Mutation Testing of Convolutional Neural Network (CNN)

Abdulhayyu H Lawal

March 2019

1 Motivation

In the past few years, the use of Deep Learning (DL) has become increasingly an important concept in many applications and automation systems for better analytical decisions. Learning from large amount of unlabeled data, deep learning models complex relationships between the given training data yielding expected output results. The quality of these deep learning architectures remain a major concern. Convolutional Neural Network (CNN), one of the deep learning architectures has proven to be very effective in the area of visual recognition and classification and have achieved success in many field areas, such as facial identification, objects analysis and in autonomous driving field. However, the quality, correctness and performance of the CNN architecture present a big challenge especially in highly mission critical applications [6], [11].

In this thesis proposal, Mutation Testing (MT) is chosen to be the type of testing technique to be applied to test and evaluate the quality and accuracy of Deep Neural Network (Convolutional Neural Network) in an autonomous driving system. By applying mutation testing, I am assessing and evaluating the CNN's quality, the quality of test cases, analyzing test data, and CNN's quality performance in a deep autonomous driving system. The test is planned to be applied by creating different test cases with some injected faults in the input training data source (Mutant data) and also another set of test cases mutant operators directly introduced to the convolutional neural network model program (Mutant program) [6]. The convolutional neural network architecture will be trained and executed separate and also in a system under test (SUT) consisting another regular application program (The Driving Controller system) [3]. Applying the proposed testing type to both the convolutional neural network architecture alone and to the system under test will assist in evaluating the quality of the convolutional neural network and to that of the the SUT based on the result produced [11].

2 Problem Statement

Majority of the research efforts conducted in the deep learning domain focused mainly on building more accurate models that can better achieve their intended goals [10]. However, very little work has been done on assuring the quality and correct performance of the deep learning applications [6],[10]. The need to measure, ensure and evaluate the quality of a convolutional neural network arises and becomes more and more crucial [10]. This thesis proposal looks to address this concern by evaluating the quality and correctness of a convolutional neural network in autonomous deep driving system. Mutation testing is a type of software testing technique to test the CNN model and the test data. This testing technique is to be carried out on a autonomous deep driving system that will consist of both the convolutional neural network and the Driving Controller Software. The convolutional neural network output and the output from combining the two under software under test approach will be analyzed to evaluate the performance of the convolutional neural network in an autonomous deep driving system.

3 State-of-the-Art and Related work

There has not been many research attempts on how mutation testing technique can be applied to specifically test Deep learning systems's quality and correctness [6]. Although, there are many mutation testing research work designed to test software systems, but proving the quality and effectiveness of deep learning systems by applying mutation testing have not been given the attention it deserves [11]. The most recent adequate research work that applied mutation testing on deep learning system was conducted by [6]Lei Ma et al. August 2018.

In this section, I have gathered some of the literature that are relevant to mutation testing, convolutional neural network or related to both. Besides the papers mentioned, there are more other research papers that are read and are considered to be supportive in the implementation of the thesis.

Lei et al. Aug 2018.[6] The paper proposes a mutation testing framework to measure the quality of deep learning systems. The paper proposed and designed two sets of mutation testing operators, a set of source-level mutation operators to inject faults to the source data and another set of model-level mutation operators to introduce faults to the model program. The paper evaluates the quality of the test case by analyzing the extent to which the injected and introduced faults can be detected.

Muhammed et al. 2017.[1] This paper conducted a review of different convolution neural network architectures, tested and evaluated their performance especially in autonomous driving systems. The paper also explained in details how convolutional neural network are used and applied in autonomous vehicles application, helping to make better critical decisions.

Xie et al. 2011.[10] This is a research paper focuses on applying Metamorphic Testing to test and Validate Machine Learning Classifiers. Machine learning classification applications have become more crucial due to their prevalence in society and the paper help to address the quality of such applications or programs by applying the test technique and validating the results. The paper conducted cross-validation and mutation analysis to prove the effectiveness and correctness of the classifiers.

Rene et al. 2014.[5] provides some suggestions on how to improve mutation testing analysis. They examined how mutants faults are differentiated from real faults in software testing.

Vincent et al. 2014.[2] focused on mutation analysis as a way to systematically qualify and improve a set of test data for detecting faults in a program under test. They created faulty versions of programs from original programs by systematically injecting one single fault per version of the program. They measured the ability to highlight the fault injected in each mutated version (killing these mutants).

Junhuo et al. 2017.[4] proposed an approach for validating the classification accuracy of a deep learning framework that includes a convolutional neural network, a deep learning executing environment, and a massive image data set. They proposes and explains approaches to validate deep learning classifiers using metamorphic validation approach.

Goran et al. 2017.[8] showed how mutation testing can be applied to test industrial applications. Their work will help in understanding the importance of mutation testing, quantifying the costs of unproductive mutants and suggesting ways to effectively handle the current challenges of efficiently and effectively applying mutation testing in an industrial-scale software development process.

Chen et al. 2015.[3] developed a model that is built upon the state-of-the-art deep CNN framework that automatically learn image features for estimating driving affordance and driving decision behavior. The framework and model implemented by them will be used in my thesis work.

Xiaowei Huang 2017.[9] talked about verification and testing of deep learning systems. They verify the DL systems by global optimisation approach and by layer-by-layer approach with exhaustive search.

Murphy 2016.[7] discussed some of the major network parameters and learning techniques related to convolutional neural network. They provided an overview of some CNN architectures and their recent developments.

4 Research Goal

The main goal of this thesis is to apply mutation testing to convolutional neural network and to evaluate the quality of the model in an autonomous deep driving system by injecting faults to the CNN model and to a driving controller system in SUT. The test is planned to be carried out in two test cases. First, the convolutional neural network model is mutated (input data and program model) only and passed to the original unmutated driving controller. Second, both the deep learning model and the original driving controller is mutated (Software Under Test).

The driving decisions from the original models and the mutated models are compared. The accuracy and the quality of convolutional neural networks is then determined based on the driving decisions from both cases and the model's ability to make correct driving decisions. In the process of evaluating the quality of test cases, the model and that of the SUT the following research questions will be addressed.

- Can mutation testing be applied to estimate and evaluate the quality of convolutional neural network through different test cases (Mutants) AND can the CNN model be tested and executed togehter with the driving controller under software under test (SUT)?
- How good or bad is a convolutional neural network's quality and performance accuracy in case of introducing faults/mutants?
- What makes more sense to test? test cases, input data, the CNN model, or both as a single system including the driving controller as SUT?
- Which of the test cases perform better or has the least impact as mutants?
- Which convolutional neural network architecture type is more resilient to mutation? Atleast two or more CNN architecture will be tested.
- What is the overall quality and effectiveness of convolutional neural network after applying mutation testing?

5 Proposed Methodology

5.1 Test Subject

The data for training and testing the model will be collected from TORCS game engine. TORCS game engine is an open source racing car simulator widely used in artificial intelligence research field. Images will be recorded from a driving scenario in the TORCS environment. Alternatively, the dataset already recorded by [3] can be reused as well. Either the training or testing or both fractions of the data can be mutated and compared with the corresponding unmutated data results. From the perspective of the convolutional neural network, a mutation is a change in the CNN's neural architecture which will then be checked to see if after the change there is also change in the output and also if the output of the driving controller from [3] behaves differently given different input (result from mutated CNN).

5.2 Implementation

The implementation of this thesis will involve designing of mutation testing technique that will execute convolutional neural network and the driving controller system under different mutations. The convolutional neural network and the driving controller together is referred to as a software under test (SUT). Mutation is achieved by designing mutations operators that introduce potential faults into the software under test to create mutated versions. Three different sets of mutation operators are to be designed (see Table 1), source-level mutation operators that introduced faults into the source data input of the deep learning system, model-level mutation operators that introduced faults into the deep learning system and application-level mutation operators that are injected into the driving controller subsystem in the SUT. In initial execution, the SUT will be without faults/mutants and in the further attempts, faults/mutants can be introduced in three different test rounds. In the first test round faults are introduced into the convolutional neural network only (training data, training program or both), second round of the test is to introduce faults to the driving controller only and finally, faults are introduced into the SUT (convolutional neural network and the driving controller), this is to enable accurate evaluation of each of the different system/program and to answer some of the research questions. For example, evaluating system/program that is more resilient to mutation testing.

The process of applying the test technique to the convolutional neural network (see Figure 1) and into the software under test (see Figure 2) involves re-training of the models any time faults are injected to obtained mutated versions. Each mutant model version ($M_1{}'$, $M_2{}'...M_n{}'$) [6] generated is executed and analyzed against the test set T in correspondence to original system model M. With a particular given input test set of t ϵ T, M' is analyzed against M and if the results of M and M' are inconsistent then faults/mutants are detected [6]. The software under testing in figure 2 showed the overview process of the system under test. The mutation process of the driving controller or the SUT as a whole will involve similar process of general mutation where a mutated version of the original system is produced through faults injection. The model or the system's quality is nevertheless evaluated and analyzed to make conclusion. Automated training data mutant generator for mutating the input data and python training program mutant generator will be used to generate the mutants in both models. The implementation will run on Linux and will be built based on Keras consisting converted models from Caffe, and Tensorflow.

5.3 Mutation Testing Operators

The mutation operators in table 1 are some of the proposed mutation operators to be used in mutating of the programs or systems. The actual operators to be used will depend on the design and implementation process and on the research questions or on what I plan to evaluate from the model.

5.4 Evaluation

I will analyzed and evaluate convolutional neural network's quality by applying mutation testing technique. The evaluation will be based on the mutation score that will be obtained from mutation process execution, the number of faults/mutants killed in the mutated version detected by the test case and in comparison against the original version of the system. Mutation score is calculated by comparing the ratio at which the input test set T is able to detect and killed mutants M' against all generated and introduced mutants (number of mutant killed over the number of all mutants generated) [6]. When the SUT

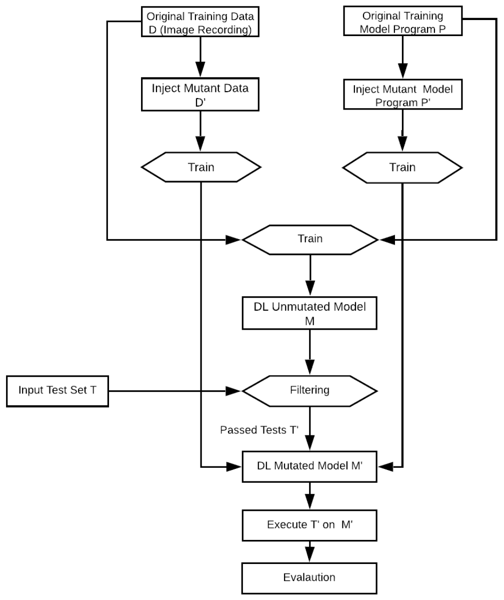

Figure 1: Mutation Testing Architecture [6].

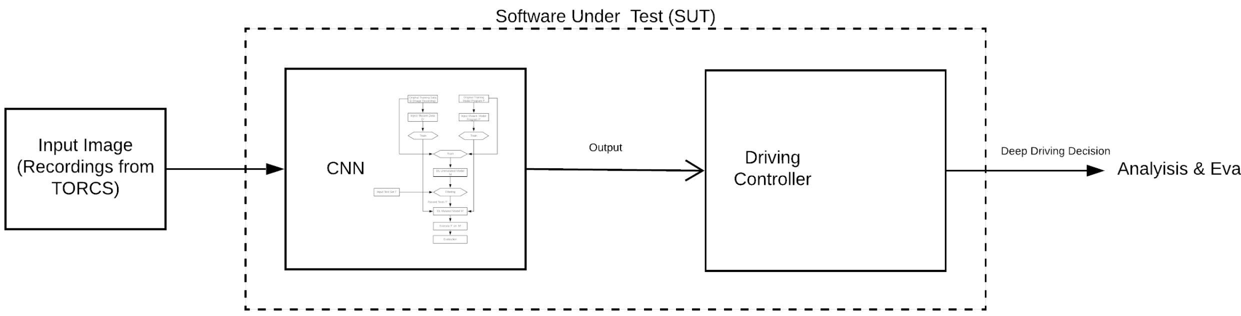

Figure 2: Overview of the Software Under Test.

List of the proposed mutation operators		
Image Recording Mutant Type	Model Program Mutant Operators	Driving Controller Mutant Operators
Data Missing	Layer Deactivation	Decision Tempering
Data Repetition	Activation Function Removal	Wrong Input
Data Shuffle	Neuron Switch	Data/input Missing

Table 1: Mutation testing proposed operators [6].

is evaluated, only the mutated output version will be compared against the original output version.

Upon completion of this thesis, the research question to whether mutation testing can be applied to evaluate convolutional neural network's quality and CNN in an autonomous driving would be successfully answered and also be able to tell which mutants or faults have more impact on the performance quality of the CNN. Depending on the number of convolutional neural network architecture type is tested by applying mutation testing, each architecture will be executed and evaluated separately based on its performance quality, evaluating which of the architecture is more resilient to mutation testing.

6 Requirement List and Schedule

6.1 Requirement List

The thesis requirement is organized into three categories as shown in Table 1:

- MUST HAVE requirements are the most important part of the thesis that I must implement and accomplish.
- MIGHT HAVE requirements are to support and improve the quality of this thesis.
- MUST NOT requirements define the scope of this thesis that are not planned to be implemented.

6.2 Schedule

The schedule in Table 2 shows how the thesis is planned to be carried out, highlighting time allocation for phases, duration and tasks to be involved in the thesis.

Requirement List	
Requirements Options	Remark
Must Have	• Applying Mutation Testing to deep learning system (convolutional neural network) and to the mixed software system, the system under test (SUT). • Evaluating the deep learning system and the SUT base on the output result from the test execution by comparing mutated result against the original unmutated program system.
Might	• Mutate atleast two different convolutional neural network architecture type. Find out which architecture is more resilient to mutation testing. • Perform the test also on some widely used datasets apart from the training dataset (Recordings from the TORCS environment).
May Not	• No own driving decision controller program will be developed, the driving controller developed by [3] paper will be modified and used. • Sending of output results back to TORCS environment after execution may not be implemented. The output or the driving decision produced after comparing will be analyzed and evaluated.

Table 2: Thesis requirement list.

Schedule		
Phase	Duration in Months	Tasks
Initiation, Planning, Proposal and presentation	2-3	topic research, -related material gathering, -state-of-the-art research, proposal documentation, - submission and presentation of proposal
Implementation and Documentation	1-4	reading, -coding, -design, -documentation -discussion/feedback
Testing and Submission	2	final testing, -presentation, -submission, -closing

Table 3: Time allocation

7 Conclusion

By the end of successful implementation of this proposed thesis. There will be two evaluation analysis reports, evaluation analysis on the quality performance of the convolutional neural network and another evaluation analysis on the software under test system. It will become clear to whether mutation testing can be applied to evaluate convolutional neural network's quality and if the same testing technique can also be carried out when convolutional neural network is executed along another deep driving application of an autonomous driving system. I proposed to design three different set of mutation operators that can be used in introducing faults to the systems to evaluate their quality performance. A mutation score framework is to be used to find out the number of mutants detected and killed from the mutated system version. The higher the mutation score obtained, the higher the system's quality performance.

References

[1] M. Al-Qizwini, I. Barjasteh, H. Al-Qassab, and H. Radha. Deep learning algorithm for autonomous driving using googlenet. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 89–96, June 2017.

[2] Vincent Aranega, Jean-Marie Mottu, Anne Etien, Thomas Degueule, Benoit Baudry, and Jean-Luc Dekeyser. Towards an automation of the mutation analysis dedicated to model transformation. *Software Testing, Verification and Reliability*, 25(5-7):653–683, 2015.

[3] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[4] J. Ding, X. Kang, and X. Hu. Validating a deep learning framework by metamorphic testing. In *2017 IEEE/ACM 2nd International Workshop on Metamorphic Testing (MET)*, pages 28–34, May 2017.

[5] Rene Just, Darioush Jalali, Laura Inozemtseva, Michael D Ernst, Reid Holmes, and Gordon Fraser. Are mutants a valid substitute for real faults in software testing? 11 2014.

[6] Lei Ma, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Felix Juefei-Xu, Chao Xie, Li Li, Yang Liu, Jianjun Zhao, et al. Deepmutation: Mutation testing of deep learning systems. In *2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE)*, pages 100–111. IEEE, 2018.

[7] John Murphy. An overview of convolutional neural network architectures for deep learning. 2016.

[8] G. Petrovic, M. Ivankovic, B. Kurtz, P. Ammann, and R. Just. An industrial application of mutation testing: Lessons, challenges, and research directions. In *2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 47–53, April 2018.

[9] Youcheng Sun, Xiaowei Huang, and Daniel Kroening. Testing deep neural networks. *CoRR*, abs/1803.04792, 2018.

[10] Xiaoyuan Xie, Joshua WK Ho, Christian Murphy, Gail Kaiser, Baowen Xu, and Tsong Yueh Chen. Testing and validating machine learning classifiers by metamorphic testing. *Journal of Systems and Software*, 84(4):544–558, 2011.

[11] Yuhao Zhang, Yifan Chen, Shing-Chi Cheung, Yingfei Xiong, and Lu Zhang. An empirical study on tensorflow program bugs. In *Proceedings*

of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis, ISSTA 2018, pages 129–140, New York, NY, USA, 2018. ACM.