Revisiting Neuron Coverage and its Application to Test Generation

Stephanie Abrecht[1], Maram Akila[2], Sujan Sai Gannamaneni[2], Konrad Groh[1]

Christian Heinzemann[1], Sebastian Houben[2], and Matthias Woehrle[1]

1 Robert Bosch GmbH, Germany

2 Fraunhofer IAIS, Germany

Outline

- Motivation
  - From software testing to Neuron Coverage
- Selection of research questions
- Experiment & setup
- Conclusion

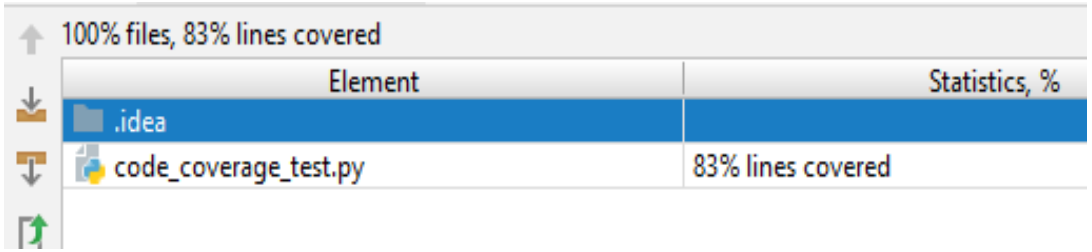
Motivation
Coverage in software testing

- Statement coverage
  - Check whether statement in code is executed
- Academic perspective on coverage requirements[1]
  - Need a static model
  - Possibility to achieve full coverage
  - Actions to increase coverage are known (new tests)
  - Exit strategy
- But not restricted to lines of code
  - Ex: Coverage of requirements

code_coverage_test.py

```
1   def foo():
2       print("never called")
3
4
5   def main():
6       print("executed")
7
8
9   if __name__ == '__main__':
10      main()
11
```

Statement coverage in PyCharm



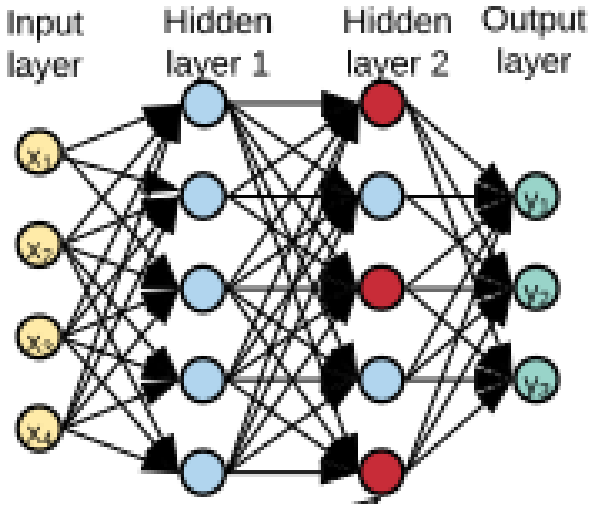
[1] - Bron, A., Farchi, E., Magid, Y., Nir, Y., Ur, S.: Applications of synchronization coverage. In: Symposium on Principles and Practice of Parallel Programming. pp. 206-212 (2005)

Motivation
Neuron Coverage

- Code coverage for ML models is *not* a good testing strategy
  - Almost always 100 % code coverage is achieved

- Alternative: Check coverage of neurons (DeepXplore[1])
  - *IF Neuron output > threshold THEN*

    return active
  - Once a neuron is active for one test sample, it is considered active throughout
- But, how reliable is coverage as testing strategy?
  - Is definition ”reliable“ if 100 % Neuron Coverage can be easily obtained?

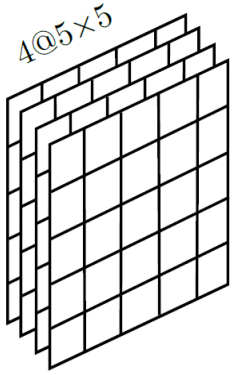


[1] - Pei, K., Cao, Y., Yang, J., Jana, S.: DeepXplore: Automated whitebox testing of deep learning systems. In: Symposium on Operating Systems Principles. pp. 1-18 (2017)
Image source - Gerasimou, Simos, et al. "Importance-Driven Deep Learning System Testing." *arXiv preprint arXiv:2002.03433* (2020).

Definition of Neuron Coverage

- **Code**: Line was run or not (binary criterion)
- **Neuron**: Requires threshold
  - Here: ReLU, natural offset = 0
- How to count neurons?
  - Trivial for Fully Connected Layer
  - **Differing definitions and results for Convolutional Layer**
    - Count each activation output as single neuron (DXC)
    - Count each element of activation output as neuron (LWC)
  - Should max pooling layer be considered in neuron count

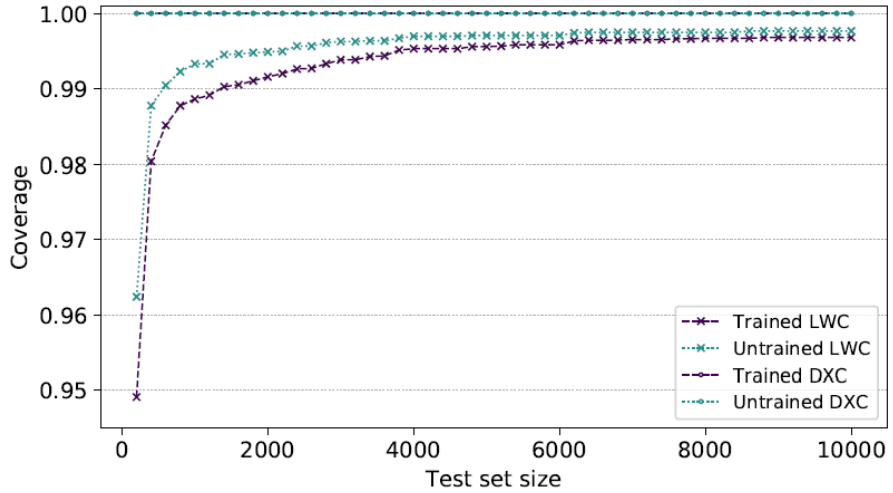
- $Final\ coverage = \frac{Number\ of\ covered\ neurons}{Total\ number\ of\ neurons}$



Number of Neurons
DXC = 4
LWC = 4×5×5 = 100

Architecture	DXC	Sun *et al.* [17]	LWC
MLP 784x128x64x10	986	976	986
MNIST CNN in [17]	258	14,208	17,738
LeNet-5	258	6,508	8,094
VGG-19	16,168	14,861,288	16,391,656

Neuron counts for different definitions and several architectures.



Coverage of first Convolutional Layer

Research Questions and Experiments

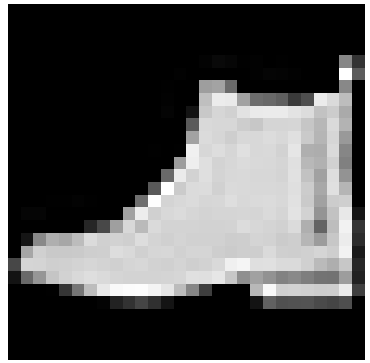
1. What impact does augmentation have on coverage in initial layers?
2. Call full coverage be achieved with only a subset of classes?
3. Are differential test generation methods better than test time augmentation methods?

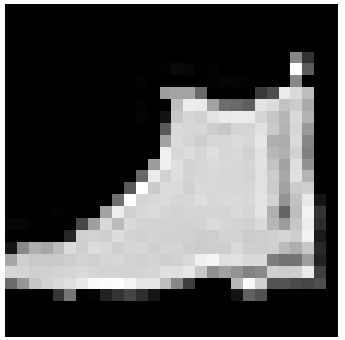
- Greyed out parts can be found in the paper

Experiment Setup

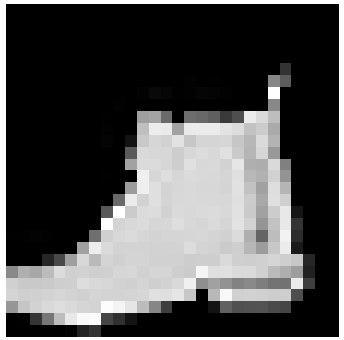
- **Datasets**
  - MNIST and F-MNIST
- **Networks**
  - Lenet5 and a simple 6-layer CNN
- **Data augmentations**
  - Weak augmentation
    - translation bound to ±10% and a rotation bound to ±5 degrees
  - Strong augmentation
    - translation bound to ±20% and a rotation bound to ±10 degrees

Original





Weak augmentation



Strong augmentation

Impact of Augmentation on Coverage in Initial Layers

- **Observations**
  - 100% coverage not achieved with standard MNIST testset
  - Inactive neurons are not caused by training regime but testset itself
  - Weak augmentation and strong augmentation strengthen testset

Coverage of first Convolutional Layer. Number of inactive neurons in parenthesis

Train On \ Test on	Standard	Weak	Strong	Strong-500
Weak Augment.	0.9968 (37)	0.9999 (1)	1.0 (0)	1.0 (0)
Strong Augment.	0.9950 (58)	0.9997 (3)	1.0 (0)	1.0 (0)

Conclusion

- Neuron definition matters
- Layer-wise coverage provides a more sensitive method for testing
- For MNIST-like tasks,
  - Augmentations are good enough to increase coverage
  - Reliability of test in question as high coverage is easily obtained

  - High coverage level can be obtained from only subset of classes (see paper)
  - Augmentations achieve higher coverage than differential test generation methods(see paper)

- For deeper DNNs on more complex tasks
  - Coverage-guided semantic test generation remains to be investigated

Thank you