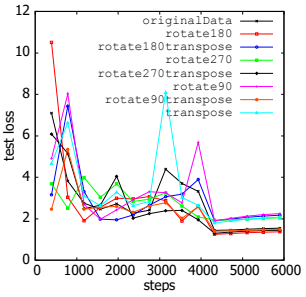
(k) MR-2 on mutant c116 with CIFAR10 data



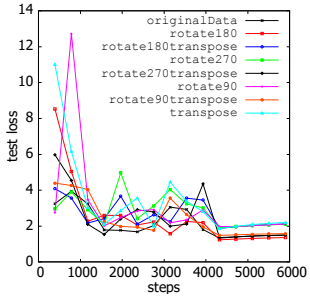
(l) MR-2 on mutant c221 with CIFAR10 data



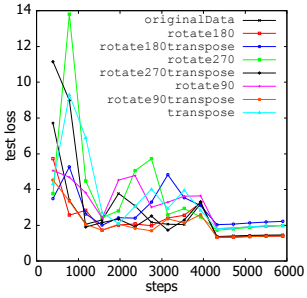
(m) MR-2 on mutant r6 with CIFAR10 data



(n) MR-2 on mutant r48 with CIFAR10 data



(o) MR-2 on mutant r49 with CIFAR10 data



(p) MR-2 on mutant r67 with CIFAR10 data

Figure 4: Test loss from from MR-2: Permutation of CONV order for the mutants of ResNet application.

3 APPENDIX C

In this section, we will provide some additional details pertaining to the execution of the mutants with the MRs.

3.0.1 Non-deterministic nature. One of the challenges working with the ResNet application (and possibly deep neural networks) is the stochastic nature of the output -i.e. for the same set of inputs, the application gives different results for two runs (see Table 2. 'Original' denotes the code as available in the application). Such differing outputs is a challenge for Metamorphic Testing, as all the MRs are based on relation between the outputs of subsequent runs. We investigated the reason for stochasticity and found it to be due to the random seeds that are used in TensorFlow. We updated the code to use a fixed seed for each run (this version is termed as as 'deterministic' in Table 2). Fixing the seed made the application deterministic when run on the CPU, unfortunately, the application was still stochastic when executed on the GPU. We could not determine the exact cause for the non-determinism on the GPU, but it appears to be an issue with the NVidia CUDA libraries [3] [1].

Code	Hardware	Num of Epochs	Run Num	Test accuracy	Test Loss	Loss μ	Loss σ
Original	CPU	5	1	0.23861386	5.9598055	5.2752532	1.898073321
Original	CPU	5	2	0.14554456	3.1301816		
Original	CPU	5	3	0.1079208	5.5404329		
Original	CPU	5	4	0.18613862	7.9412613		
Original	CPU	5	5	0.12871288	3.8045847		
Deterministic	CPU	5	1	0.13564357	5.4860325	5.4860325	0
Deterministic	CPU	5	2	0.13564357	5.4860325		
Deterministic	CPU	5	3	0.13564357	5.4860325		
Deterministic	CPU	5	4	0.13564357	5.4860325		
Deterministic	CPU	5	5	0.13564357	5.4860325		
Original	GPU	5	1	0.21089108	2.48737	7.30143576	3.728118329
Original	GPU	5	2	0.089108914	9.8327751		
Original	GPU	5	3	0.18613862	4.8291798		
Original	GPU	5	4	0.13663366	11.752038		
Original	GPU	5	5	0.10990099	7.6058159		
Deterministic	GPU	5	1	0.11881188	4.6922503	8.32788488	6.32573703
Deterministic	GPU	5	2	0.15643564	3.6995111		
Deterministic	GPU	5	3	0.1069307	18.292465		
Deterministic	GPU	5	4	0.14257425	10.990524		
Deterministic	GPU	5	5	0.12772277	3.964674		

Table 2: Experimental results on the original code. We modified the code by introducing fixed seeds for each run. This made the application deterministic on the CPU, however, the GPU was still non-deterministic.

REFERENCES

[1] antares1987. 2017. About Deterministic Behaviour of GPU implementation of tensorflow. https://github.com/tensorflow/tensorflow/issues/12871. (2017). [Online; accessed 15-Jan-2018].

[2] Olga Belitskaya. 2018. Handwritten Letters 2. https://www.kaggle.com/olgabelitskaya/handwritten-letters-2. (2018). [Online; accessed 12-Jan-2018].

[3] jkschin. 2017. Non-determinism Docs. https://github.com/tensorflow/tensorflow/pull/10636. (2017). [Online; accessed 15-Jan-2018].

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[5] Min Lin, Qiang Chen, and Shuicheng Yan. 2013. Network in network. *arXiv preprint arXiv:1312.4400* (2013).

[6] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. 2011. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, Vol. 2011. 5.

[7] Mihai Oltean. 2018. Fruits 360 Dataset. https://www.kaggle.com/moltean/fruits/data. (2018). [Online; accessed 12-Jan-2018].

[8] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).