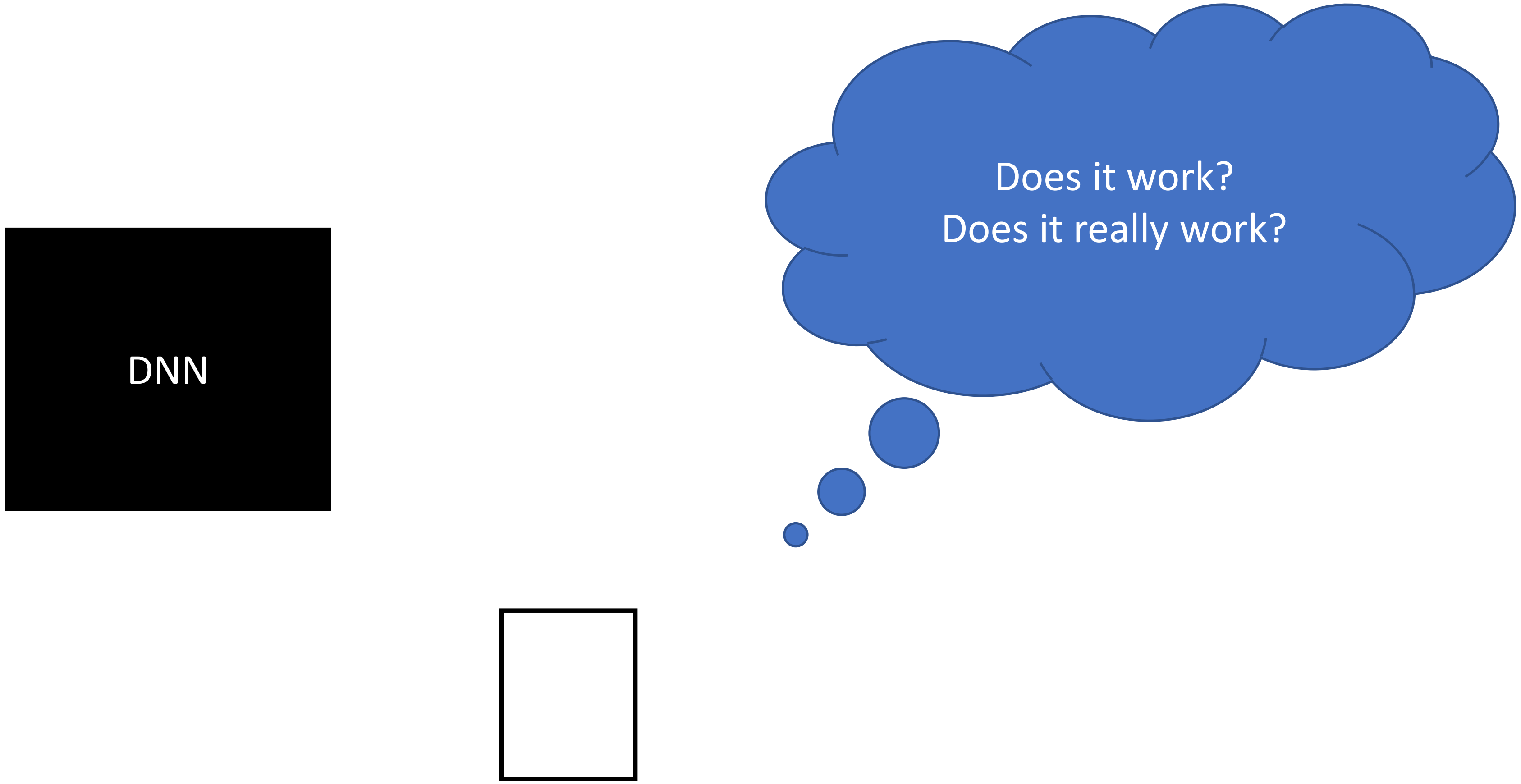
Not all Neurons are created equal:
Towards a feature level Deep Neural Network Test Coverage Metric

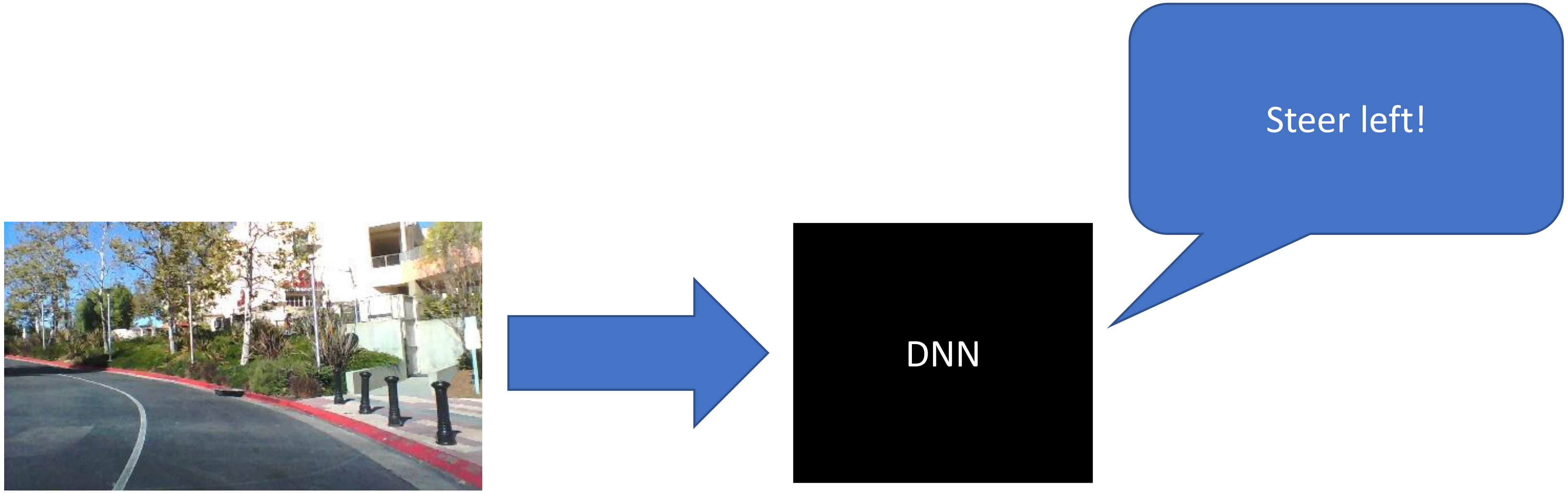
Nils Wenzler - CSC2125: Topics in Software Engineering Winter 2019

Problem

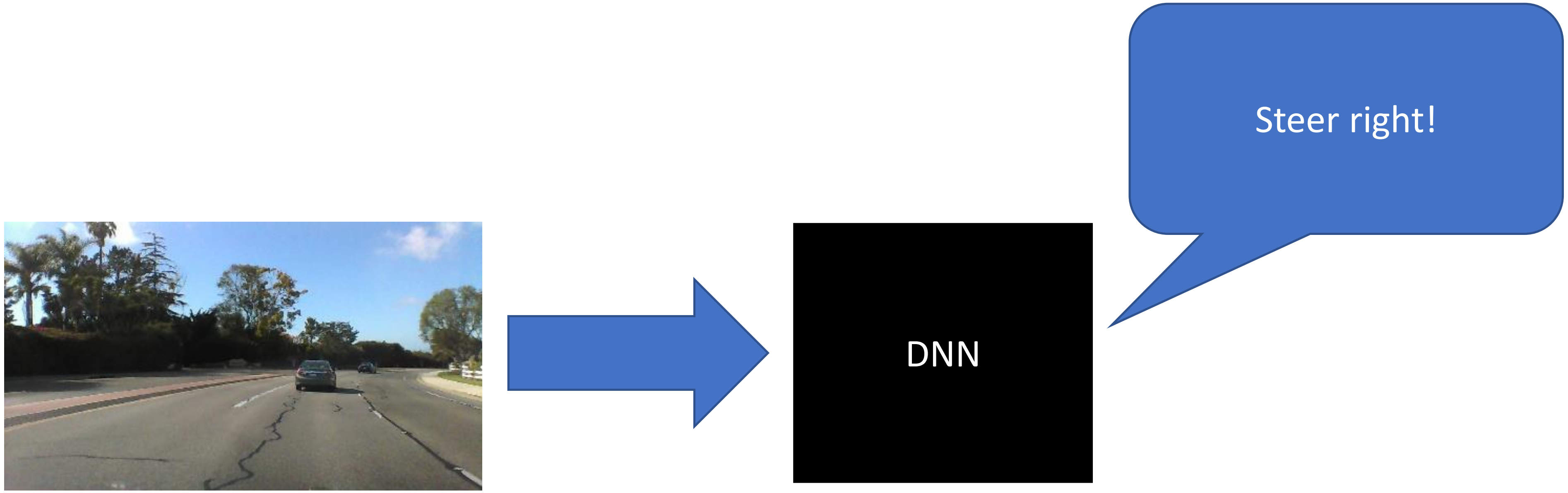
Problem



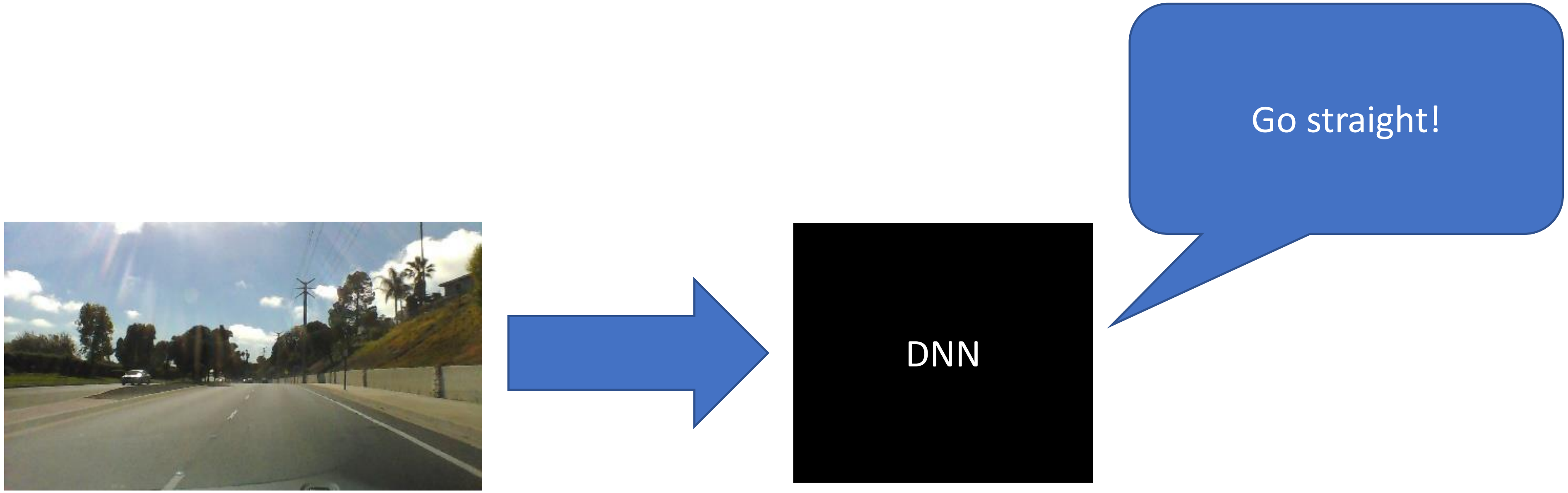
Problem



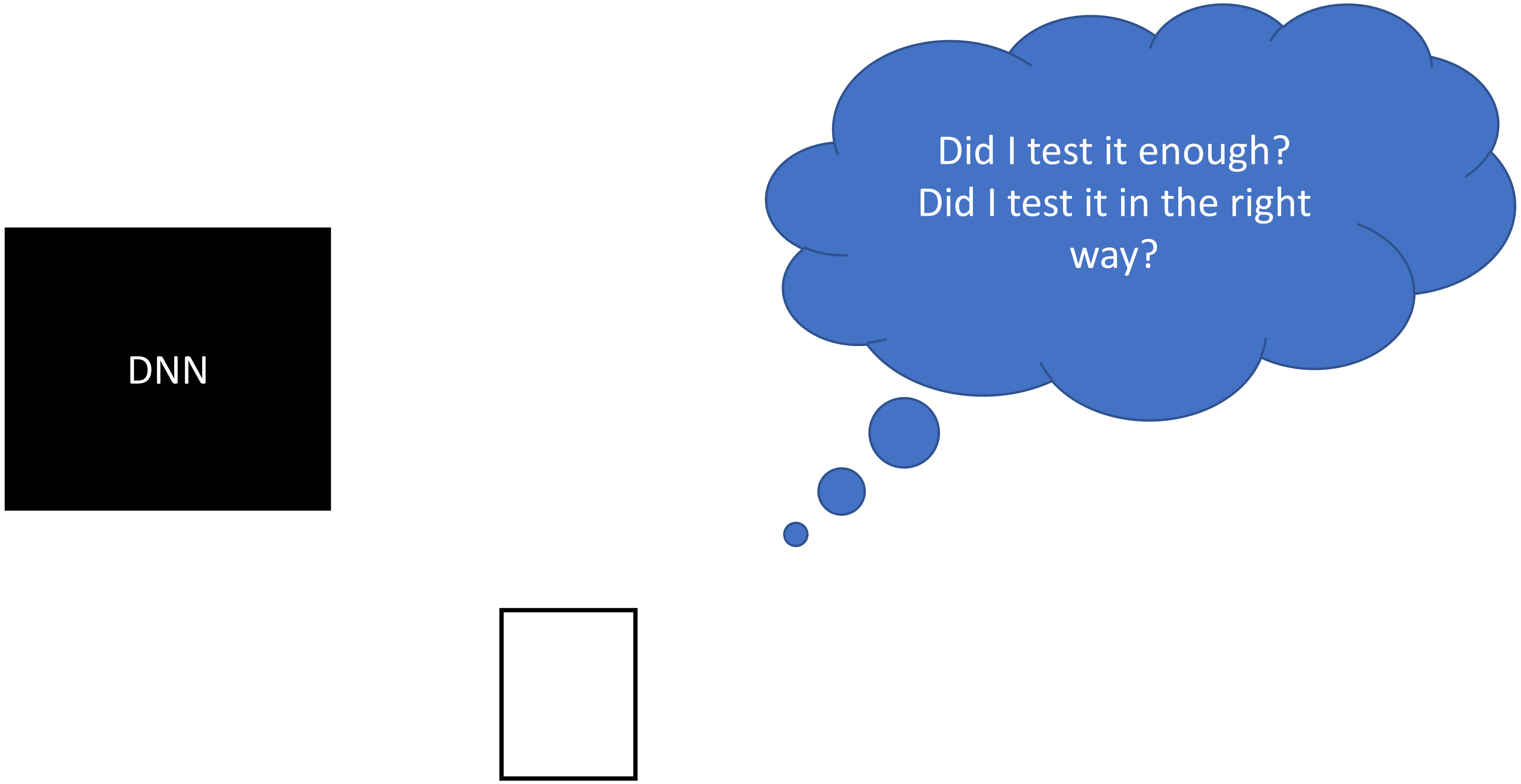
Problem



Problem



Problem



Structure

1. Problem
2. Current DNN Test Coverage Metrics
3. α-Bin Coverage
4. Practical Evaluation

General Approach

Use a test coverage metric for

- Building test suites that
- Cover all significant behaviours of a deep neural network

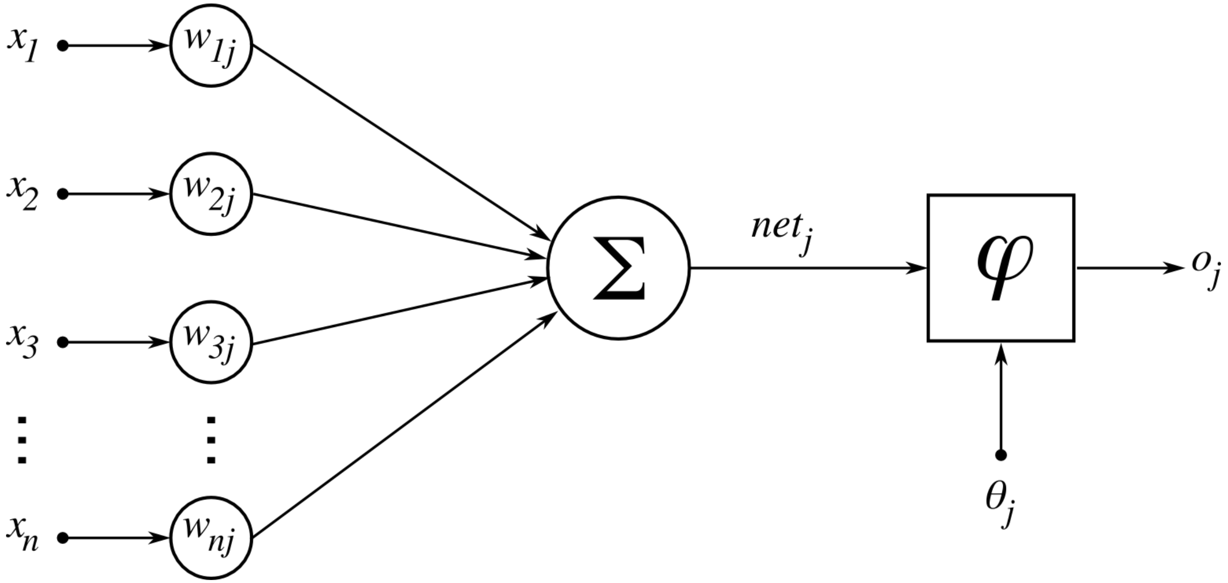
Not a proof of correctness but evidence towards correctness!

Current DNN Test Coverage Metrics

Current DNN Test Coverage Metrics

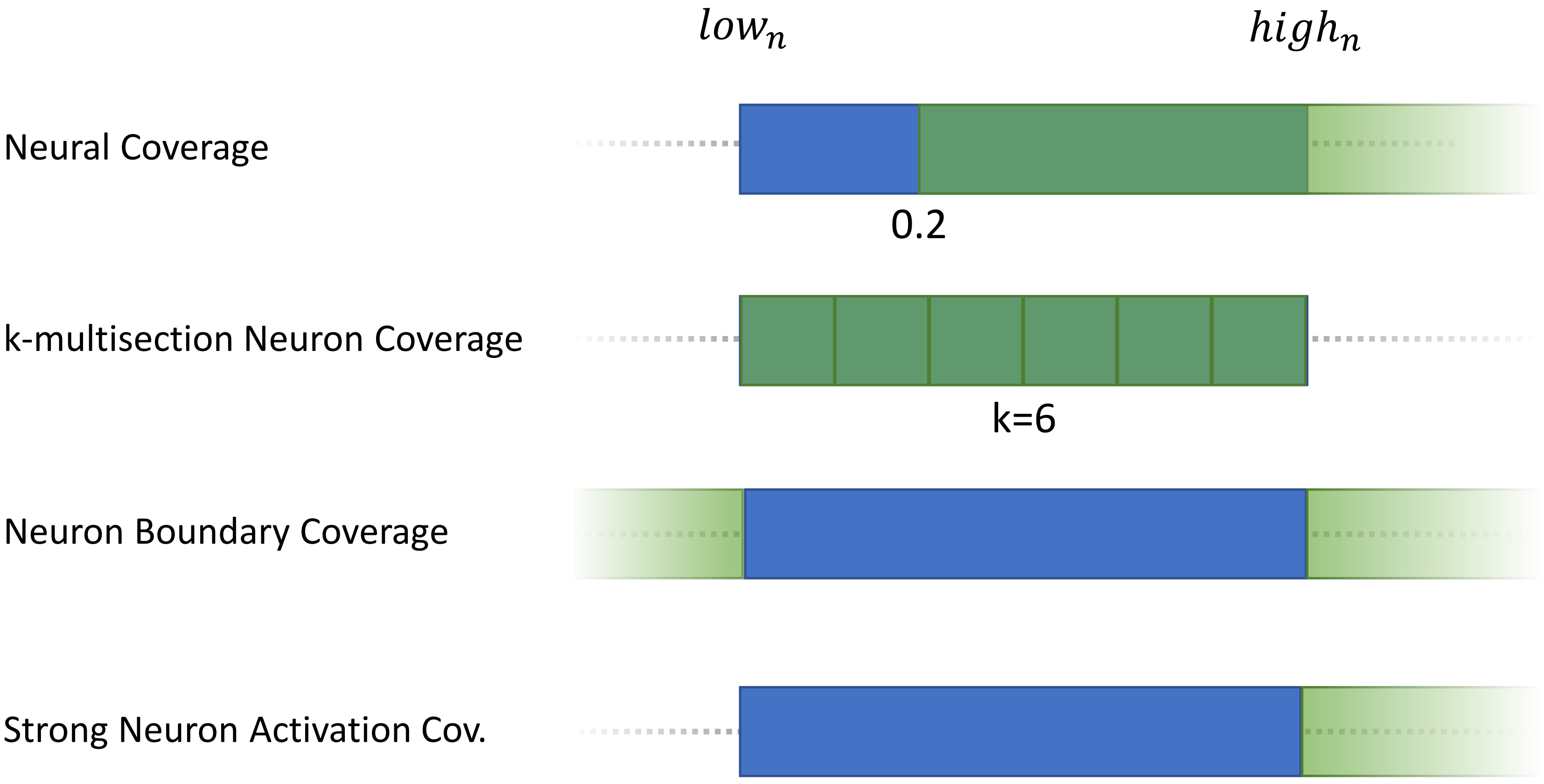
- High research interest
- White-box testing
- Focused on single neurons

Current DNN Test Coverage Metrics



low_n: lowest output value during training
$high_n$: highest output value during training

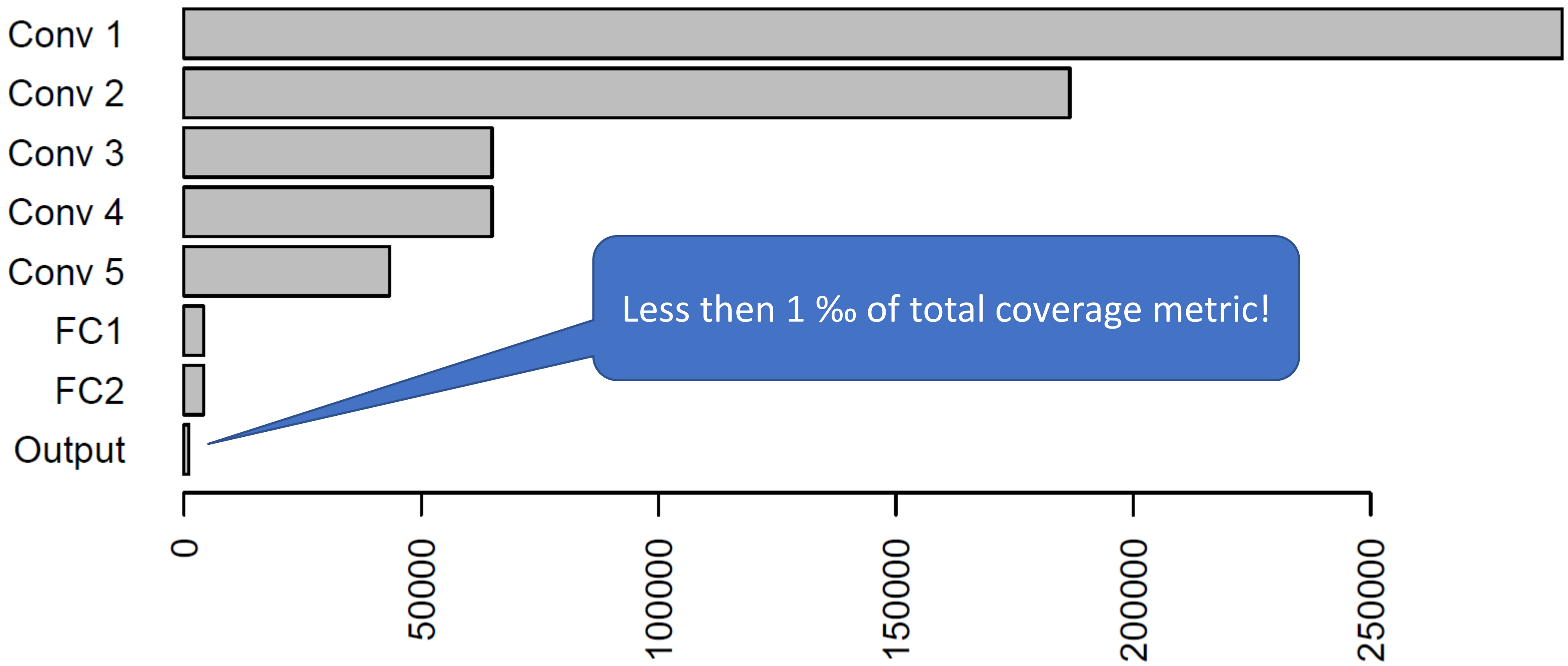
Current DNN Test Coverage Metrics



Structure

1. Problem
2. Current DNN Test Coverage Metrics
3. α-Bin Coverage
4. Practical Evaluation

Yet another metric?



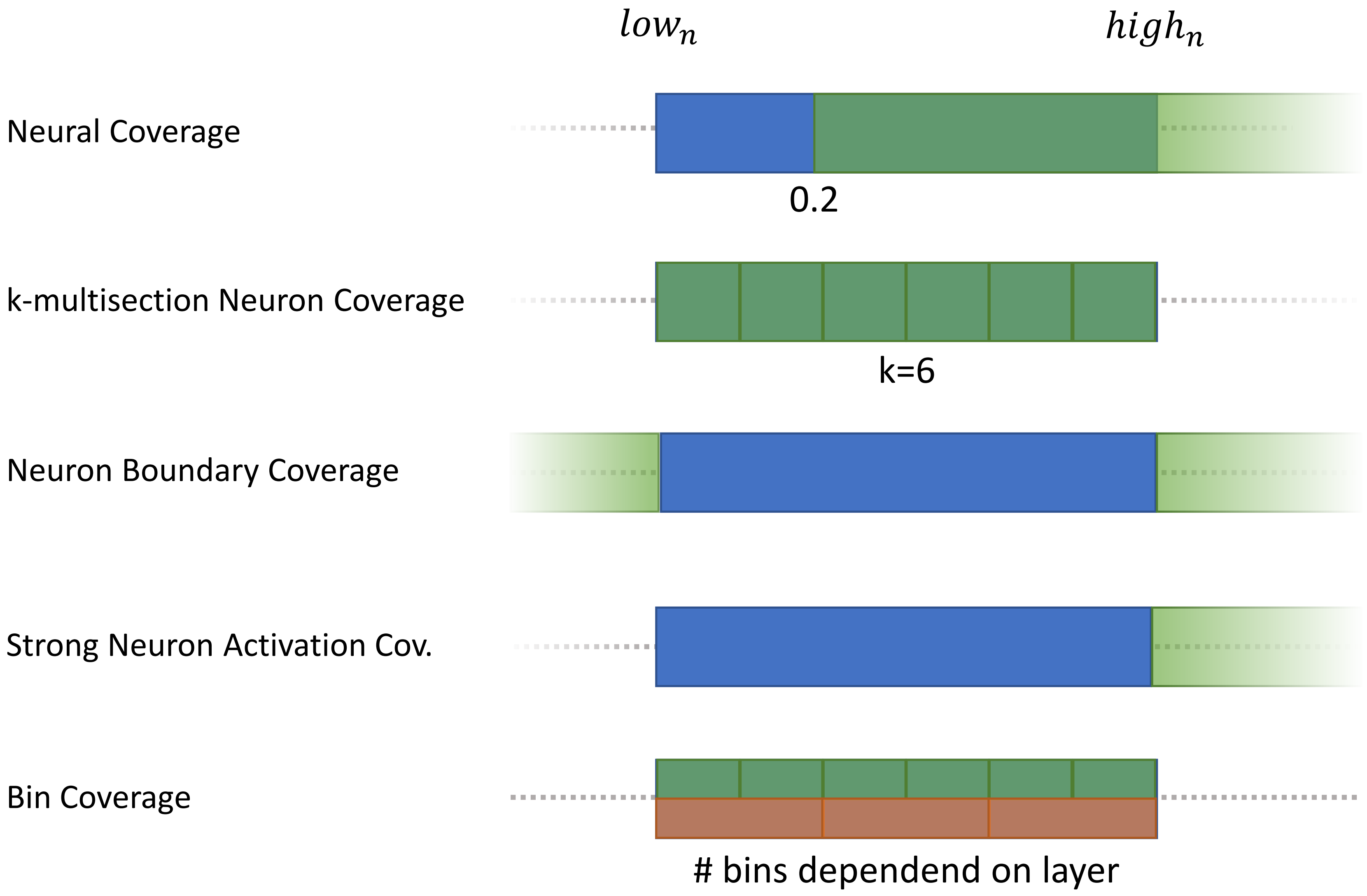
Number of neurons per layer in AlexNet

Not all Neurons are created equal

Current metrics put equal emphasis on each neuron, but:

Is a first layer neuron as important as an output layer neuron?

Make use of domain specific knowledge concerning layer architectures!

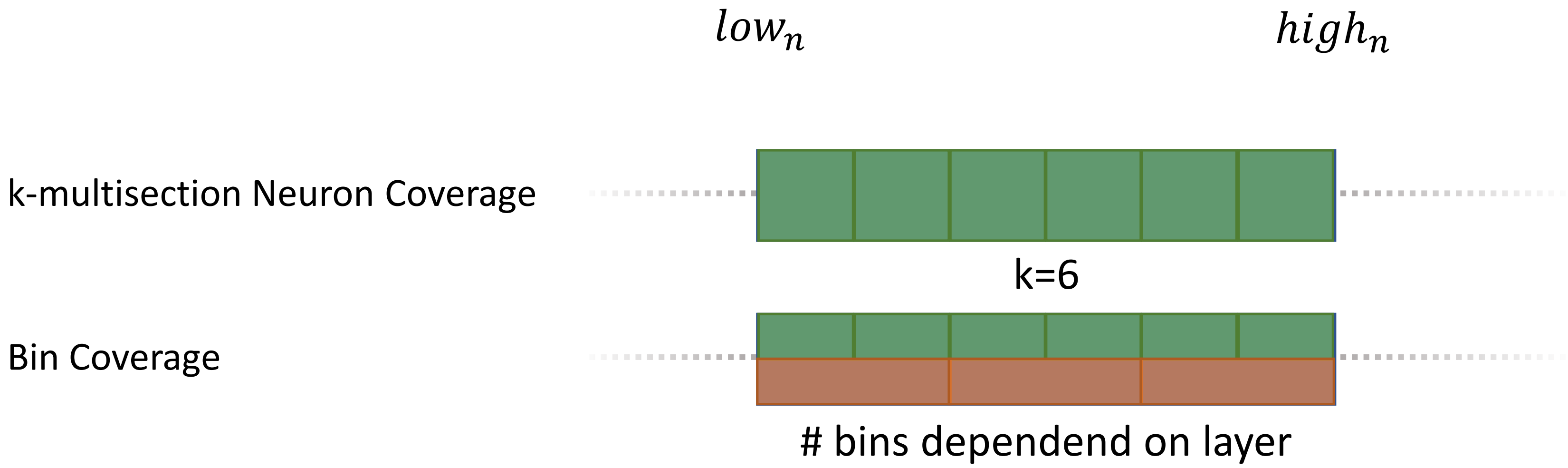


low_n
$high_n$
Neural Coverage
0.2
k-multisection Neuron Coverage
k=6
Neuron Boundary Coverage
Strong Neuron Activation Cov.
Bin Coverage
bins dependend on layer

α-Bin Coverage

Equally distribute so-called bins throughout layers.

Each layer contributes approximate same share to coverage metric.



α-Bin Coverage

Let L_i denote the number of neurons in Layer i.
Let L_{max} be the maximum of all L_i. Let $\alpha \in (0, \infty]$.

The minimum number of bins per layer for α-Bin Coverage is defined as:

$$Bins = L_{max} \cdot \alpha$$

The number of bins per neuron in Layer i is defined as:

$$k_i = \left\lceil \frac{Bins}{L_i} \right\rceil$$

Structure

1. Problem
2. Current DNN Test Coverage Metrics
3. α-Bin Coverage
4. Practical Evaluation

Practical Evaluation

The main questions:

1. Can α-Bin Coverage be implemented in a practically feasible way?
2. Can α-Bin Coverage be optimized with a greedy search approach?
3. How does α-Bin Coverage relate to other DNN coverage metrics?
4. Can α-Bin Coverage be used to find wrong behaviours?

Practical Evaluation

The main questions:

1. Can α-Bin Coverage be implemented in a practically feasible way?
2. Can α-Bin Coverage be optimized with a greedy search approach?
3. How does α-Bin Coverage relate to other DNN coverage metrics?
4. Can α-Bin Coverage be used to find wrong behaviours?

Practically feasible?

Test setup (1/2):

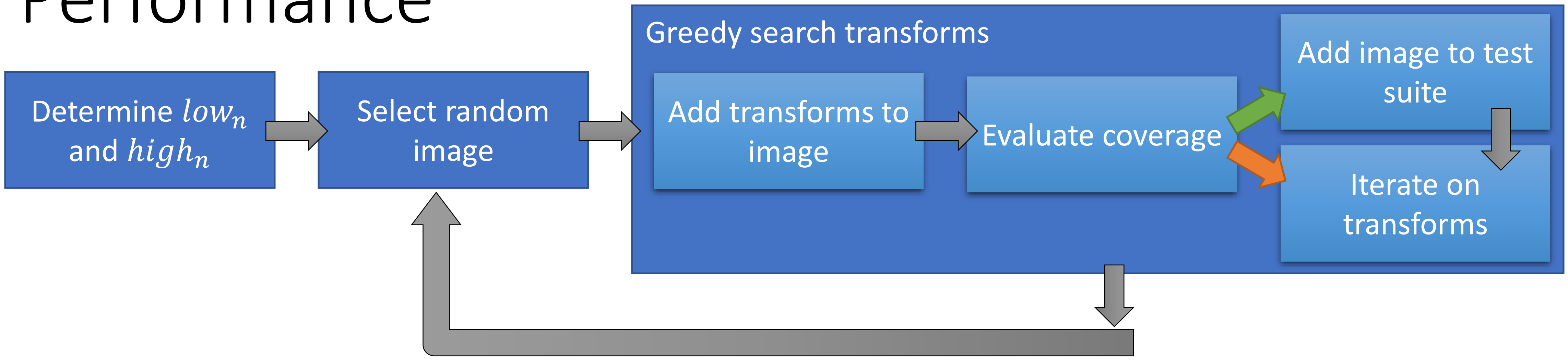
- 10 layer DNN inspired by Nvidea End to End approach using ReLu
- Trained on 45,500 publicly available labeled images
- Implemented in Python using Tensorflow

Practically feasible?

Test setup (2/2):

- Created greedy optimizer that uses image transforms to optimize coverage metric
- Compare behaviour of α-Bin Coverage & Neuron Coverage

Performance



Determining low_n and $high_n$ only needs to be done once and can be approximated through random sampling.

Calculating α-Bin Coverage incrementally:
constant time (dependend on network size).

Greedy search: Transforms

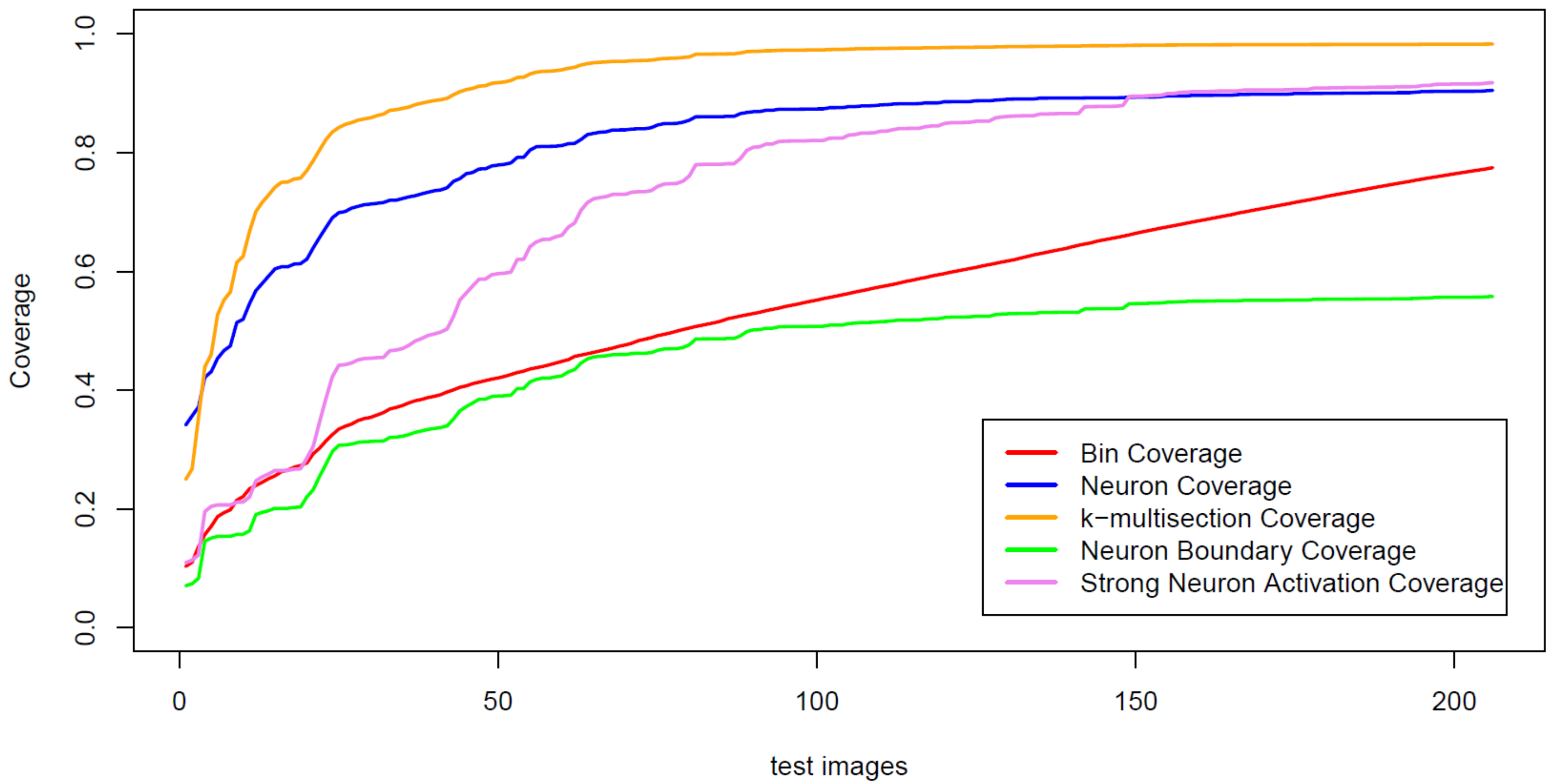
Transformations: Translation, Brightness, Contrast, Blur

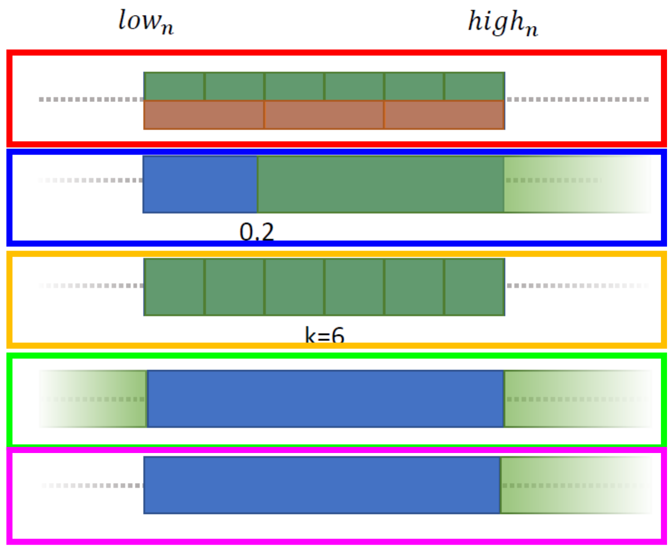
Practical Evaluation

The main questions:

1. Can α-Bin Coverage be implemented in a practically feasible way?
2. Can α-Bin Coverage be optimized with a greedy search approach?
3. How does α-Bin Coverage relate to other DNN coverage metrics?
4. Can α-Bin Coverage be used to find wrong behaviours?

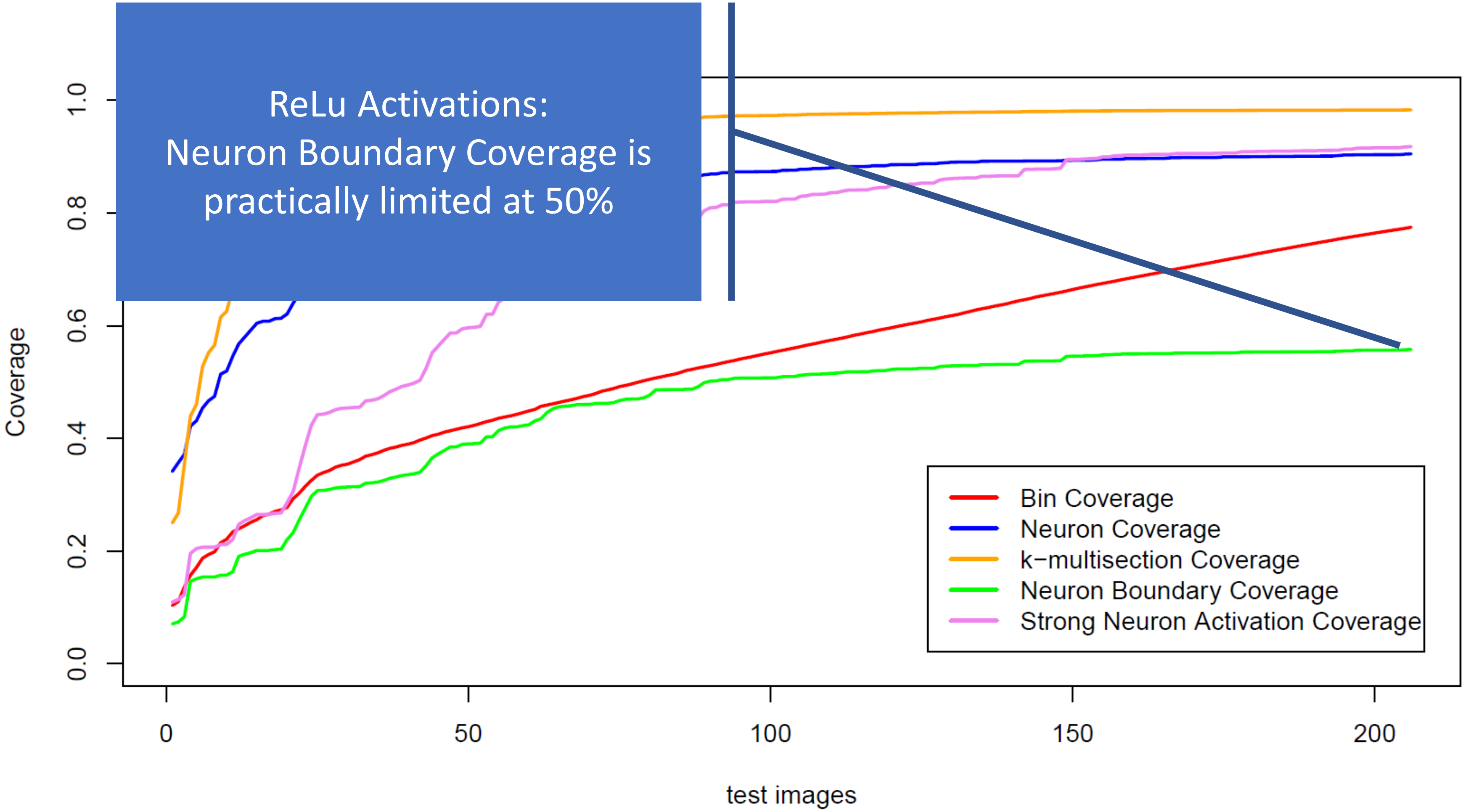
Greedy Optimization: Bin Coverage

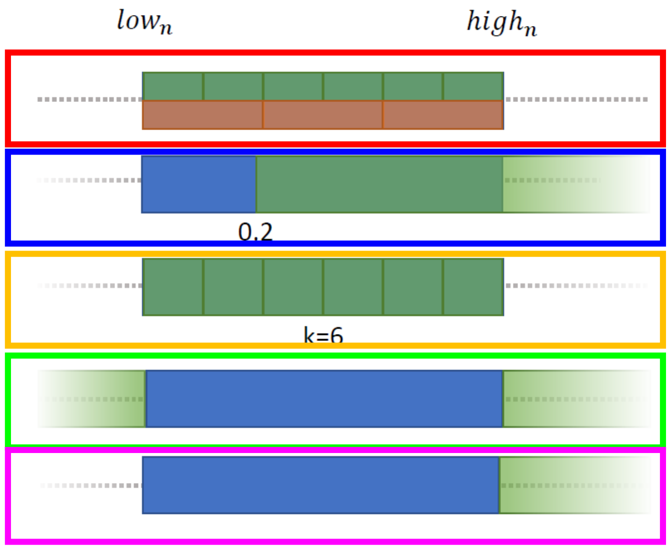




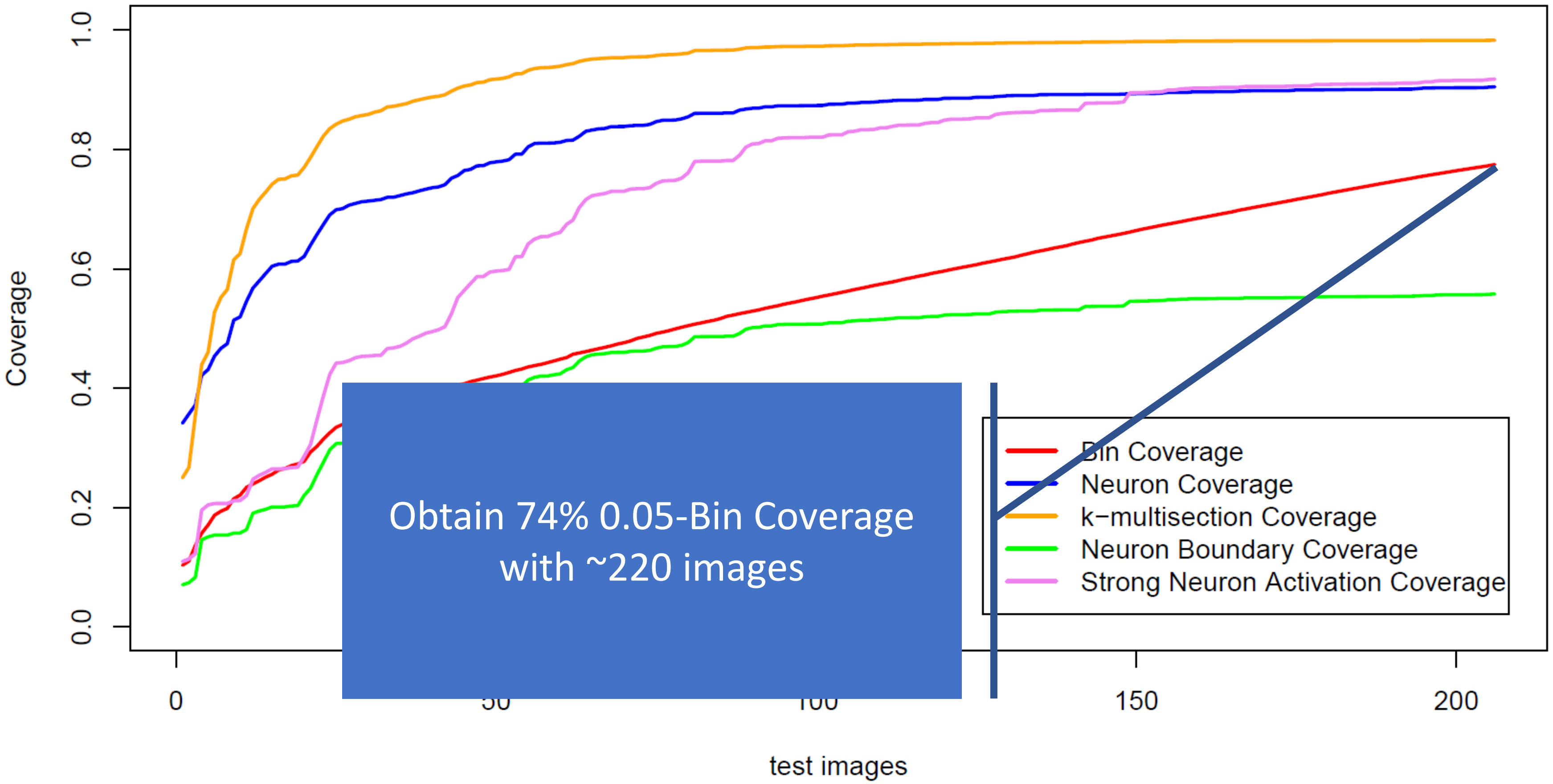
Greedy Optimization: Bin Coverage

ReLu Activations:
Neuron Boundary Coverage is practically limited at 50%

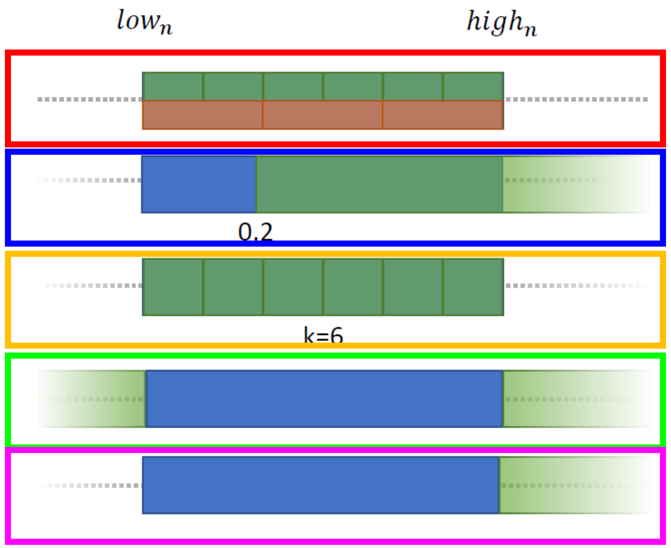




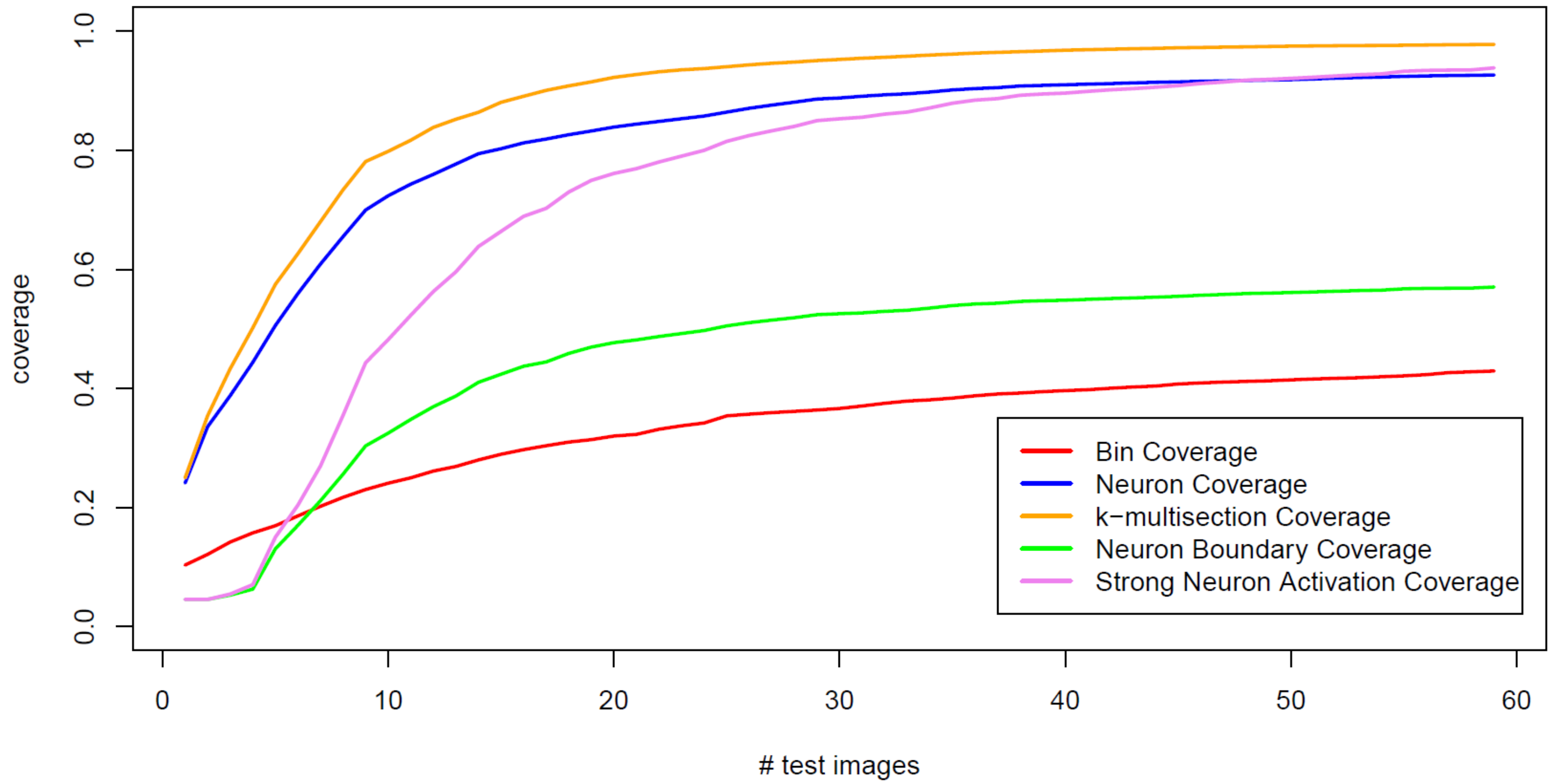
Greedy Optimization: Bin Coverage

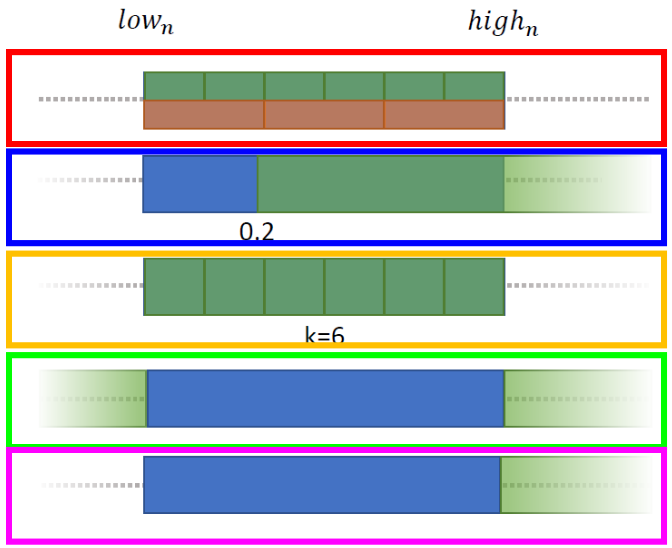


Obtain 74% 0.05-Bin Coverage with ~220 images

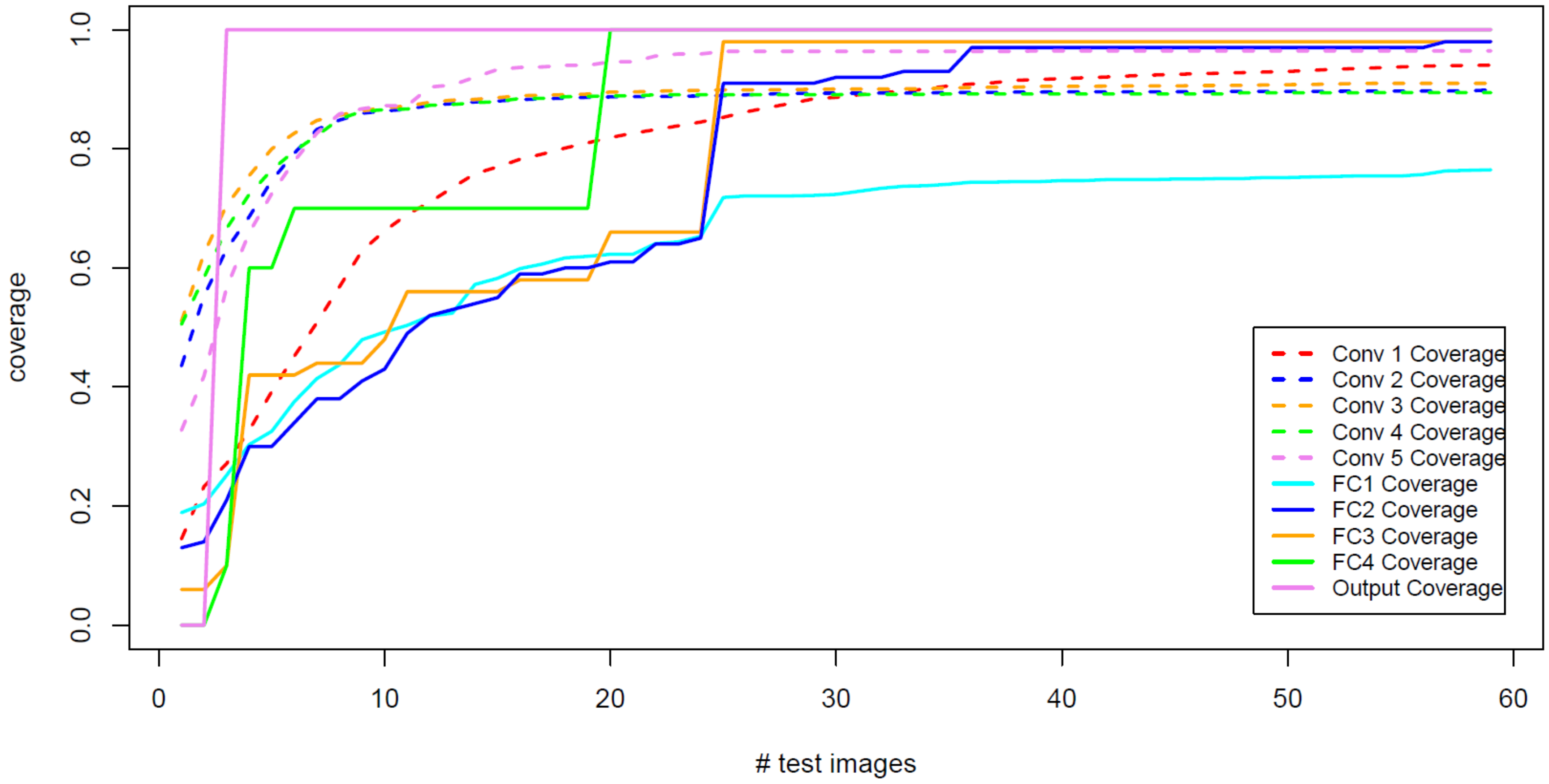


Greedy Optimization: Neuron Coverage

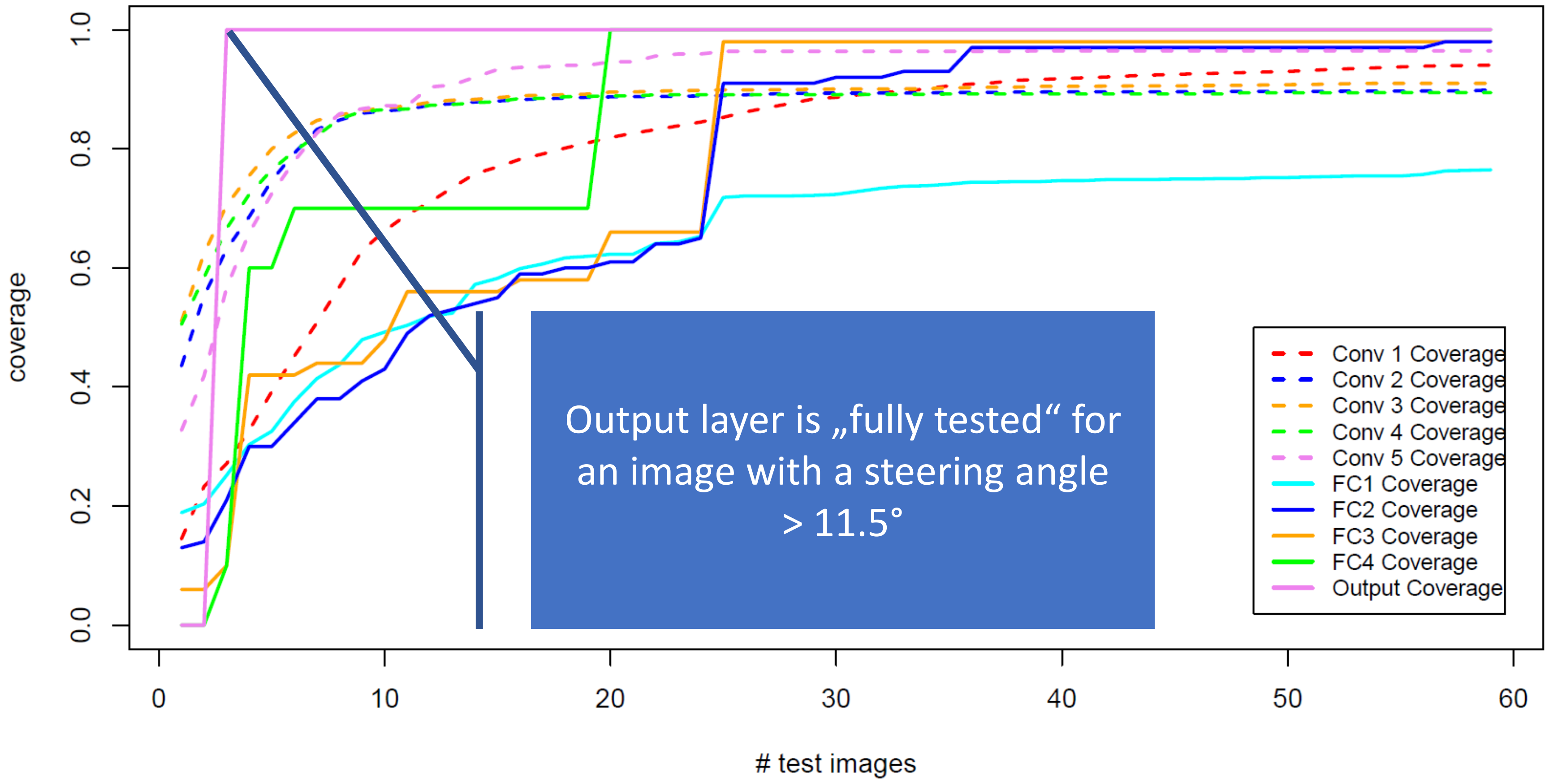




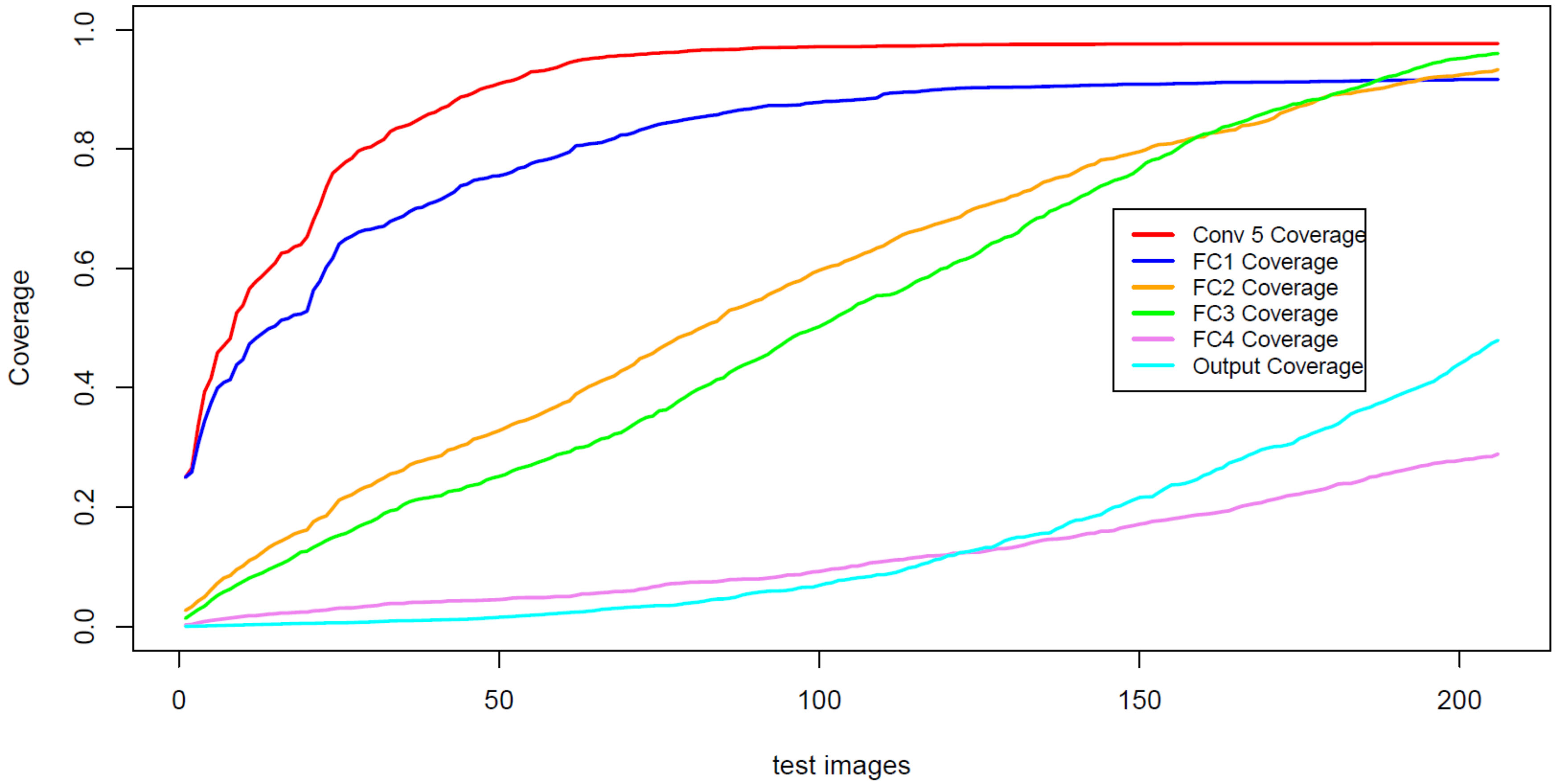
Neuron Coverage Optimization: Layer View



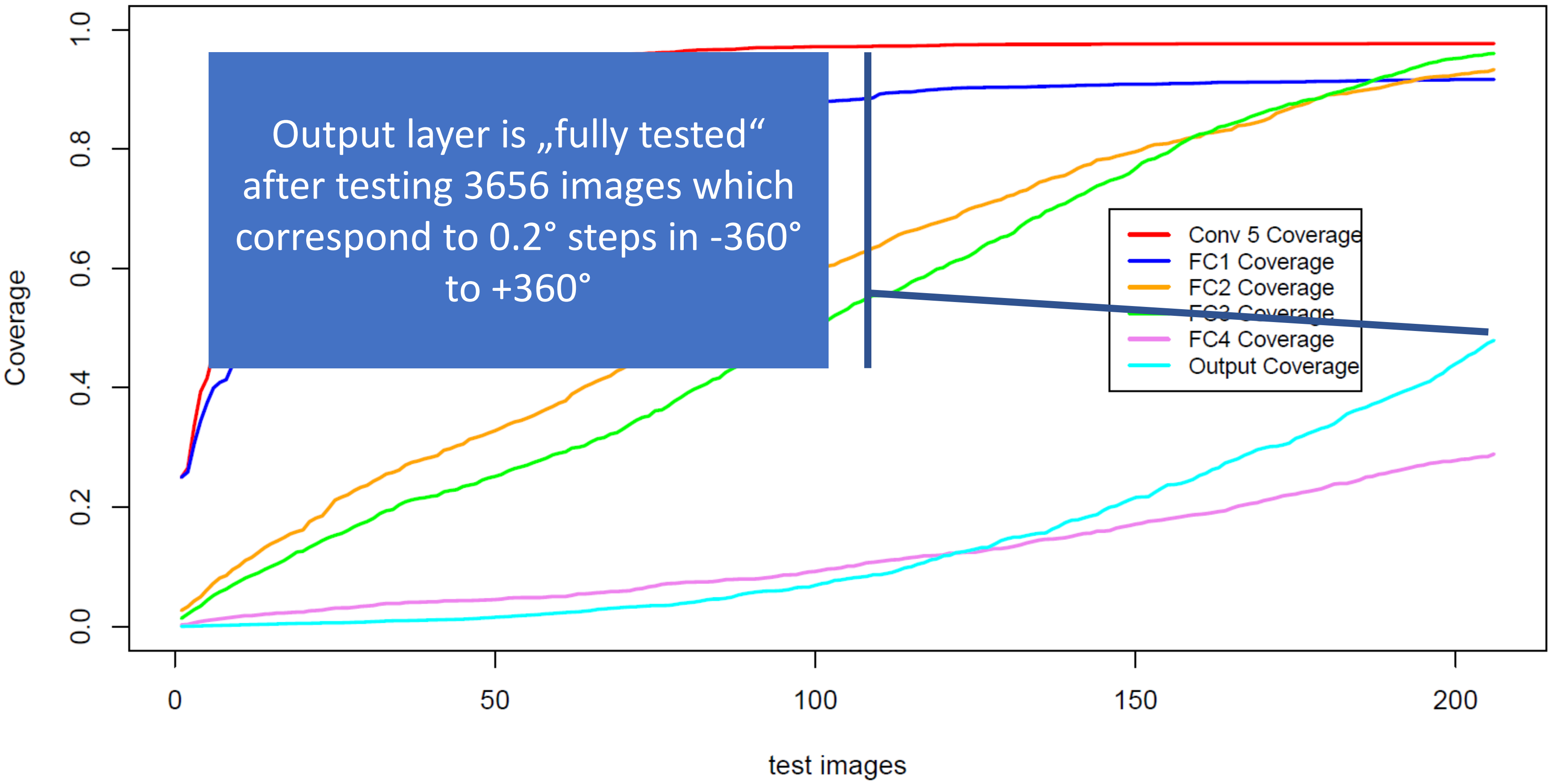
Neuron Coverage Optimization: Layer View



Bin Coverage Optimization: Layer View



Bin Coverage Optimization Layer View



Practical Evaluation

The main questions:

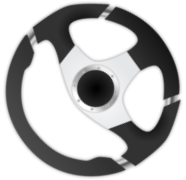
1. Can α-Bin Coverage be implemented in a practically feasible way?
2. Can α-Bin Coverage be optimized with a greedy search approach?
3. How does α-Bin Coverage relate to other DNN coverage metrics?
4. Can α-Bin Coverage be used to find wrong behaviours?

Deviation from target labels in test suite

Coverage metric	> 20 deg	total images	ratio
Neuron Coverage	2	100	2.0%
Bin Coverage	24	247	9.7%

Example:

Transformed Image



Output: 234°

Target: 160°

Conclusions

- Current DNN test coverage metrics deal all neurons equally
- This introduces an intrinsic focus on the neurons of low layers in modern architectures
- α-Bin Coverage is a practically feasible approach to equally distribute a test coverage metric over all layers
- First evidence shows that α-Bin Coverage can be used for finding erroneous behaviours and creating test suites automatically

Let‘s discuss!

Some points to consider:

- Only one model in evaluation
- Limited number of test runs
- Only one domain
- Why greedy search?
- What is this strange α value? Why do we need it?
- How about classification tasks?

Greedy search

Stack transformations on randomly selected images to optimize coverage metric.

Add an image to test suite if it significantly increases coverage metric

<u>Transformations</u>: Translation, Brightness, Contrast, Blur