Purdue University
Smart Informatix Laboratory

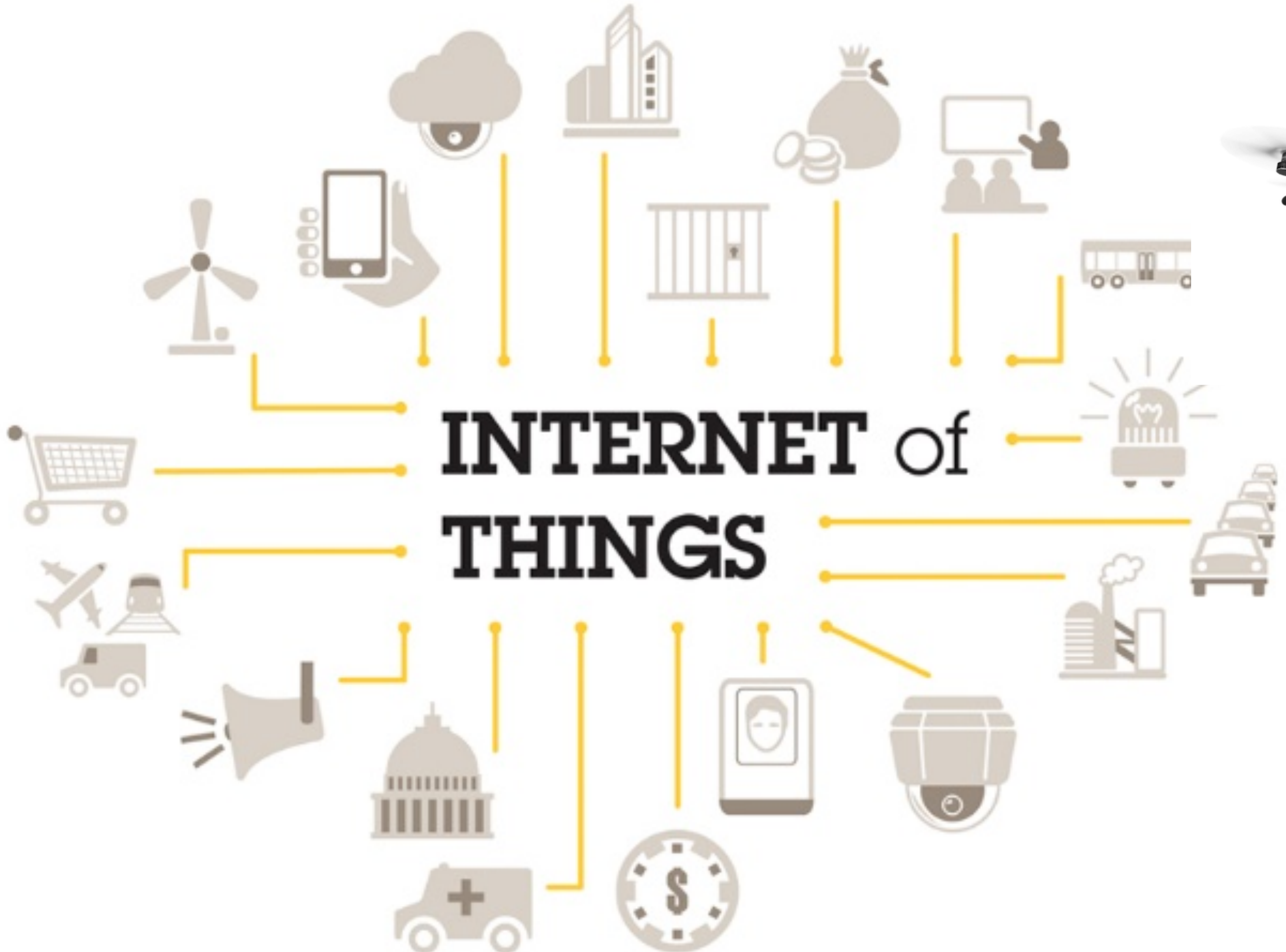
# Deep Neural Network Pruning for Efficient Edge Computing in IoT

**Rih-Teng Wu[1], Ankush Singla[2], Mohammad R. Jahanshahi[3], Elisa Bertino[4]**

[1] Ph.D. Student, Lyles School of Civil Engineering, Purdue University
[2] Ph.D. Student, Department of Computer Science, Purdue University
[3] Assistant Professor, Lyles School of Civil Engineering, Purdue University
[4] Professor, Department of Computer Science, Purdue University

March 20th, 2019

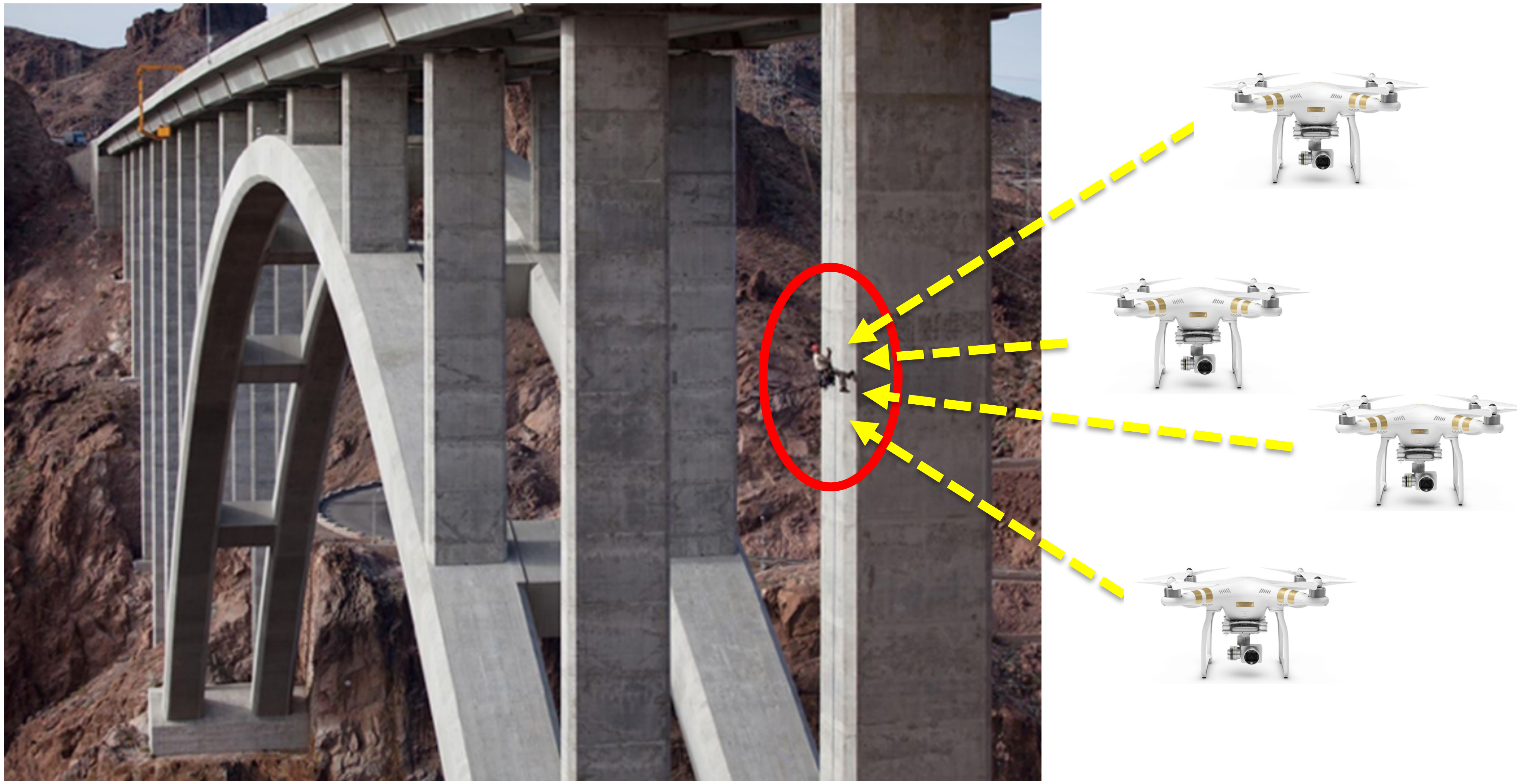
# Motivation – Internet of Things



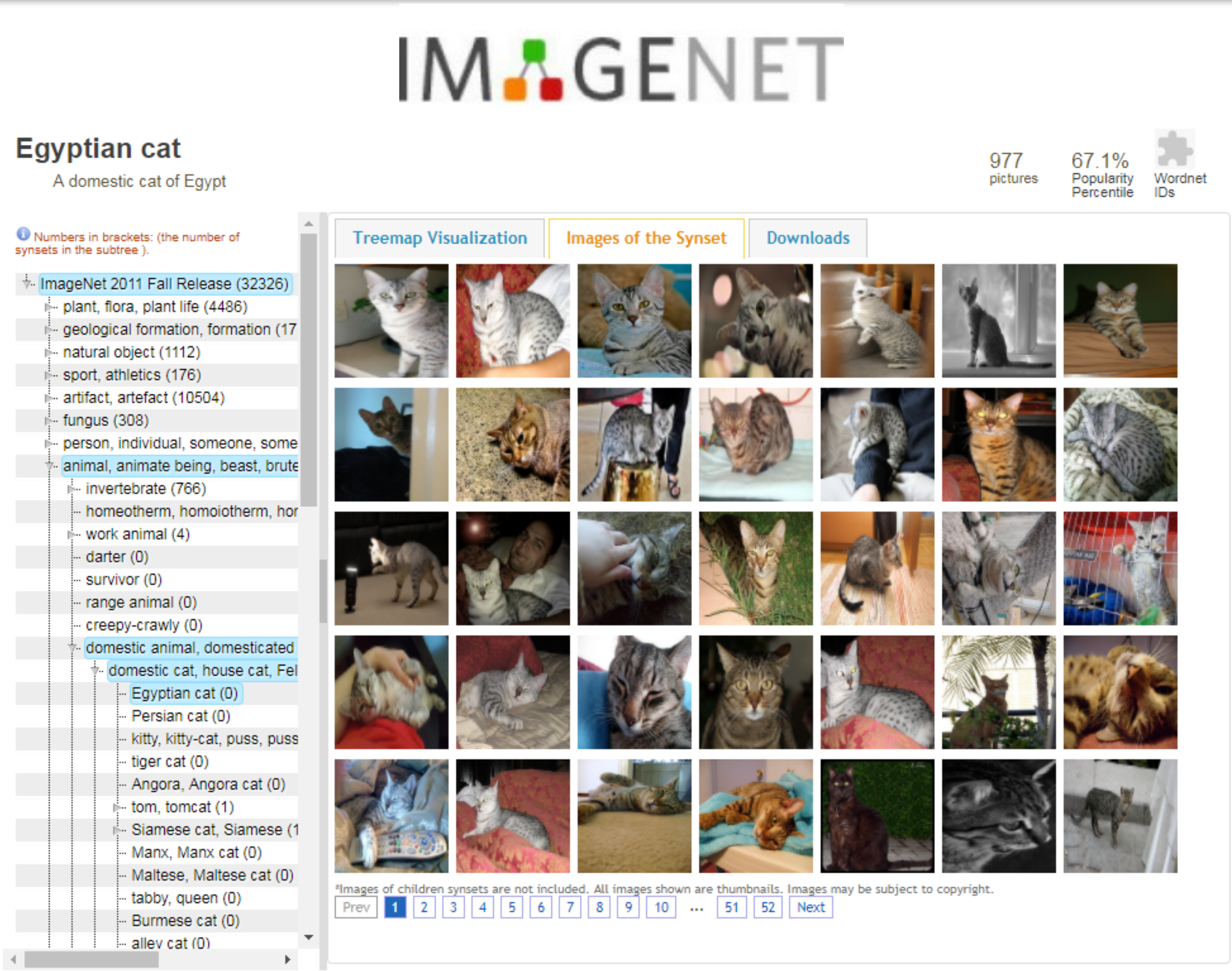

Source: https://tinyurl.com/yagpsakm

Figure adopted from: https://www.axis.com/blog/secure-insights/internet-of-things-reshaping-security/

# Motivation – Current Inspection in SHM



# Motivation – Deep Neural Networks




## Deep Convolutional Neural Network for SHM

- Specialized Architecture?
  - Needs a lot of data
- Transfer Learning?
  - Not efficient for edge computing

# Network Pruning – Inspiration from Biology

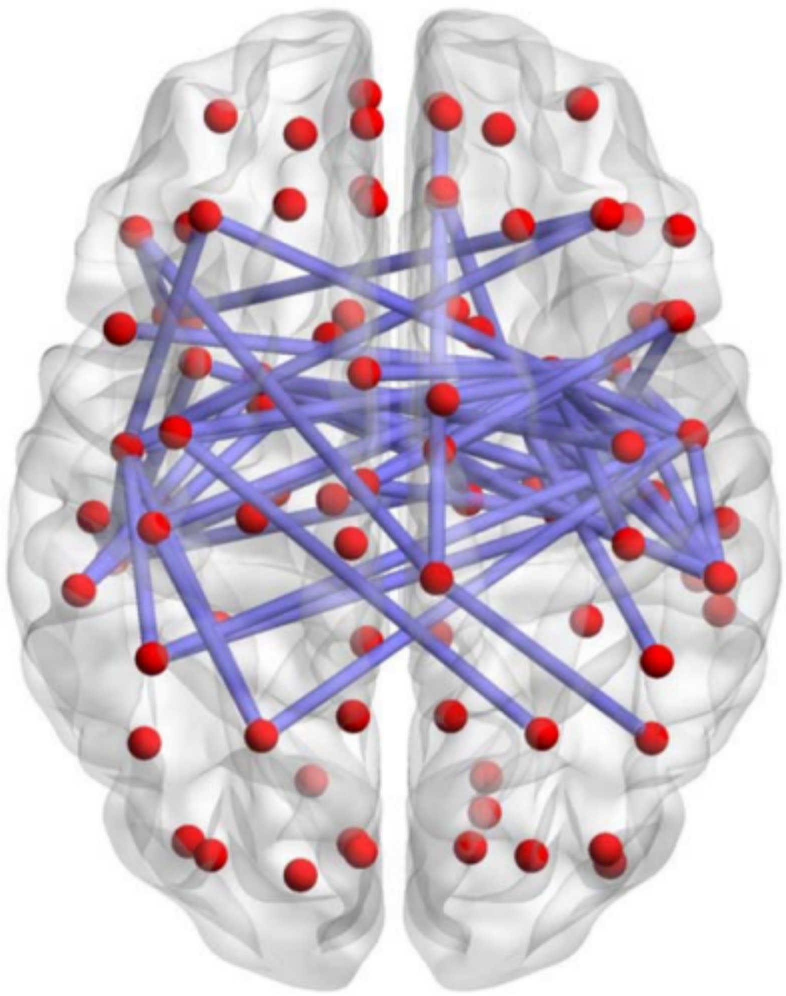


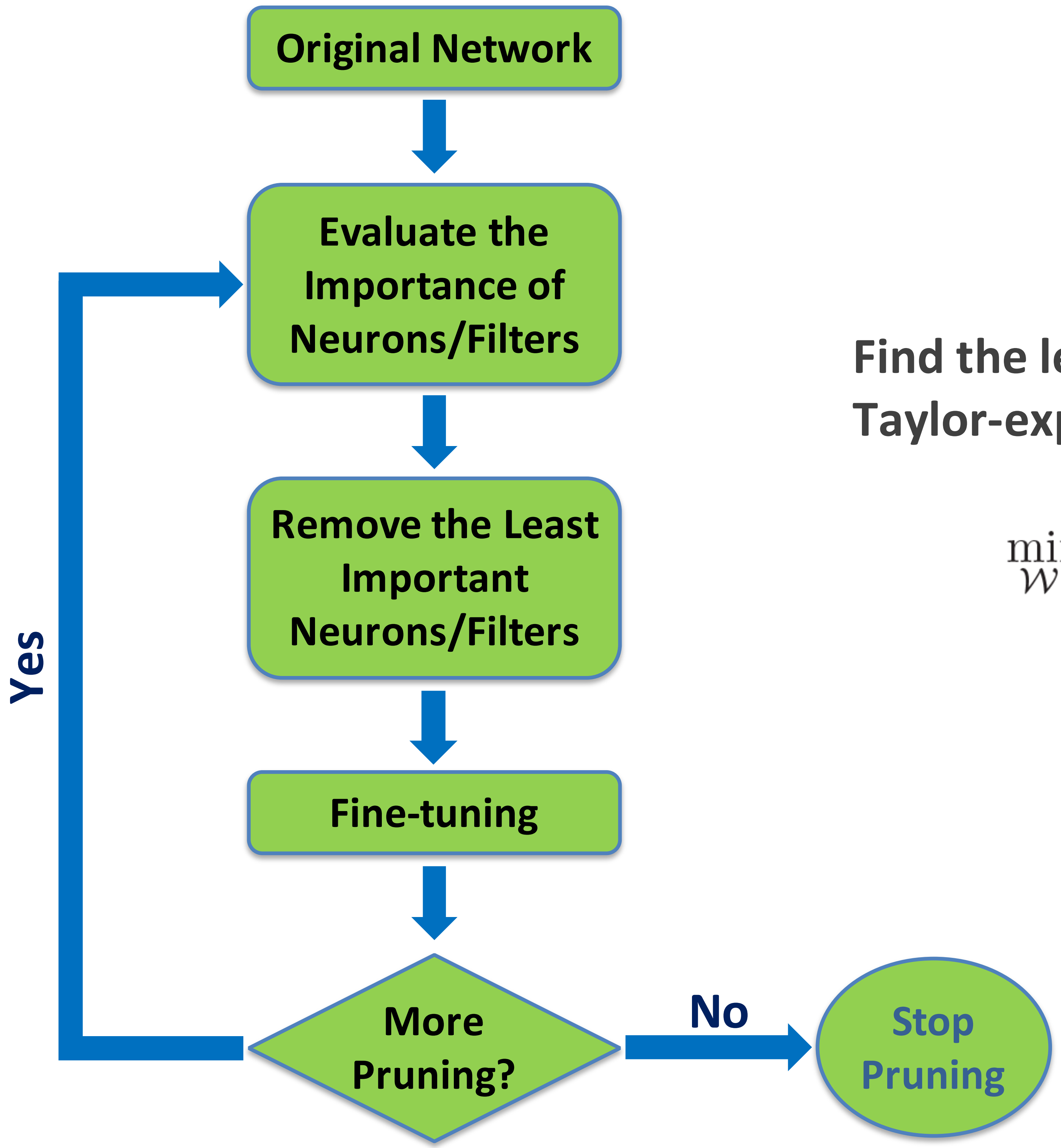
Figure adopted from Hong et al. (2013), " Decreased Functional Brain Connectivity in Adolescents with Internet Addiction."

# Existing Pruning Algorithms

- **Magnitudes of filter weights**
- **Magnitudes of activation values**
- **Mutual information between activations and predictions**
- **Regularization-based approaches**
- **Taylor-expansion based approach**

Molchanov et al. (2017), “Pruning Convolutional Neural Networks for Resource Efficient Inference”, arXiv:1611.06440v2.

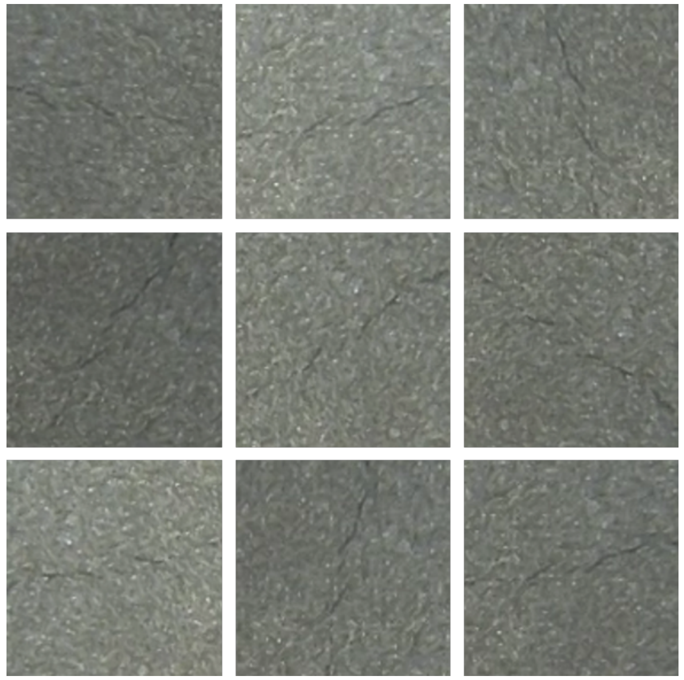
# Network Pruning with Filter Importance Ranking



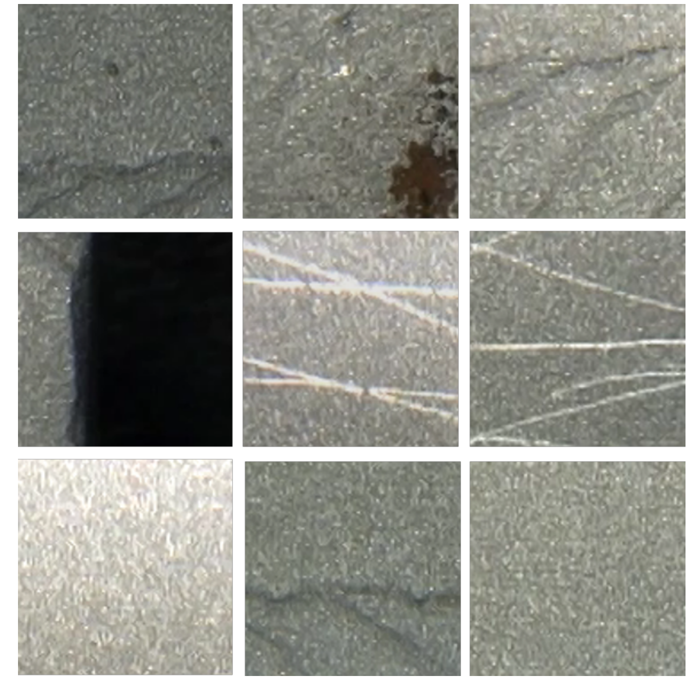

Find the least important filters based on Taylor-expansion (*Molchanov et al., 2017*)

$$\min_{\mathcal{W}'} \left| \mathcal{C}(\mathcal{D}|\mathcal{W}') - \mathcal{C}(\mathcal{D}|\mathcal{W}) \right|$$

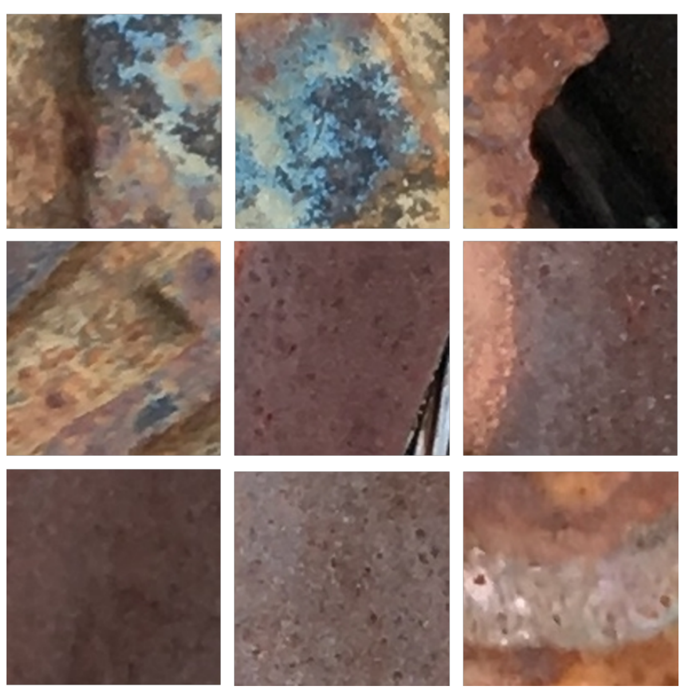
# Crack and Corrosion Datasets



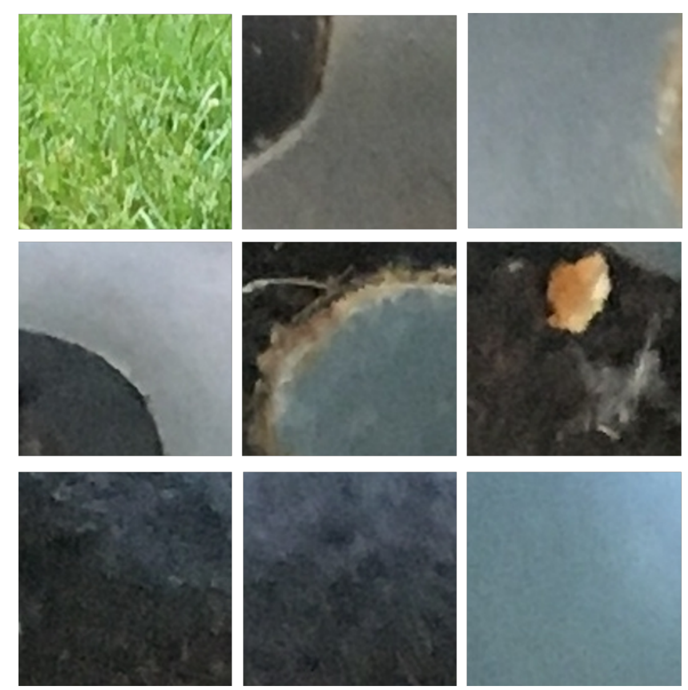
Crack (training: 25048, testing: 4420 )



Non-crack (training: 25313, testing: 4467 )



Corrosion (training: 28,083, testing: 4,956 )



Non-corrosion (training: 29,026, testing: 5,122 )

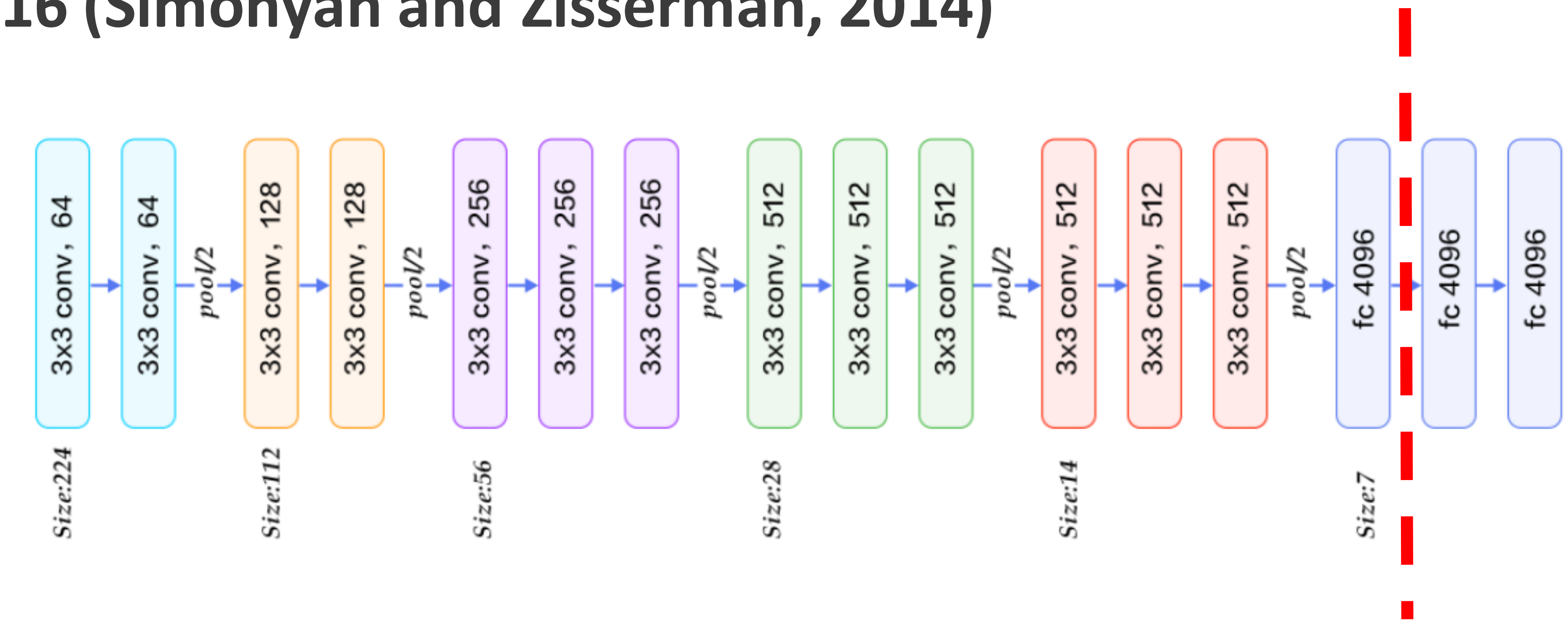
# Computing Units


Server device


Edge device

# Result – Transfer Learning without Pruning

➢ **VGG16 (Simonyan and Zisserman, 2014)**

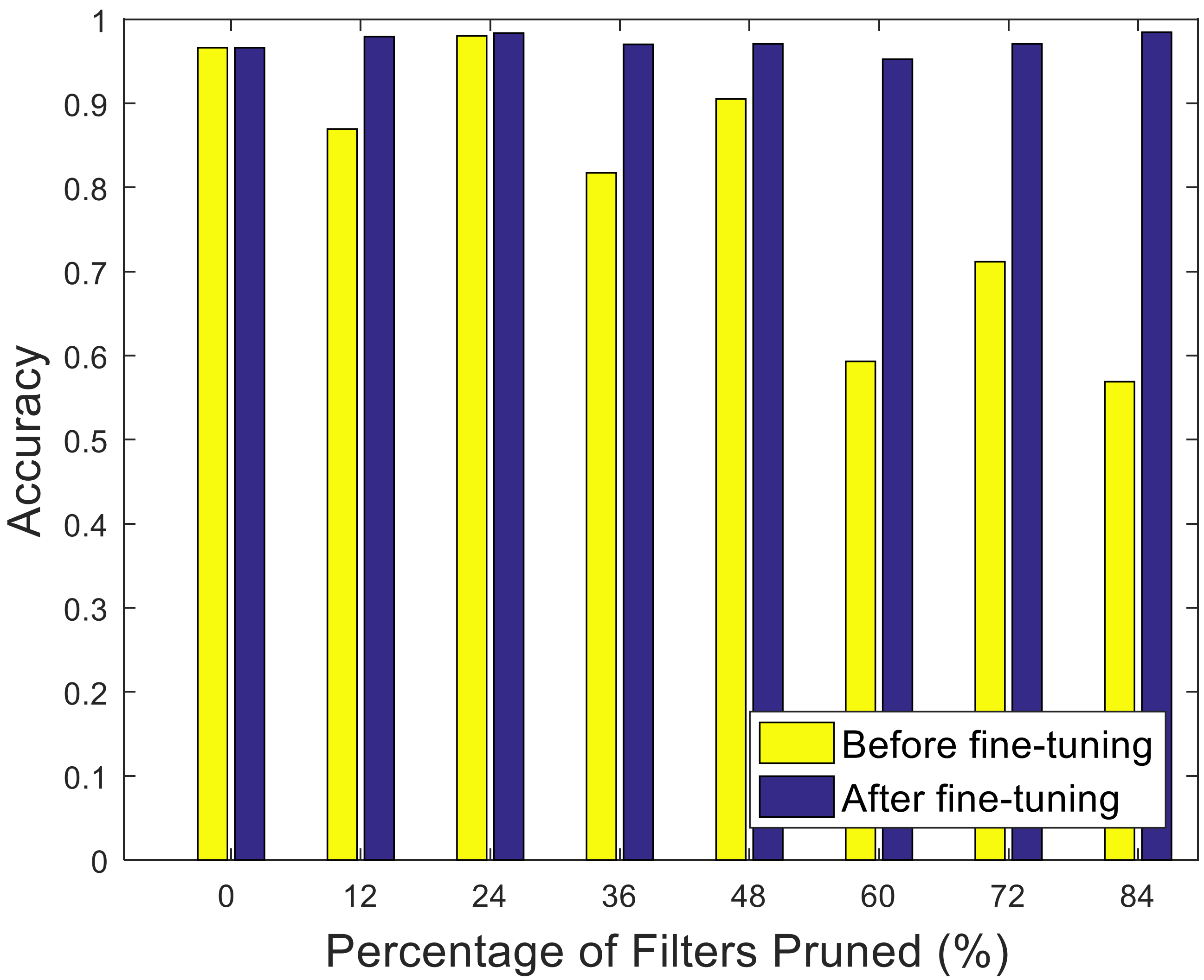



| Classifier | Model size (MB) | Inference time on Server (sec) | Inference time on Edge (sec) | Accuracy |
|---|---|---|---|---|
| KNN | 3277.000 | 96.09 | 587.58 | 0.9460 |
| SVC | 163.000 | 124.59 | 417.65 | 0.8928 |
| SVMH | 0.032 | 29.47 | $\boxed{234.84}$ | 0.8553 |

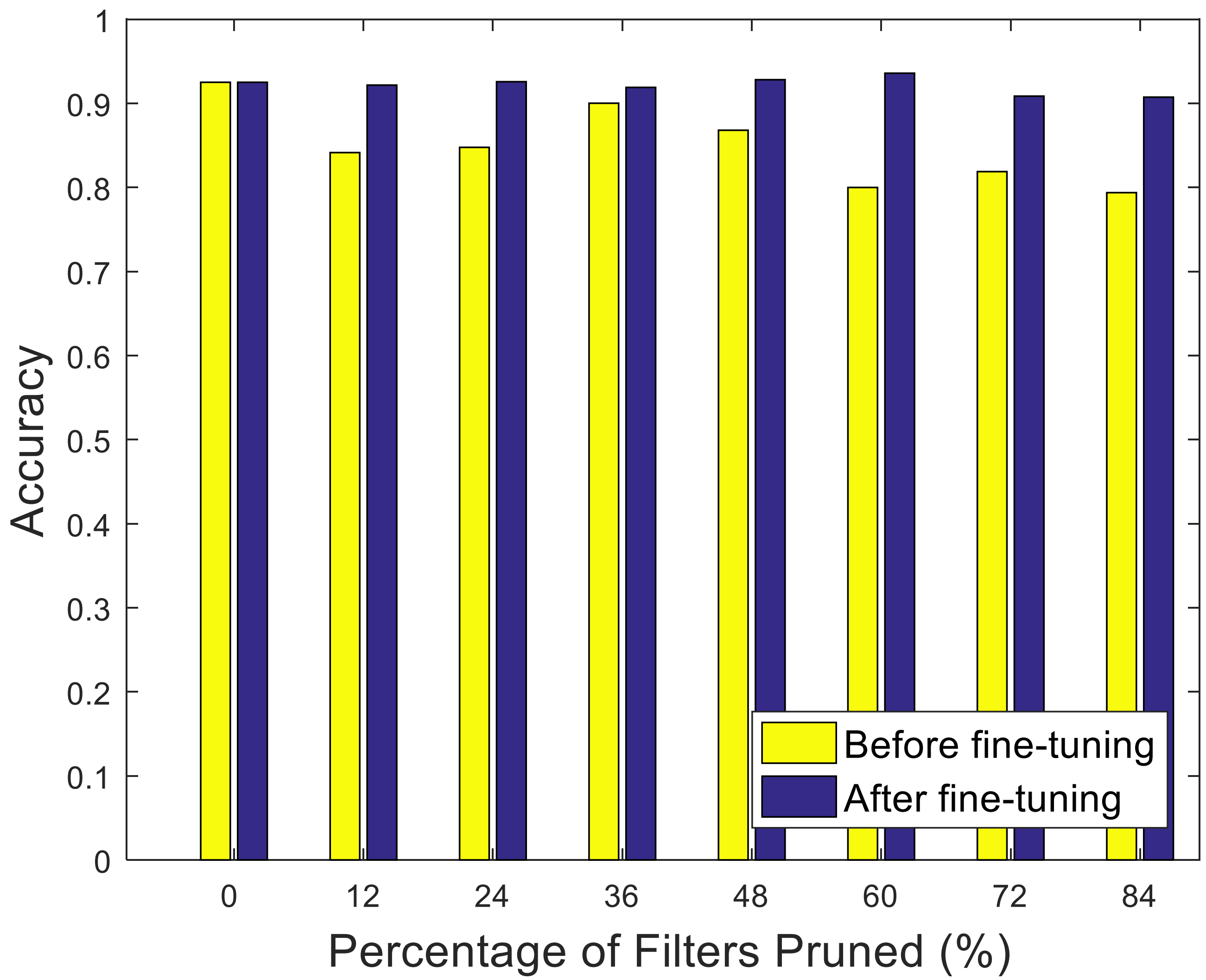
*Inference time: the total time required to classify 3,720 image patches of size 224x224.

Simonyan and Zisserman (2014), "Very Deep Convolutional Networks for Large-Scale Image Recognition", arXiv:1409.1556v6.

# Result – VGG16 with Pruning



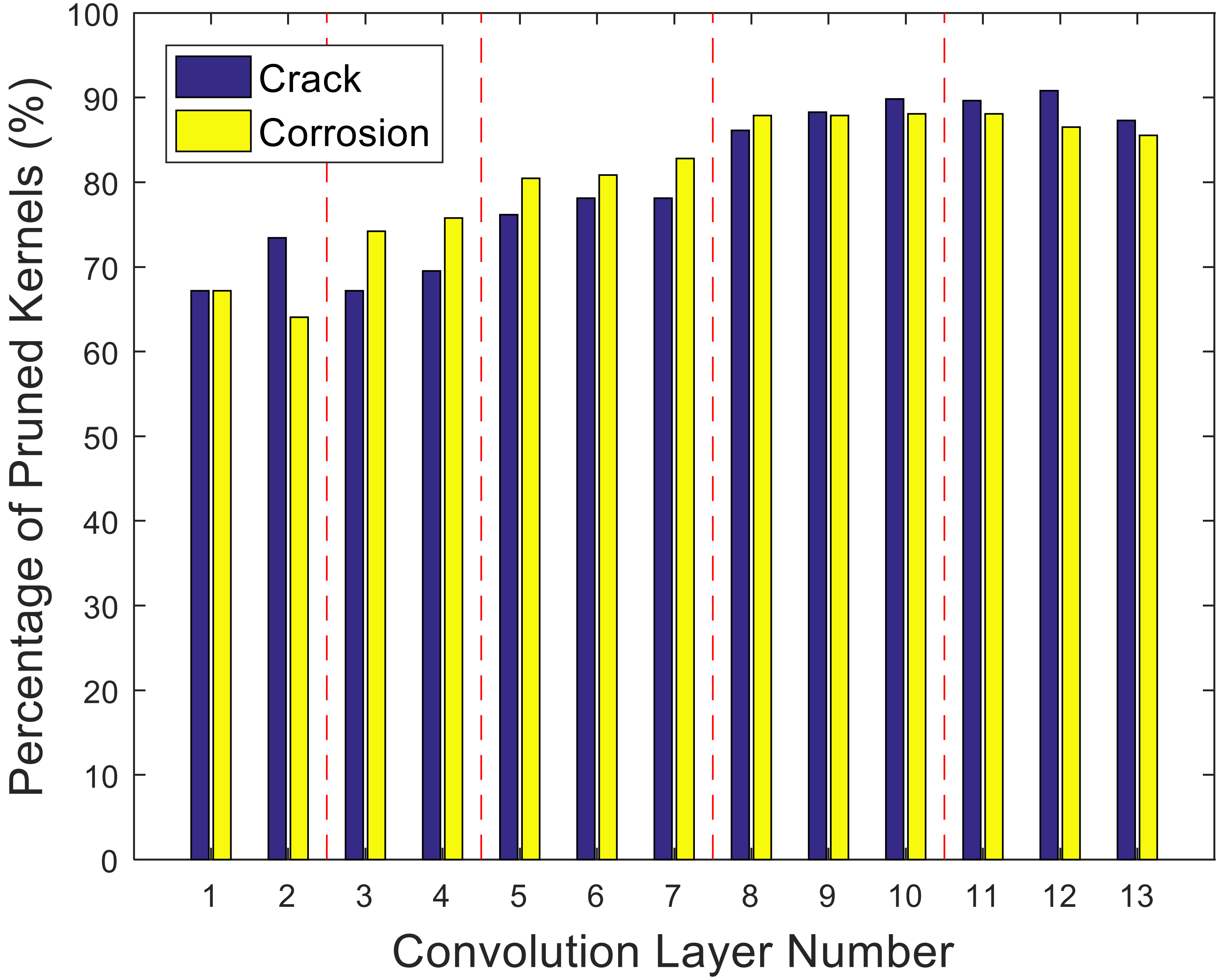

Crack




Corrosion

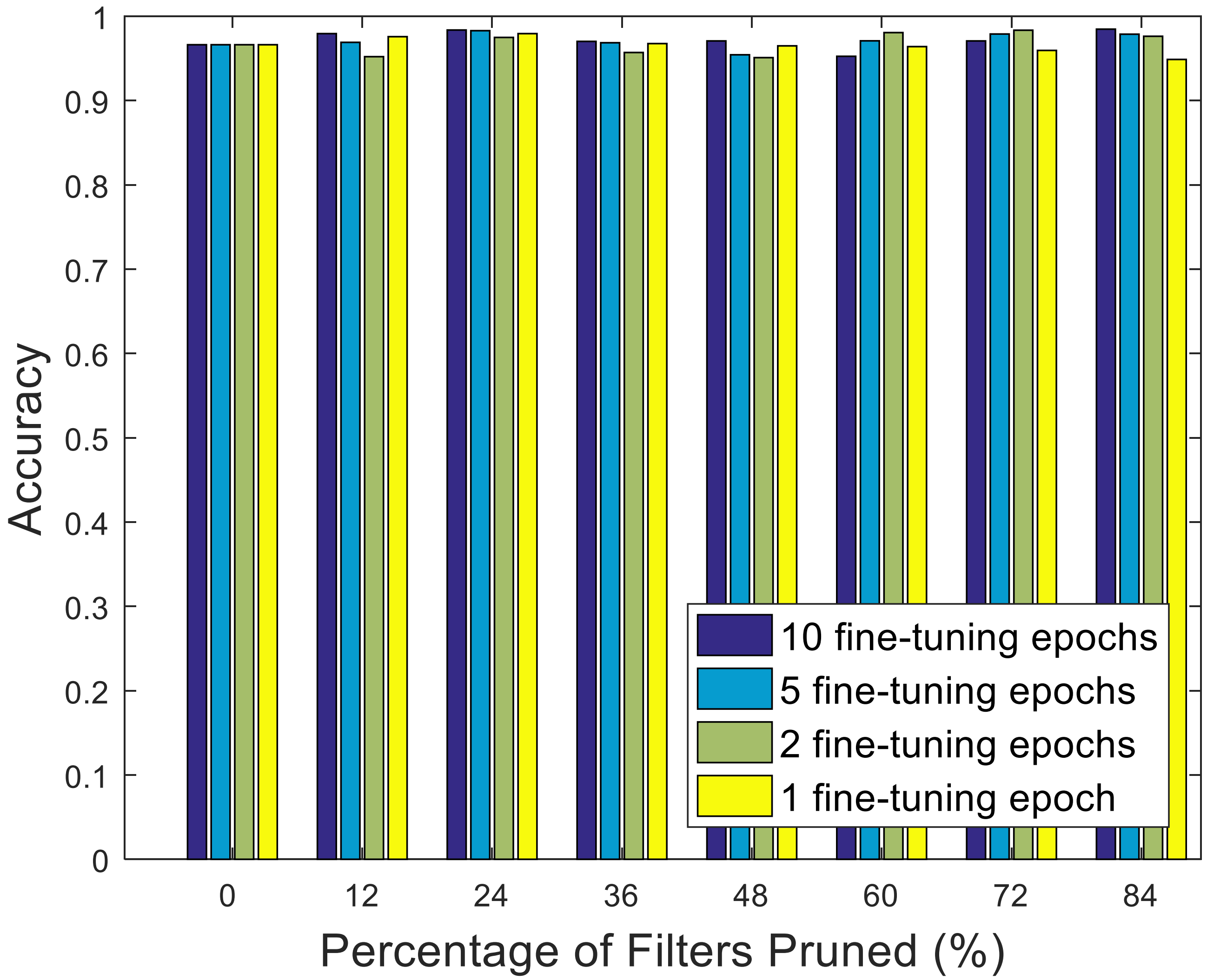
- **Pruning is conducted on the server device.**
- **Accuracy remains descent after pruning followed by fine-tuning.**

# Distribution of Pruned Convolution Kernels

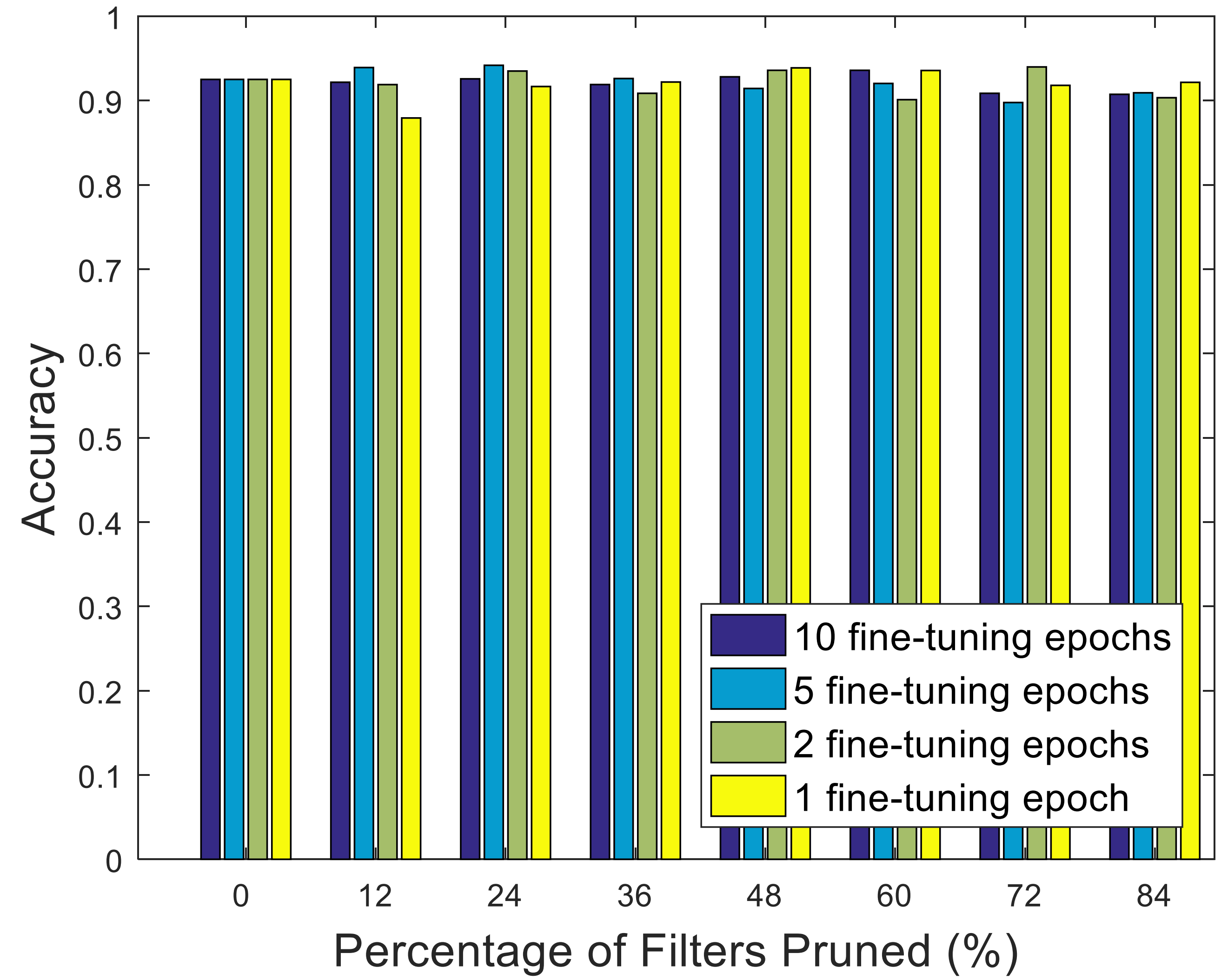



- Early layers are pruned less, indicating the importance of low-level features.
- Similar numbers of pruned kernels in layers between the pooling layers are observed.

# Sensitivity Analysis – Number of Fine-tuning Epochs



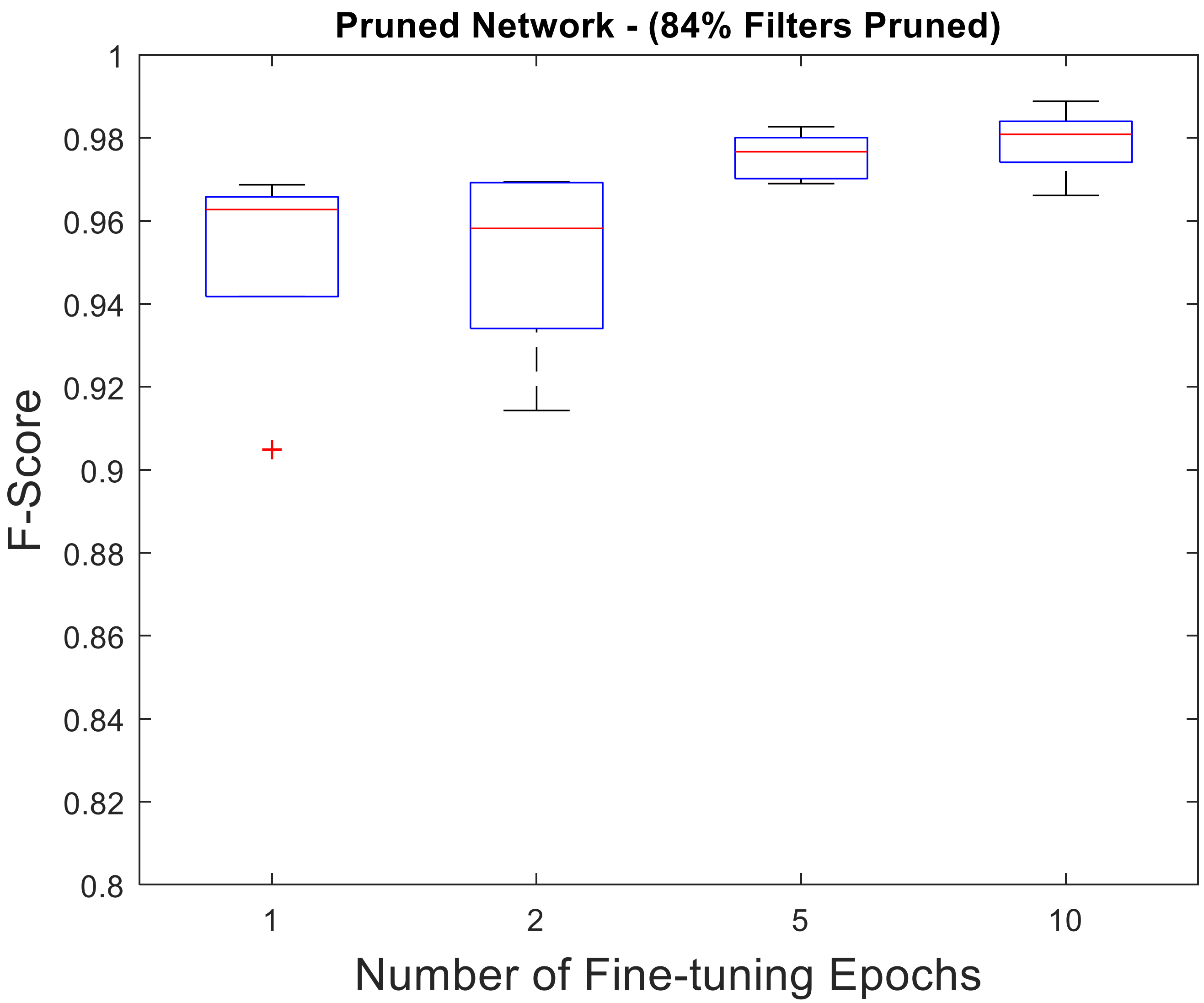

Crack



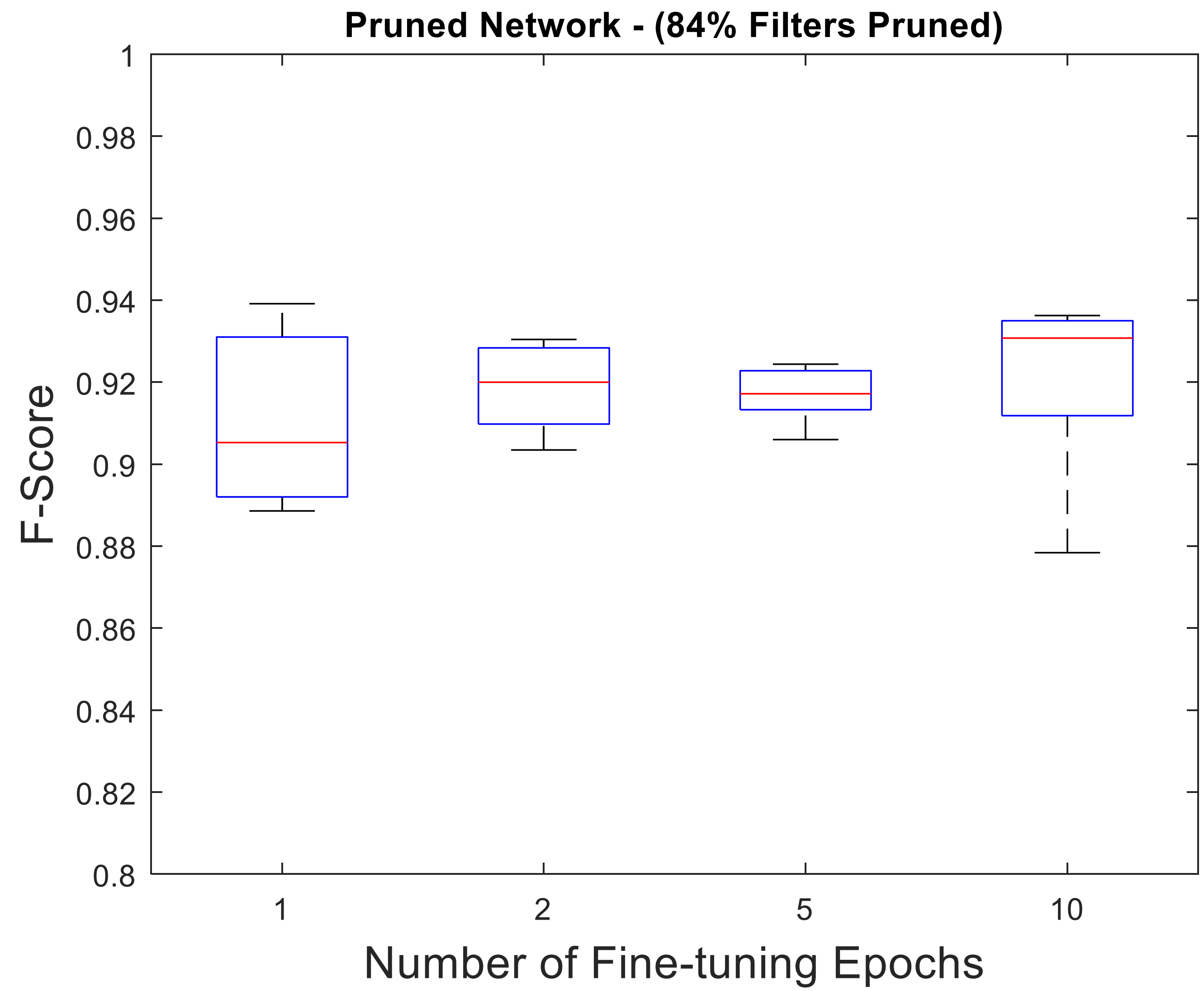

Corrosion

- **The accuracy is not sensitive to the number of fine-tuning epochs used in each pruning iteration.**

# Sensitivity Analysis – Number of Fine-tuning Epochs



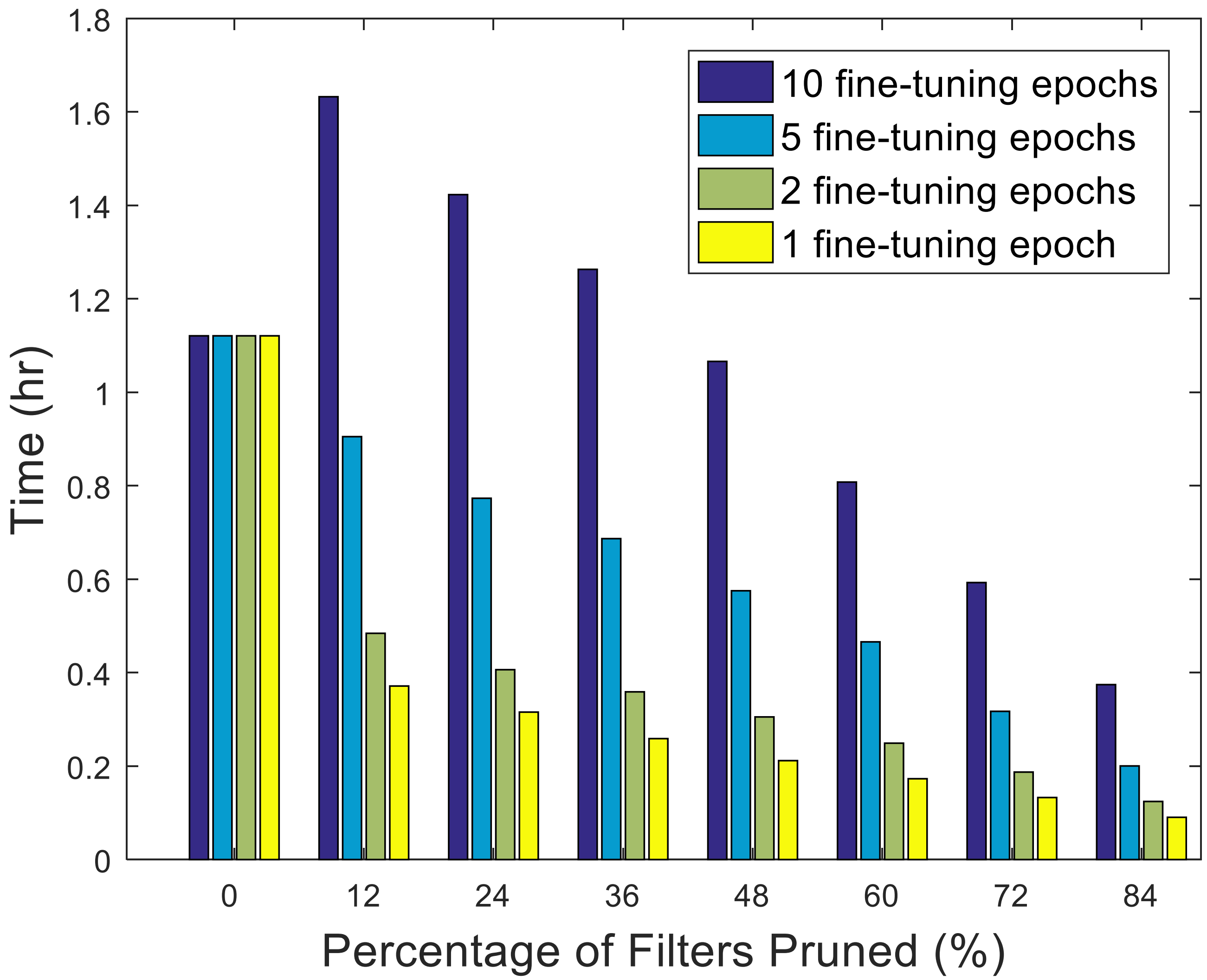

Crack



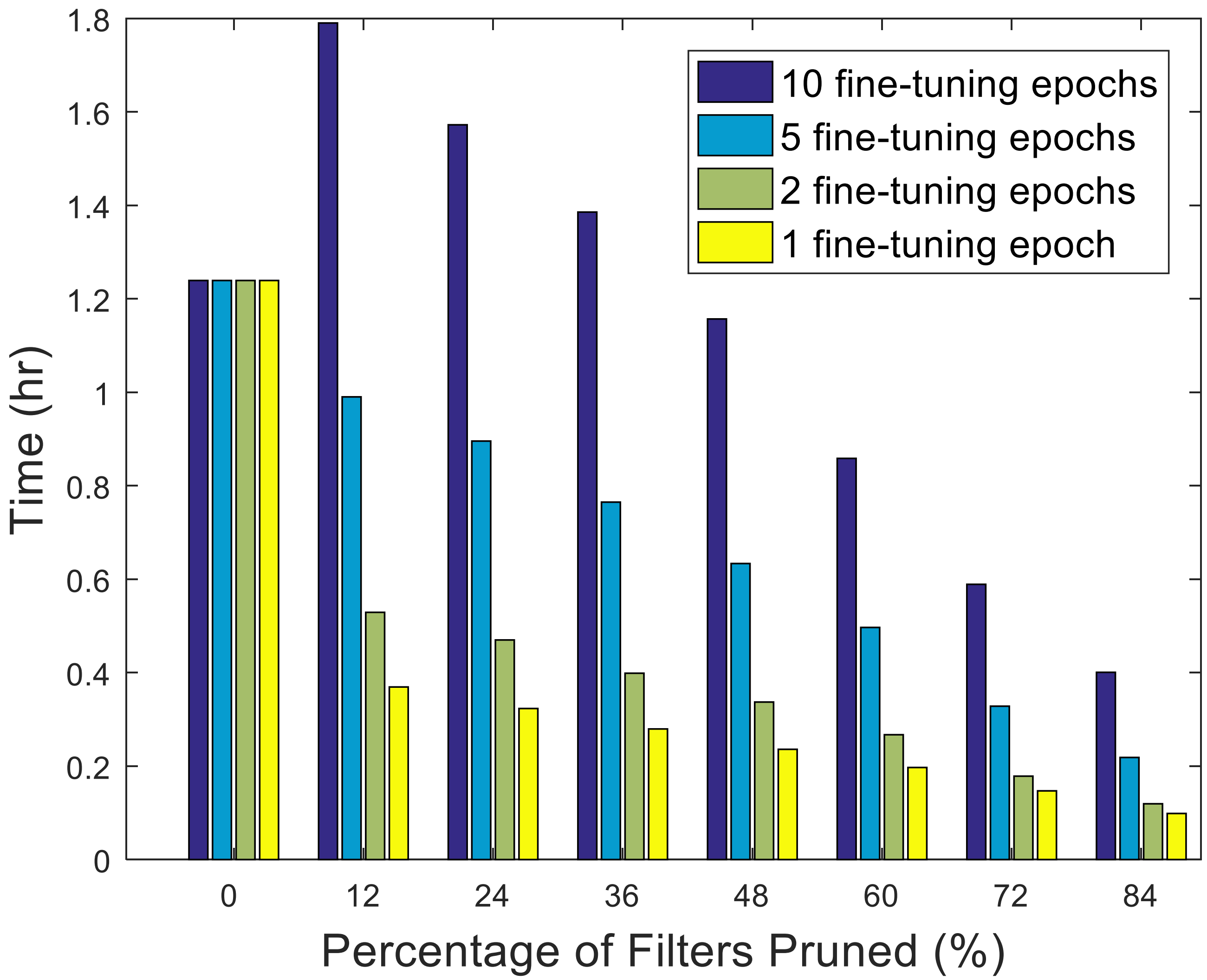

Corrosion

- **The accuracy is not sensitive to the number of fine-tuning epochs used in each pruning iteration.**

# Pruning Time Required on the Server



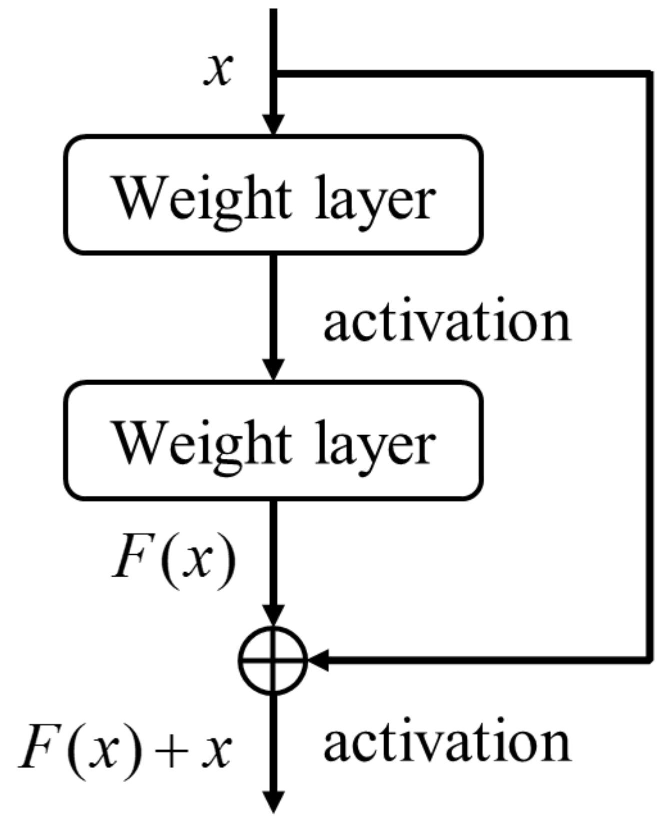

Crack



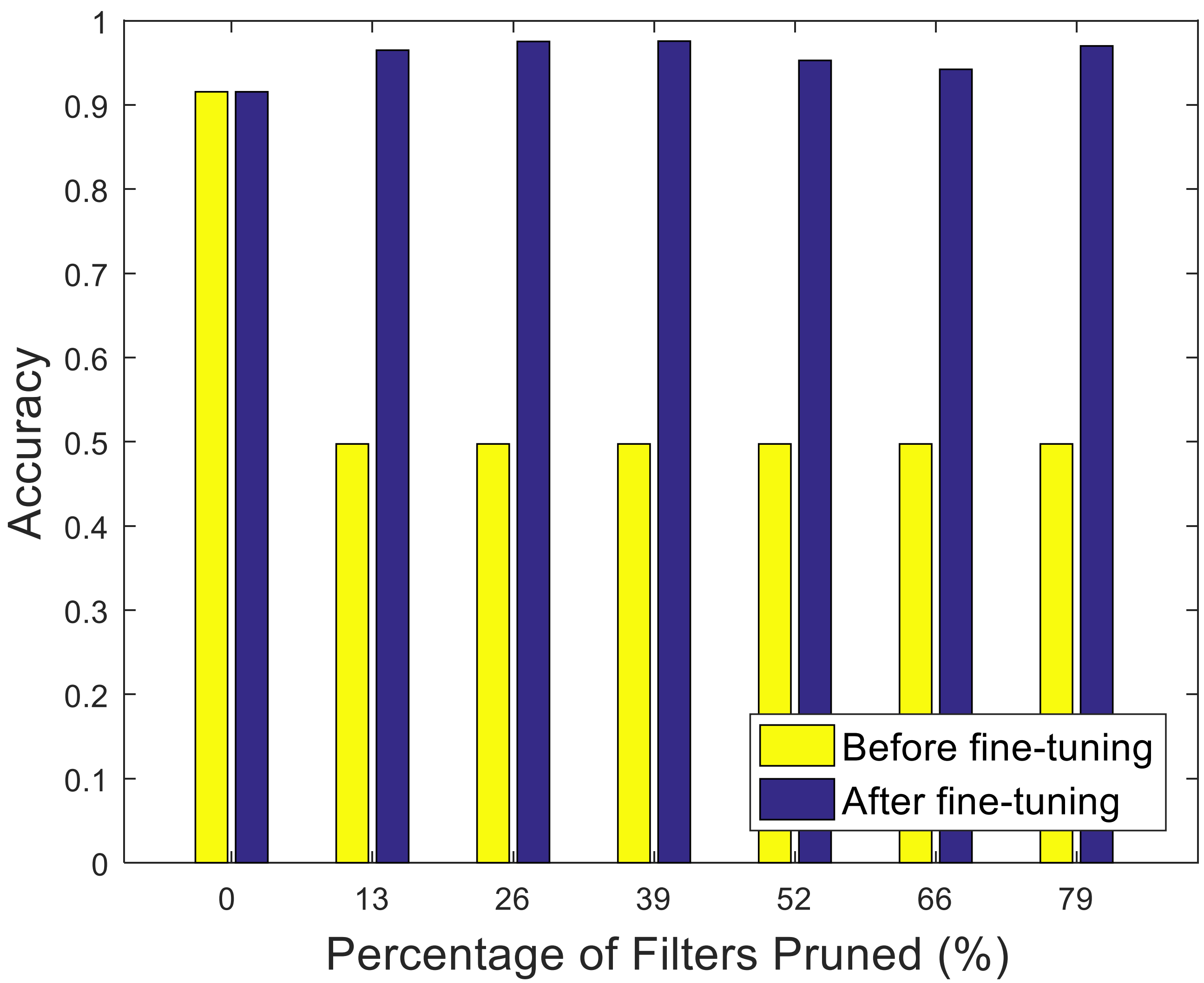

Corrosion

- **When using only 1 fine-tuning epoch, the total pruning time is reduced to 1.5(hr), which is approximately 4.6 times faster than using 10 fine-tuning epochs.**

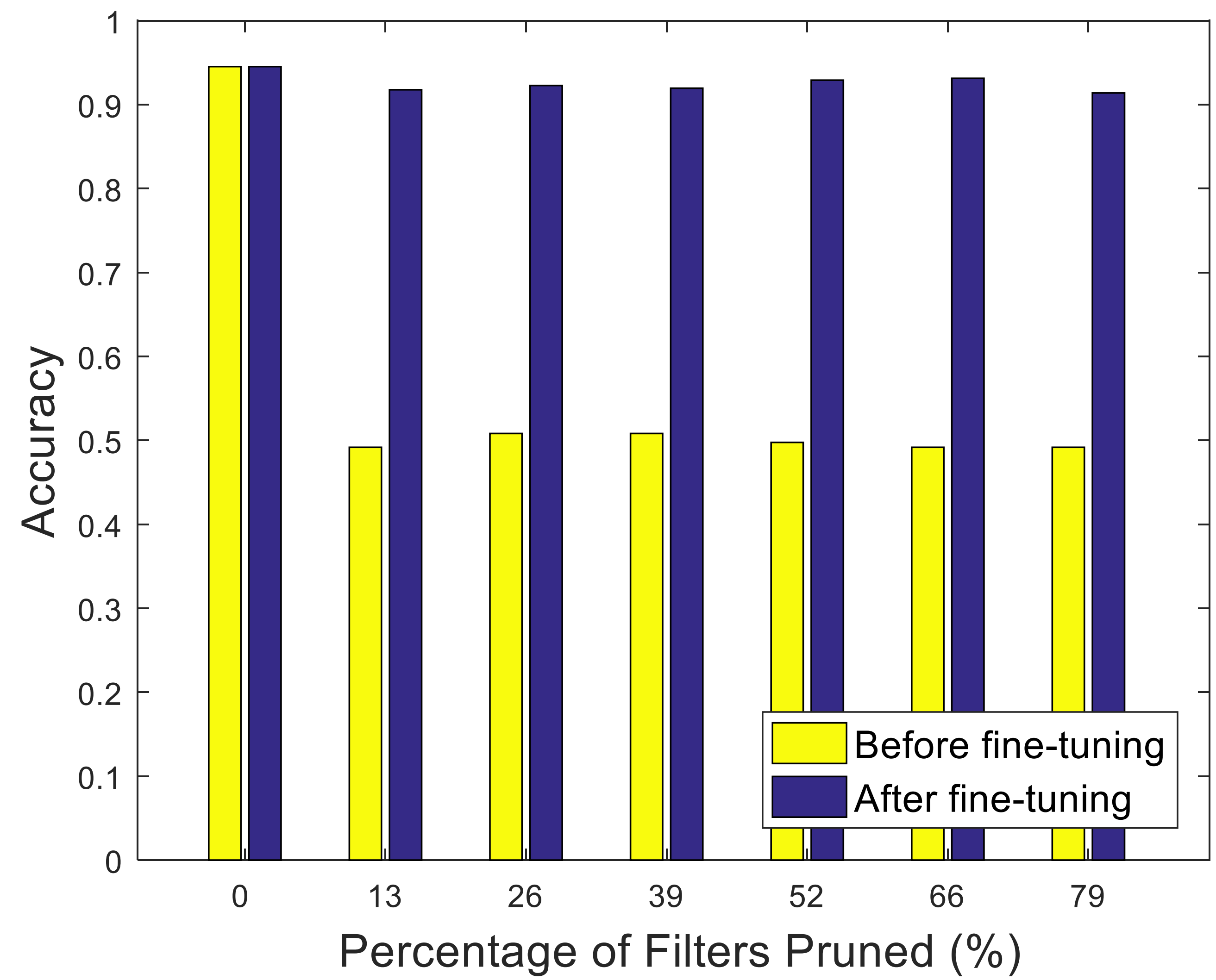
# Result – ResNet18 (He et al., 2015) with Pruning



# Result – ResNet18 (He et al., 2015) with Pruning



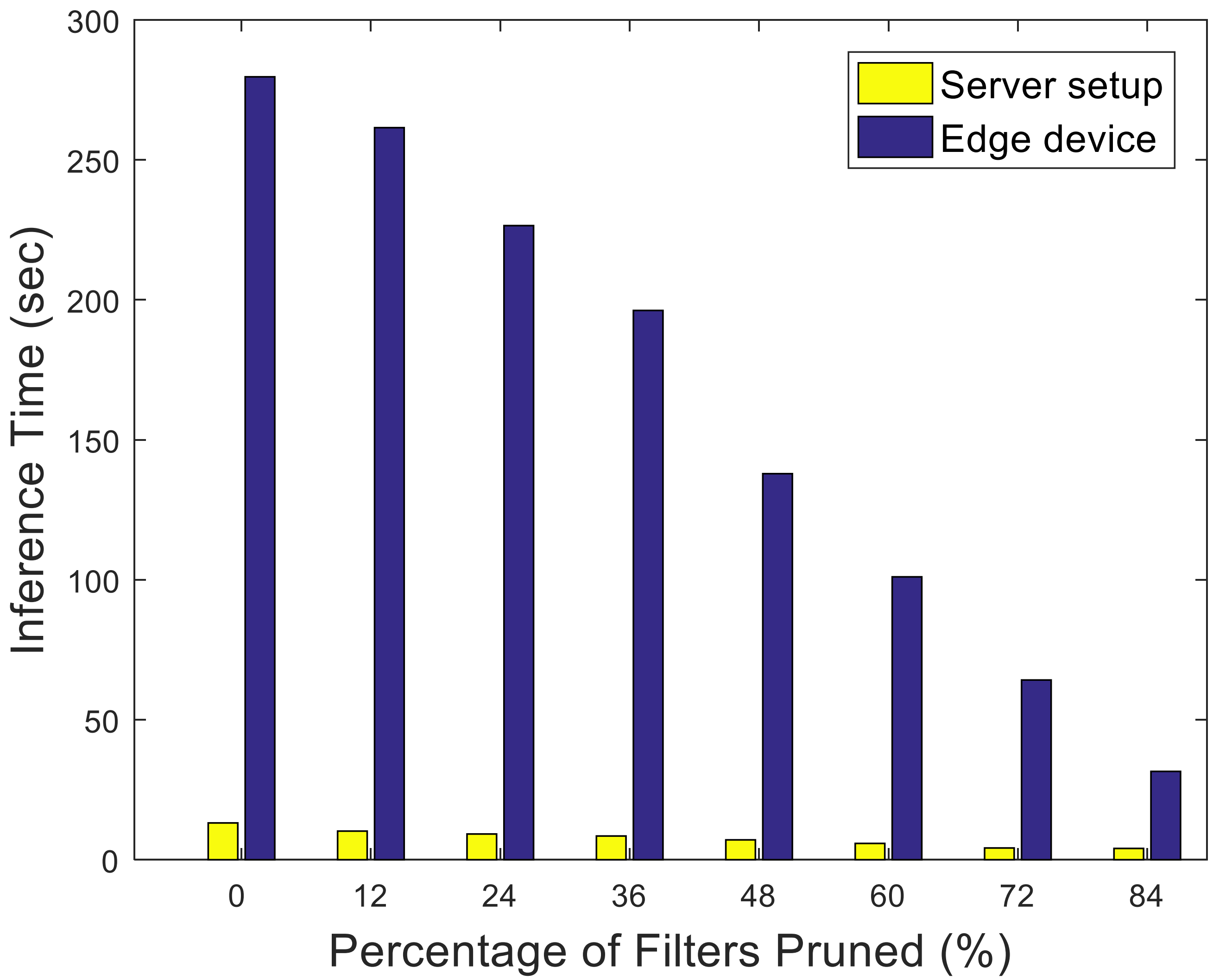

Crack



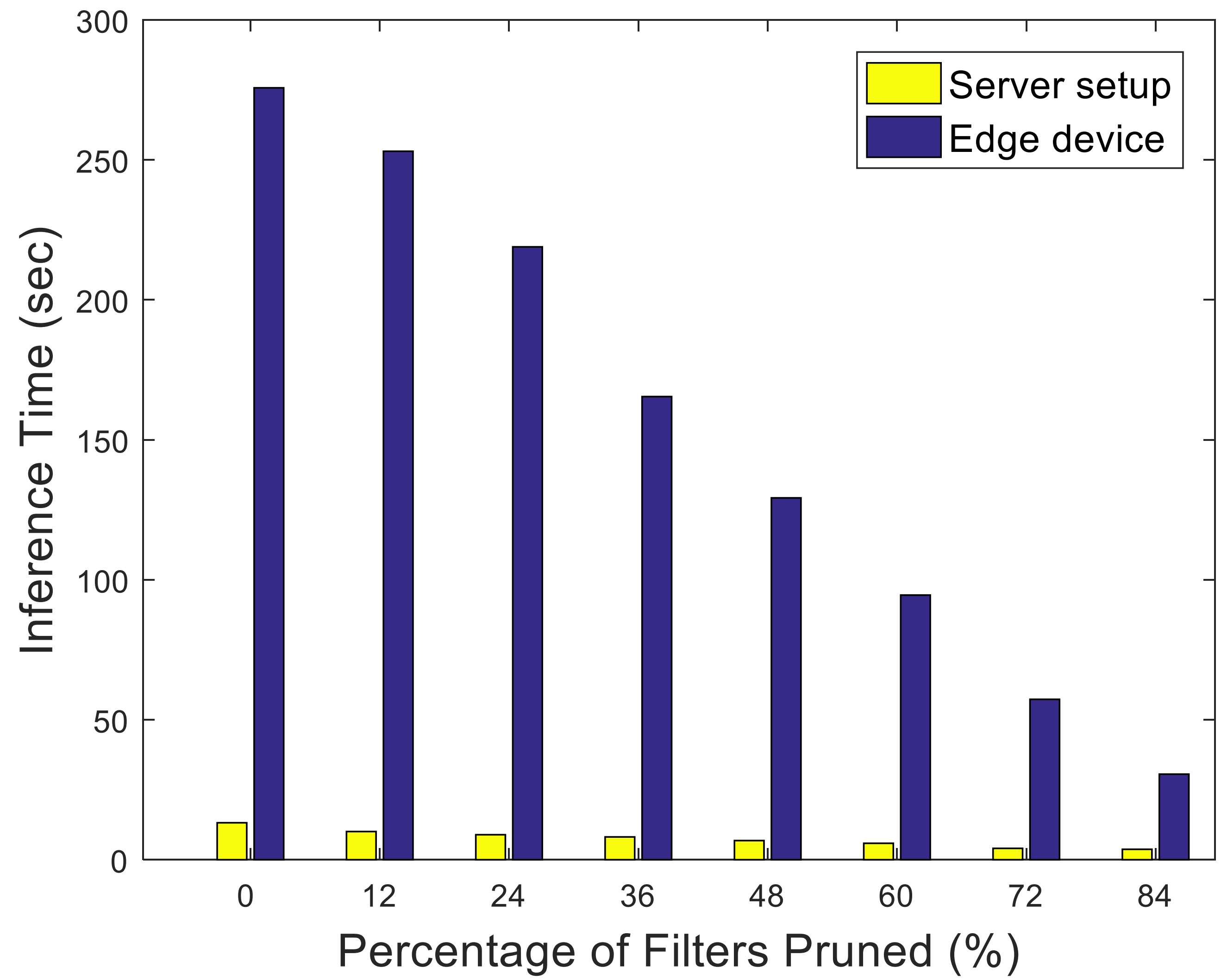

Corrosion

- **Pruning is conducted on the server device.**
- **Accuracy remains descent after pruning followed by fine-tuning.**
- **Pruning is sensitive to the network configurations.**

# Inference Time Required for Pruned VGG16




Crack

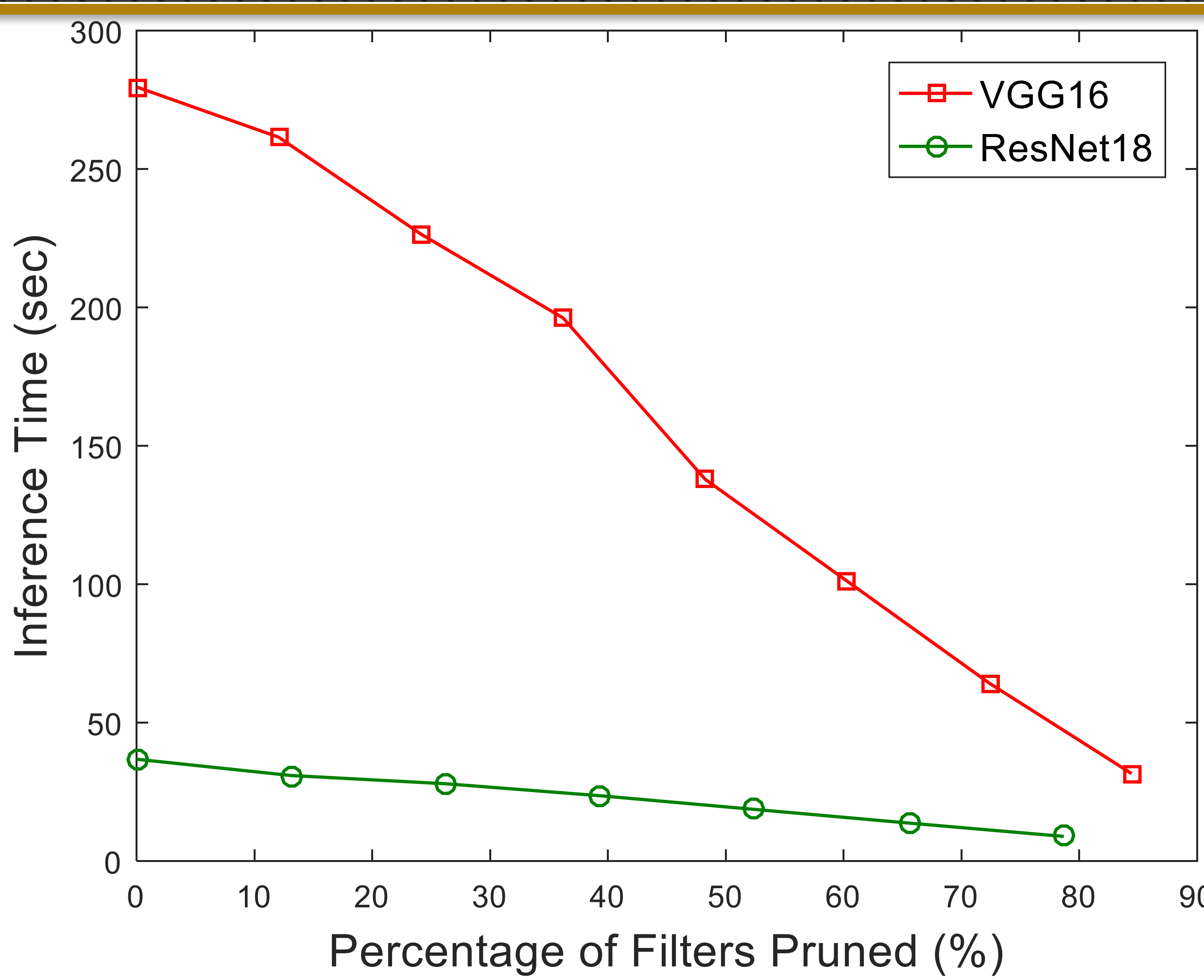



Corrosion

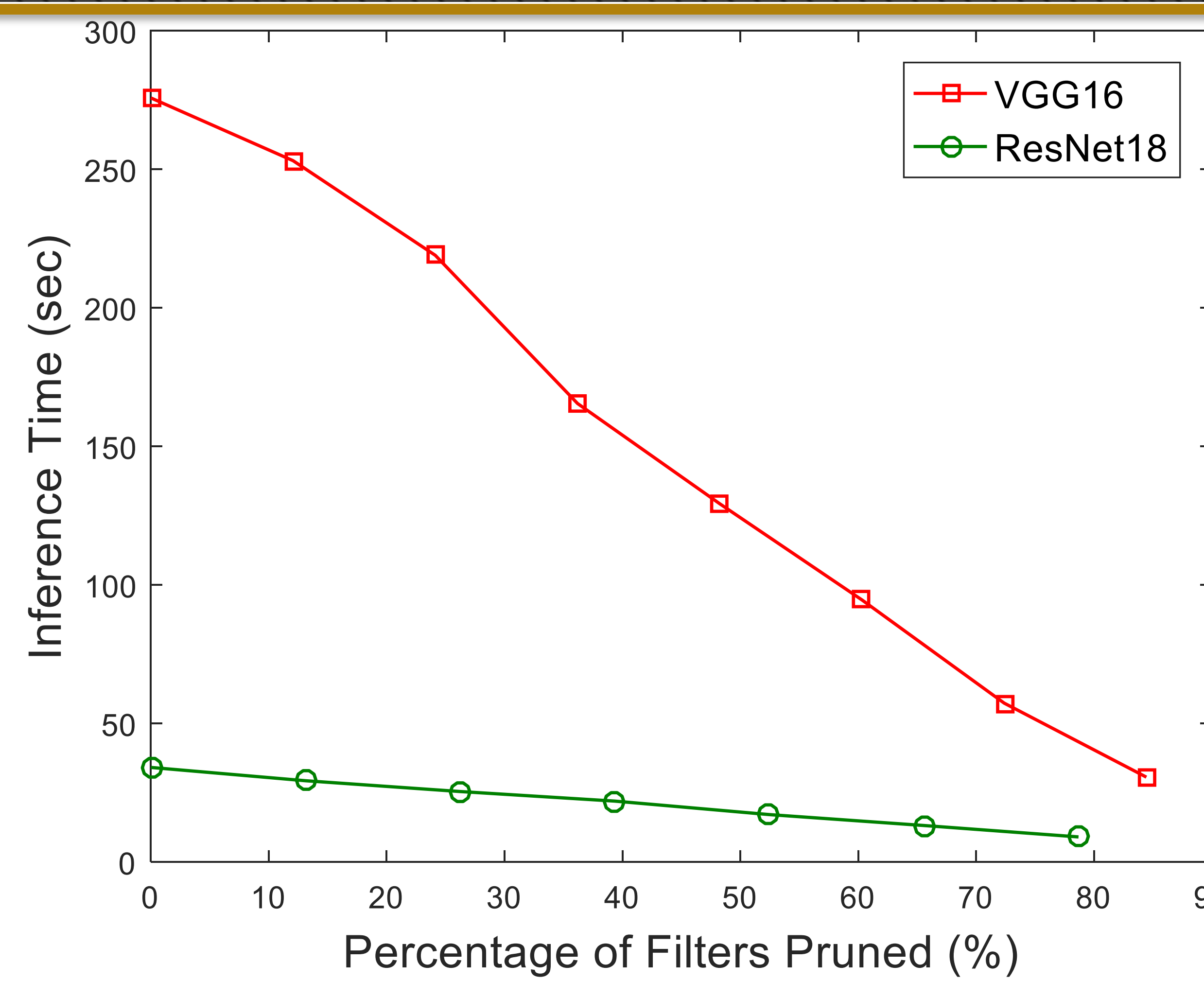
*Inference time: the total time required to classify 3,720 image patches of size 224x224.

- **Server (TITANX): 13.1 (s) is reduced to 4.0 (s) for crack data; 13.2 (s) is reduced to 3.7 (s) for corrosion data. Reduction factor: 3.5**
- **Edge (TX2): 279.7 (s) is reduced to 31.6 (s) for crack data; 275.7 (s) is reduced to 30.6 (s) for corrosion data. Reduction factor: 9**

# Inference Time on Edge Device: VGG16 VS ResNet18




Crack

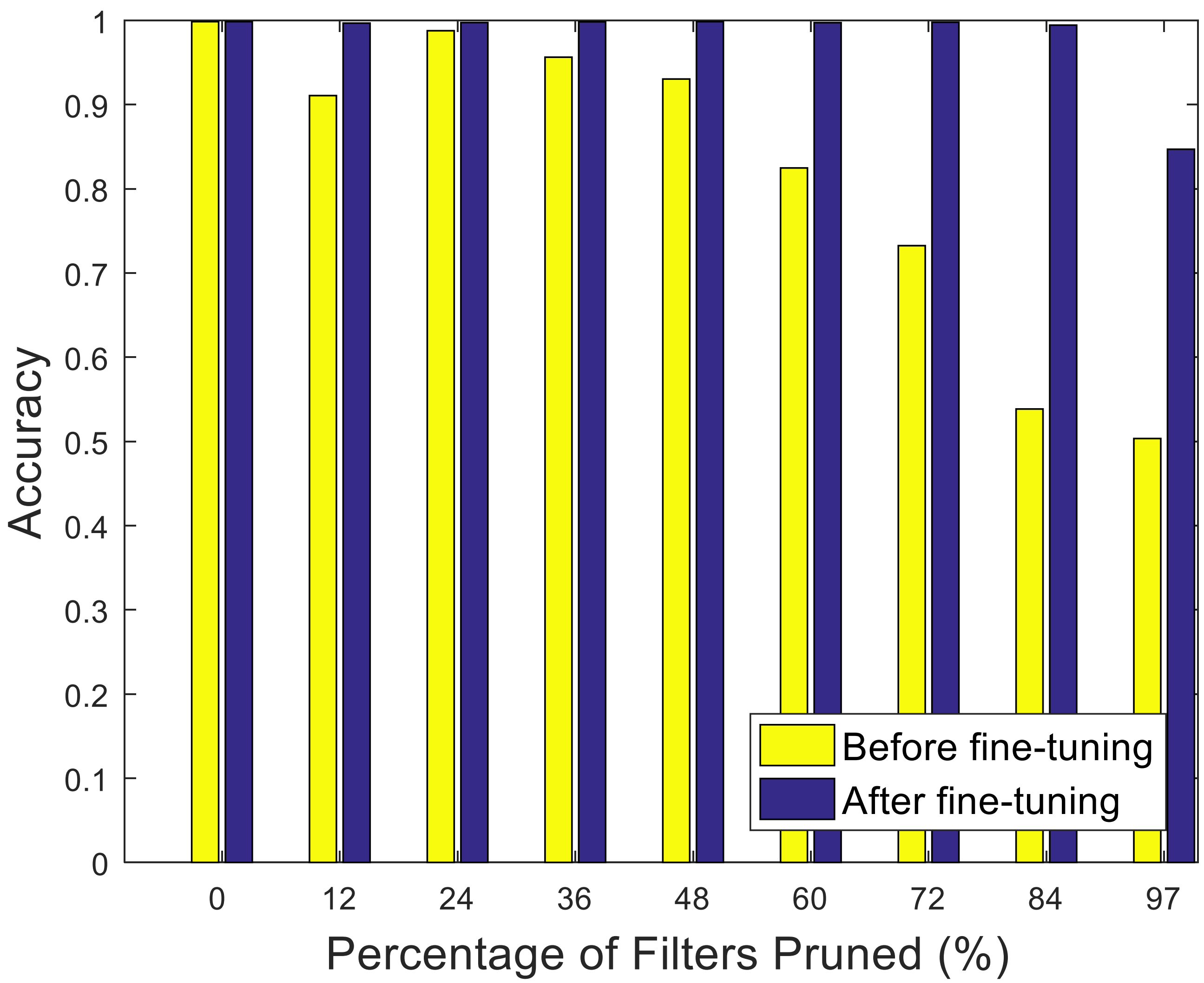



Corrosion

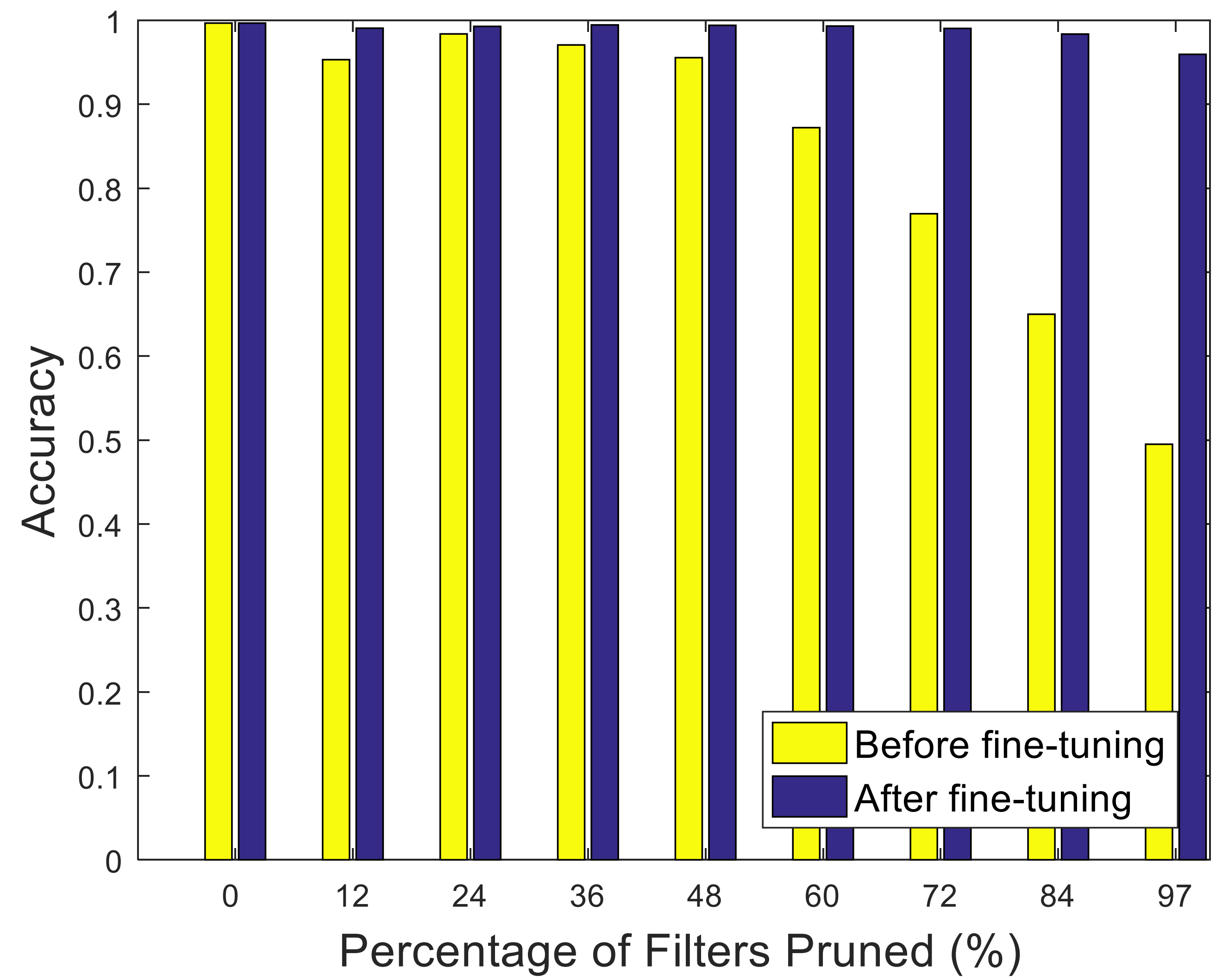
*Inference time: the total time required to classify 3,720 image patches of size 224x224.

- **Inference time**
  - **VGG16: 279.7 (s) to 31.6 (s); reduction factor: 8.9**
  - **ResNet18: 36.8 (s) to 8.9 (s); reduction factor: 4.1**
- **Memory:**
  - **VGG16: 525 (MB) to 125 (MB), 80% reduction**
  - **ResNet18: 44 (MB) to 2 (MB), 95% reduction**

# Five-fold Cross Validation Test on VGG16




Crack




Corrosion

- Mean accuracy of 5-fold cross validation test is conducted on server.
- Network fine-tuning is necessary to enhance the accuracy.

# Five-fold Cross Validation Test on VGG16 (Cont.)

| Pruned filters (%) | $\mu$ (%) before | $\mu$ (%) after | $\sigma$ (%) before | $\sigma$ (%) after |
|---|---|---|---|---|
| 0 | 99.9 | 99.9 | 0.11 | 0.11 |
| 12 | 91.1 | 99.7 | 2.43 | 0.12 |
| 24 | 98.8 | 99.7 | 1.33 | 0.11 |
| 36 | 95.6 | 99.8 | 4.05 | 0.06 |
| 48 | 93.0 | 99.9 | 8.95 | 0.05 |
| 60 | 82.5 | 99.7 | 9.12 | 0.09 |
| 72 | 73.3 | 99.8 | 19.67 | 0.10 |
| 84 | 53.8 | 99.4 | 4.59 | 0.19 |
| 97 | 50.3 | 84.7 | 0.52 | 19.86 |

Crack

| Pruned filters (%) | $\mu$ (%) before | $\mu$ (%) after | $\sigma$ (%) before | $\sigma$ (%) after |
|---|---|---|---|---|
| 0 | 99.7 | 99.7 | 0.28 | 0.28 |
| 12 | 95.3 | 99.1 | 1.76 | 0.12 |
| 24 | 98.4 | 99.3 | 0.27 | 0.21 |
| 36 | 97.1 | 99.5 | 1.40 | 0.11 |
| 48 | 95.6 | 99.4 | 1.20 | 0.05 |
| 60 | 87.2 | 99.3 | 6.26 | 0.13 |
| 72 | 77.0 | 99.0 | 16.98 | 0.20 |
| 84 | 65.0 | 98.4 | 10.20 | 0.31 |
| 97 | 49.5 | 96.0 | 0.74 | 0.95 |

Corrosion

- **The variance in the accuracy after fine-tuning is very small. However, when pruning 97% of the filters, the variance increases and the accuracy after fine-tuning drops.**
- **The pruning is stopped when the accuracy after fine-tuning drops more than 3%.**

# Five-fold Cross Validation Test on VGG16 (Cont.)

| Pruned filters (%) | $\mu$ (%) before | $\mu$ (%) after | $\sigma$ (%) before | $\sigma$ (%) after |
|---|---|---|---|---|
| 0 | 99.9 | 99.9 | 0.11 | 0.11 |
| 12 | 91.1 | 99.7 | 2.43 | 0.12 |
| 24 | 98.8 | 99.7 | 1.33 | 0.11 |
| 36 | 95.6 | 99.8 | 4.05 | 0.06 |
| 48 | 93.0 | 99.9 | 8.95 | 0.05 |
| 60 | 82.5 | 99.7 | 9.12 | 0.09 |
| 72 | 73.3 | 99.8 | 19.67 | 0.10 |
| 84 | 53.8 | 99.4 | 4.59 | 0.19 |
| 97 | 50.3 | 84.7 | 0.52 | 19.86 |

Crack

| Pruned filters (%) | $\mu$ (%) before | $\mu$ (%) after | $\sigma$ (%) before | $\sigma$ (%) after |
|---|---|---|---|---|
| 0 | 99.7 | 99.7 | 0.28 | 0.28 |
| 12 | 95.3 | 99.1 | 1.76 | 0.12 |
| 24 | 98.4 | 99.3 | 0.27 | 0.21 |
| 36 | 97.1 | 99.5 | 1.40 | 0.11 |
| 48 | 95.6 | 99.4 | 1.20 | 0.05 |
| 60 | 87.2 | 99.3 | 6.26 | 0.13 |
| 72 | 77.0 | 99.0 | 16.98 | 0.20 |
| 84 | 65.0 | 98.4 | 10.20 | 0.31 |
| 97 | 49.5 | 96.0 | 0.74 | 0.95 |

Corrosion

- **The variance in the accuracy after fine-tuning is very small. However, when pruning 97% of the filters, the variance increases and the accuracy after fine-tuning drops.**
- **The pruning is stopped when the accuracy after fine-tuning drops more than 3%.**

# Five-fold Cross Validation Test on VGG16 (Cont.)

| Pruned filters (%) | $\mu$ (%) before | $\mu$ (%) after | $\sigma$ (%) before | $\sigma$ (%) after |
|---|---|---|---|---|
| 0 | 99.9 | 99.9 | 0.11 | 0.11 |
| 12 | 91.1 | 99.7 | 2.43 | 0.12 |
| 24 | 98.8 | 99.7 | 1.33 | 0.11 |
| 36 | 95.6 | 99.8 | 4.05 | 0.06 |
| 48 | 93.0 | 99.9 | 8.95 | 0.05 |
| 60 | 82.5 | 99.7 | 9.12 | 0.09 |
| 72 | 73.3 | 99.8 | 19.67 | 0.10 |
| 84 | 53.8 | 99.4 | 4.59 | 0.19 |
| 97 | 50.3 | 84.7 | 0.52 | 19.86 |

Crack

| Pruned filters (%) | $\mu$ (%) before | $\mu$ (%) after | $\sigma$ (%) before | $\sigma$ (%) after |
|---|---|---|---|---|
| 0 | 99.7 | 99.7 | 0.28 | 0.28 |
| 12 | 95.3 | 99.1 | 1.76 | 0.12 |
| 24 | 98.4 | 99.3 | 0.27 | 0.21 |
| 36 | 97.1 | 99.5 | 1.40 | 0.11 |
| 48 | 95.6 | 99.4 | 1.20 | 0.05 |
| 60 | 87.2 | 99.3 | 6.26 | 0.13 |
| 72 | 77.0 | 99.0 | 16.98 | 0.20 |
| 84 | 65.0 | 98.4 | 10.20 | 0.31 |
| 97 | 49.5 | 96.0 | 0.74 | 0.95 |

Corrosion

- **The variance in the accuracy after fine-tuning is very small. However, when pruning 97% of the filters, the variance increases and the accuracy after fine-tuning drops.**
- **The pruning is stopped when the accuracy after fine-tuning drops more than 3%.**

# Five-fold Cross Validation Test on VGG16 (Cont.)

| Pruned filters (%) | $\mu$ (%) before | $\mu$ (%) after | $\sigma$ (%) before | $\sigma$ (%) after |
|---|---|---|---|---|
| 0 | 99.9 | 99.9 | 0.11 | 0.11 |
| 12 | 91.1 | 99.7 | 2.43 | 0.12 |
| 24 | 98.8 | 99.7 | 1.33 | 0.11 |
| 36 | 95.6 | 99.8 | 4.05 | 0.06 |
| 48 | 93.0 | 99.9 | 8.95 | 0.05 |
| 60 | 82.5 | 99.7 | 9.12 | 0.09 |
| 72 | 73.3 | 99.8 | 19.67 | 0.10 |
| 84 | 53.8 | 99.4 | 4.59 | 0.19 |
| 97 | 50.3 | 84.7 | 0.52 | 19.86 |

Crack

| Pruned filters (%) | $\mu$ (%) before | $\mu$ (%) after | $\sigma$ (%) before | $\sigma$ (%) after |
|---|---|---|---|---|
| 0 | 99.7 | 99.7 | 0.28 | 0.28 |
| 12 | 95.3 | 99.1 | 1.76 | 0.12 |
| 24 | 98.4 | 99.3 | 0.27 | 0.21 |
| 36 | 97.1 | 99.5 | 1.40 | 0.11 |
| 48 | 95.6 | 99.4 | 1.20 | 0.05 |
| 60 | 87.2 | 99.3 | 6.26 | 0.13 |
| 72 | 77.0 | 99.0 | 16.98 | 0.20 |
| 84 | 65.0 | 98.4 | 10.20 | 0.31 |
| 97 | 49.5 | 96.0 | 0.74 | 0.95 |

Corrosion

- **The variance in the accuracy after fine-tuning is very small. However, when pruning 97% of the filters, the variance increases and the accuracy after fine-tuning drops.**
- **The pruning is stopped when the accuracy after fine-tuning drops more than 3%.**

# Summary

- **Network pruning combined with transfer learning can achieve efficient inference when there is limited training data and computing power.**

- **By network pruning, the inference time on edge device is nine and four times faster than the original VGG16 and ResNet18. The network size is reduced by 80% and 95% for the VGG16 and ResNet18 networks, respectively.**

- **Different network configurations exhibit different behaviors with respect to pruning.**

- **Sensitive analysis shows that pruning can be achieved by using a smaller number of fine-tuning without losing detection performance.**

- **The computation gain on the edge device is more prominent than the gain on the server device.**

# Thank you