A Study of Transfer Learning for Skin Lesion Classification

Fábio Maia
https://fabiomaia.github.io

Structure

1. **Motivation**
2. **Objectives**
3. Background
4. Transfer Learning Experiments
5. End-to-end Learning Experiments
6. Conclusions

Motivation

- Deep learning is very exciting but requires a large amount of training data and vast computational resources
- In practice, obtaining a large dataset for training a model for a particular task is often very difficult and most research teams also don’t have many computational resources
- Transfer learning emerges as a strategy for deep learning that doesn’t require as much data or computational resources
- In particular, skin lesion classification is an application which transfer learning can solve quite usefully in order to aid diagnoses of skin lesions

Objectives

1. Train models based on different strategies of transfer learning from the VGG16 model trained on ImageNet
2. Train models of custom CNN architectures based around reasonable heuristics
3. Compare and discuss results

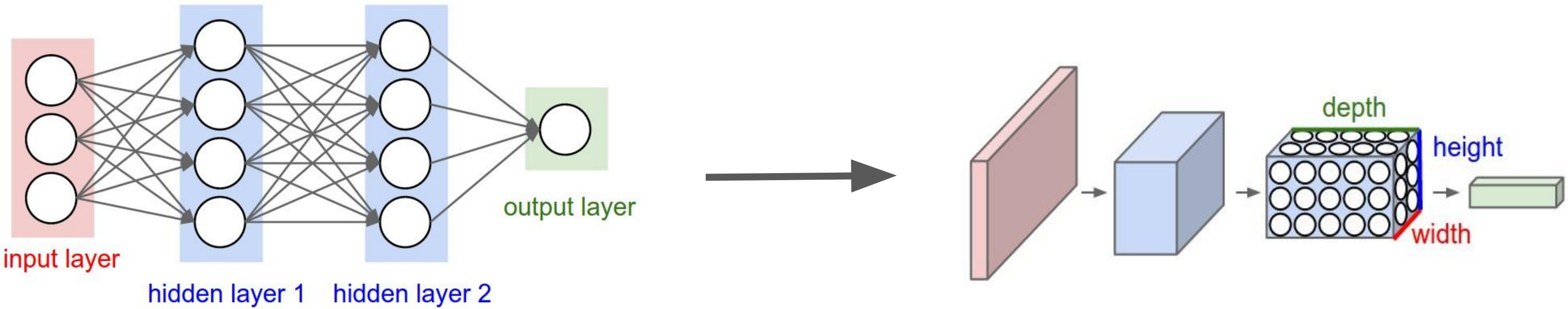
Structure

1. Motivation
2. Objectives
3. **Background**
4. Transfer Learning Experiments
5. End-to-end Learning Experiments
6. Conclusions

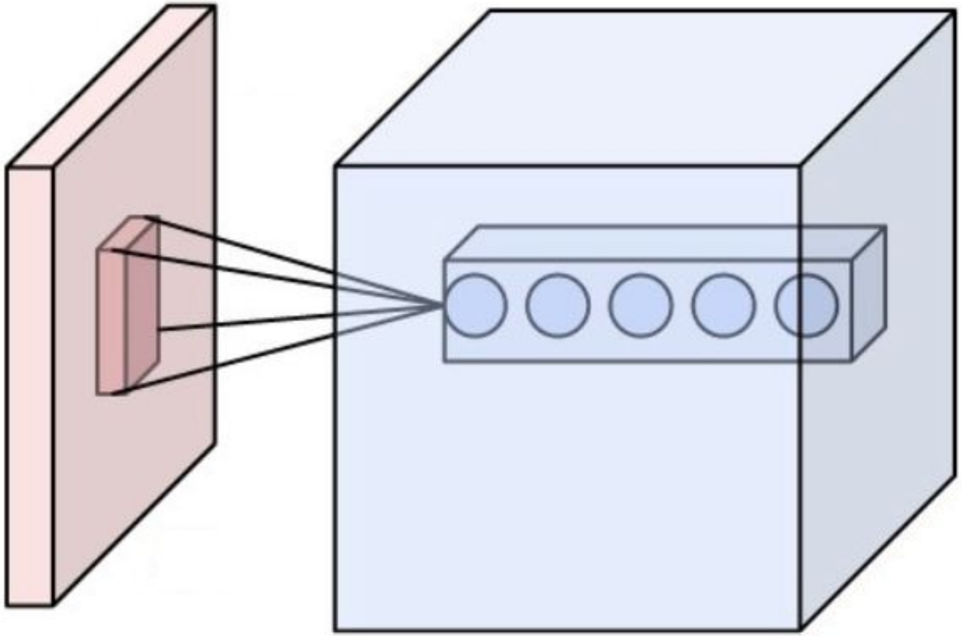
Convolutional Neural Networks

- FCNN are a class of feedforward ANN, that unfortunately don’t scale well to most problems in computer vision due to the curse of dimensionality
- CNN are another class of feedforward ANN, originally designed for computer vision problems, whose architecture takes advantage of the assumption that the input is an image

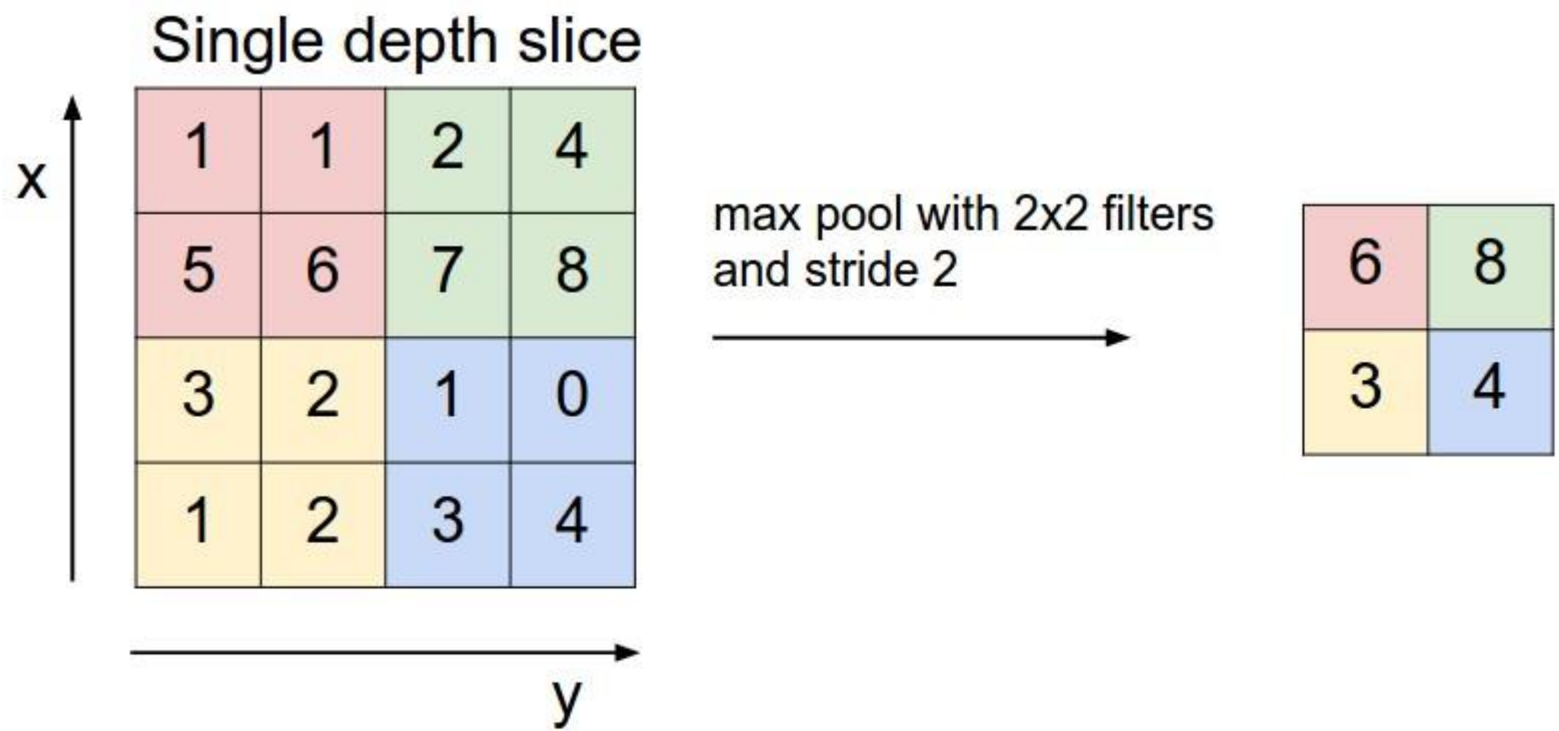
CNN: three-dimensional composition of neurons



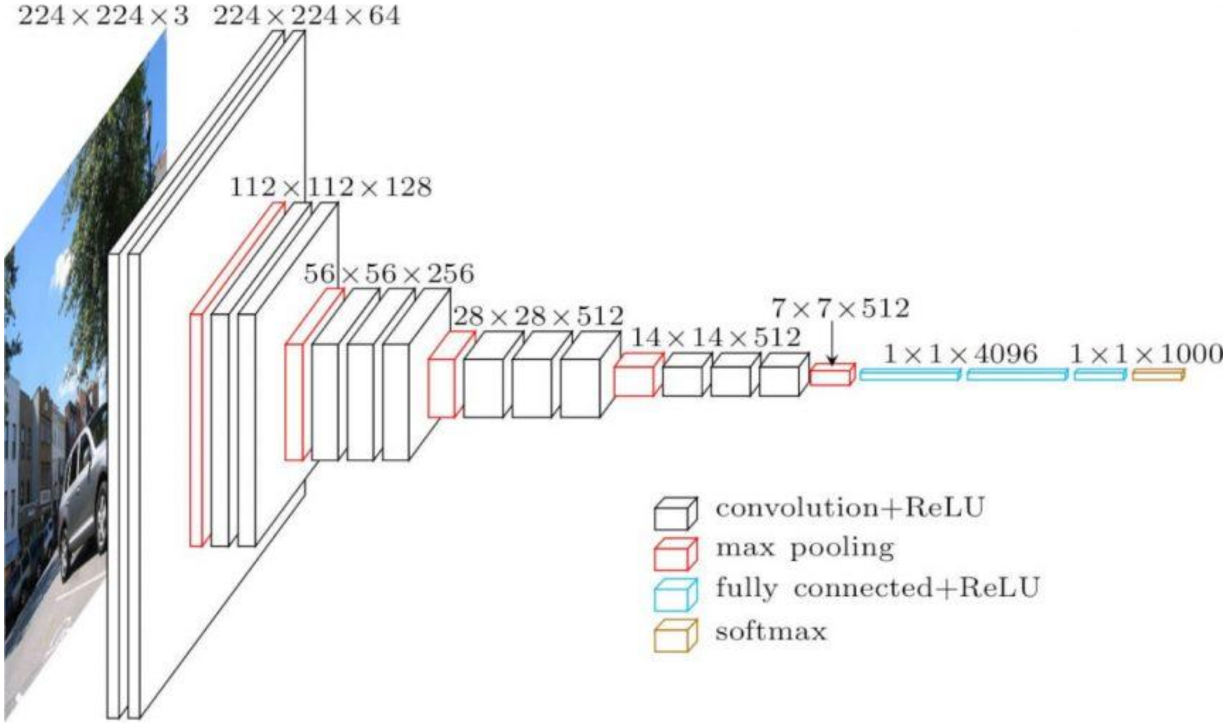
CNN: local connectivity and parameter sharing



CNN: downsampling



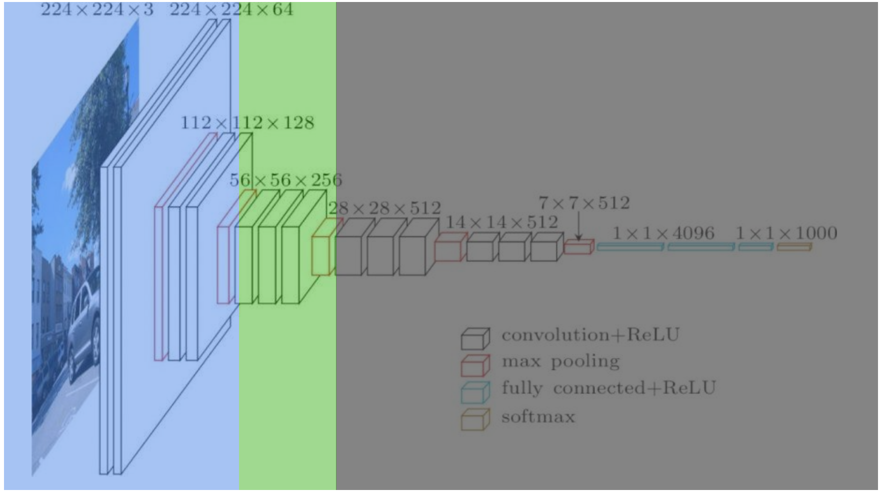
VGG16



224 × 224 × 3
224 × 224 × 64
112 × 112 × 128
56 × 56 × 256
28 × 28 × 512
14 × 14 × 512
7 × 7 × 512
1 × 1 × 4096
1 × 1 × 1000
convolution+ReLU
max pooling
fully connected+ReLU
softmax

Transfer Learning

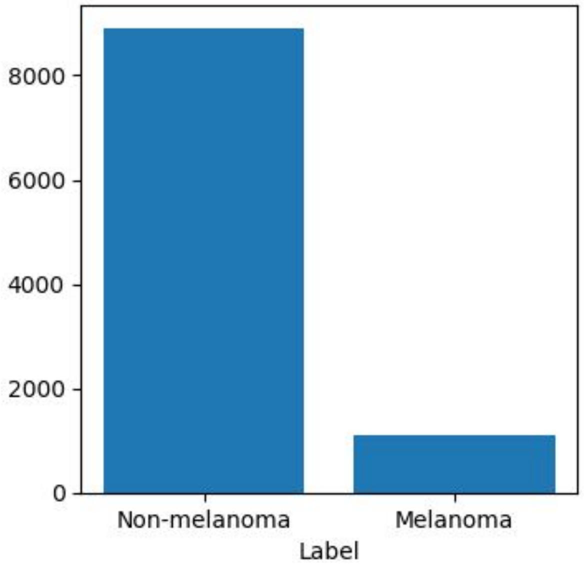
- Total Parameter Extraction without Fine Tuning
- Total Parameter Extraction with Fine Tuning
- Partial Parameter Extraction



Structure

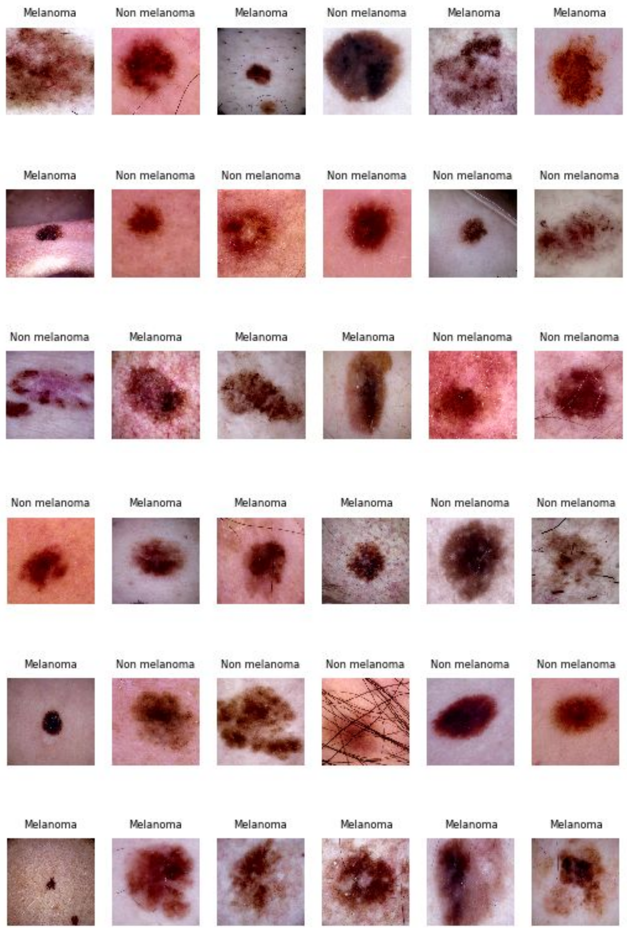
1. Motivation
2. Objectives
3. Background
4. **Transfer Learning Experiments**
5. End-to-end Learning Experiments
6. Conclusions

ISIC 2018 Dataset

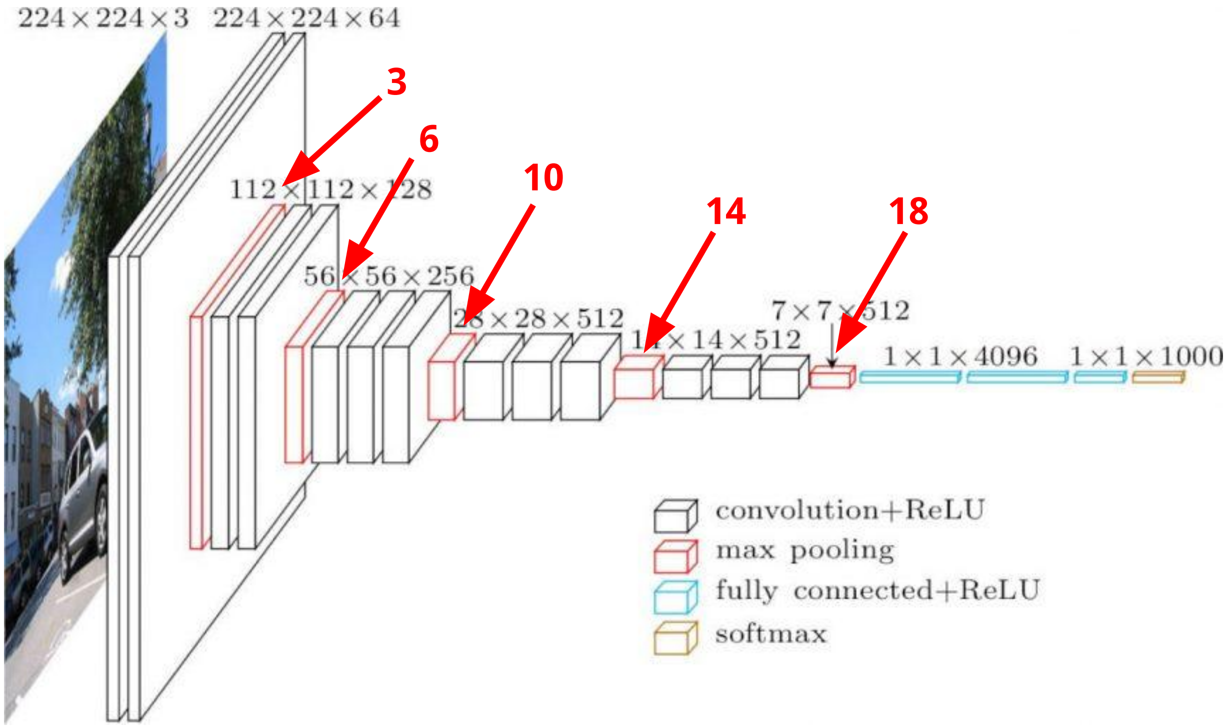


class balance

augmentation



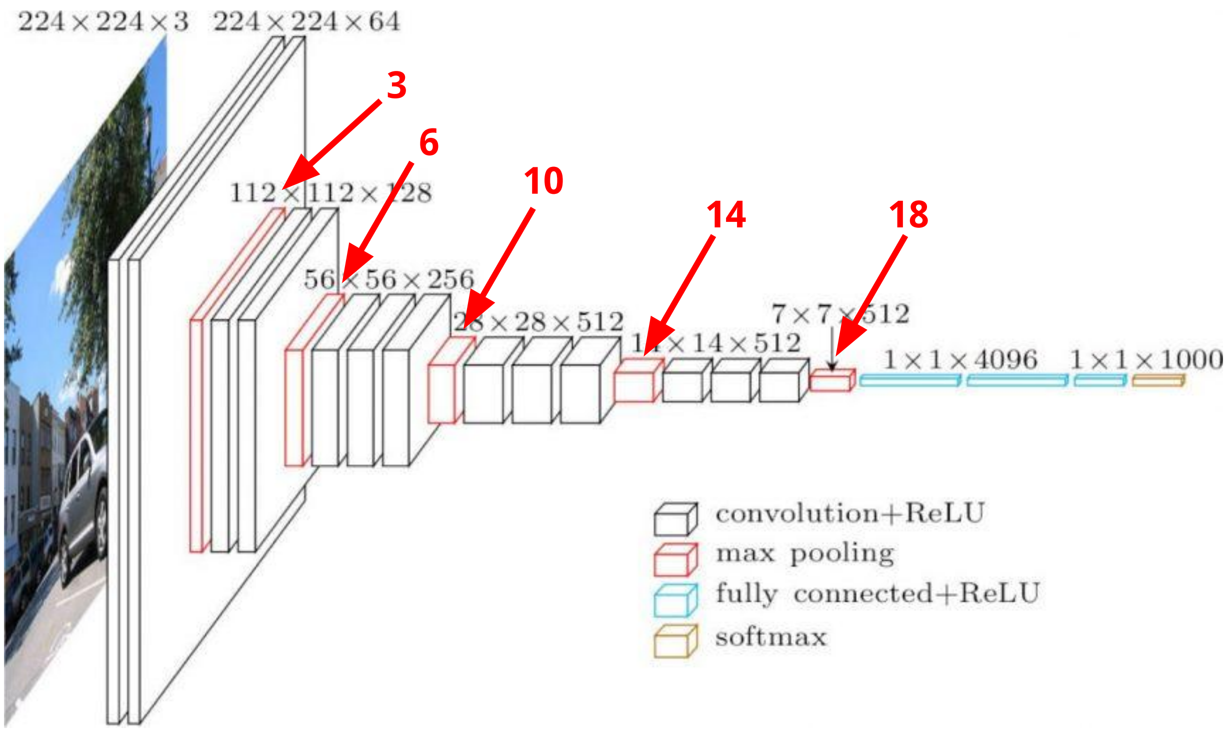
VGG16


224 × 224 × 3
224 × 224 × 64
112 × 112 × 128
56 × 56 × 256
28 × 28 × 512
14 × 14 × 512
7 × 7 × 512
1 × 1 × 4096
1 × 1 × 1000
3
6
10
14
18
convolution+ReLU
max pooling
fully connected+ReLU
softmax

Transfer Learning Common Procedure

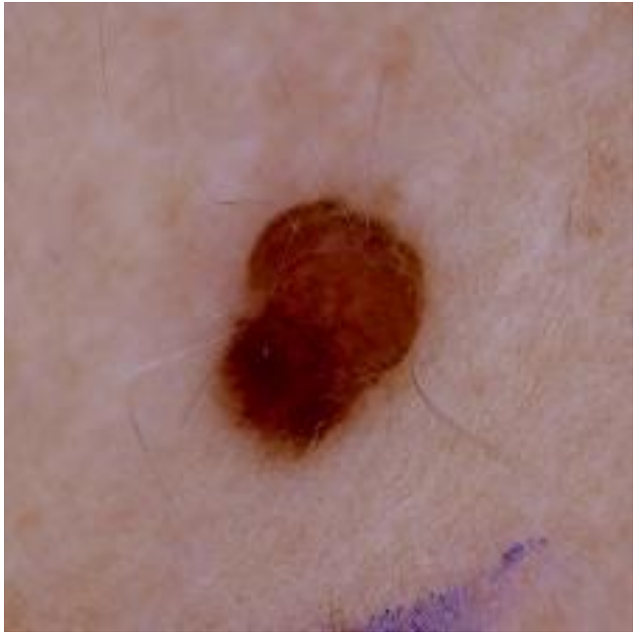
1. Standardize training and validation samples relative to ImageNet
2. Define specific network architecture
   - Extract ***e*** layers and freeze ***f*** layers from the pre-trained model
   - Global average pooling
   - 1 fully-connected layer of 512 ReLU-activated neurons
   - 1 fully-connected layer of 1 sigmoid-activated neuron
3. Some parameters are transferred and others follow Xavier initialization
4. SGD with 32 sample batches and momentum $\gamma = 0.9$
   - Binary cross entropy cost function and explicit L2 regularization with cross-validated λ
   - Initial learning rate $\eta = 10^{-4}$ that decays by a factor of 10 if validation accuracy has not improved $+10^{-3}$ in the last 10 epochs
   - Shuffle samples every epoch
   - Train for a maximum of 1000 epochs, stopping early if the loss has not changed $\pm 10^{-3}$ in the last 30 epochs

VGG16

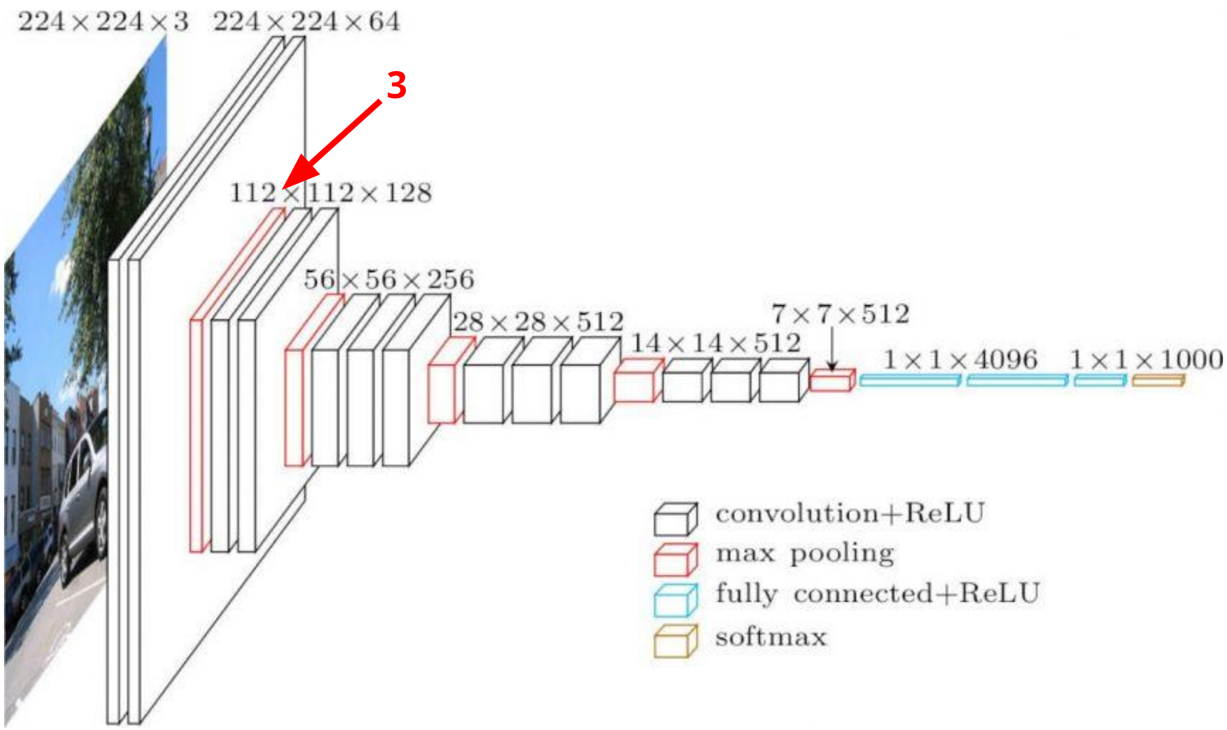


224 × 224 × 3
224 × 224 × 64
3
6
112 × 112 × 128
10
56 × 56 × 256
14
18
28 × 28 × 512
7 × 7 × 512
14 × 14 × 512
1 × 1 × 4096
1 × 1 × 1000
convolution+ReLU
max pooling
fully connected+ReLU
softmax

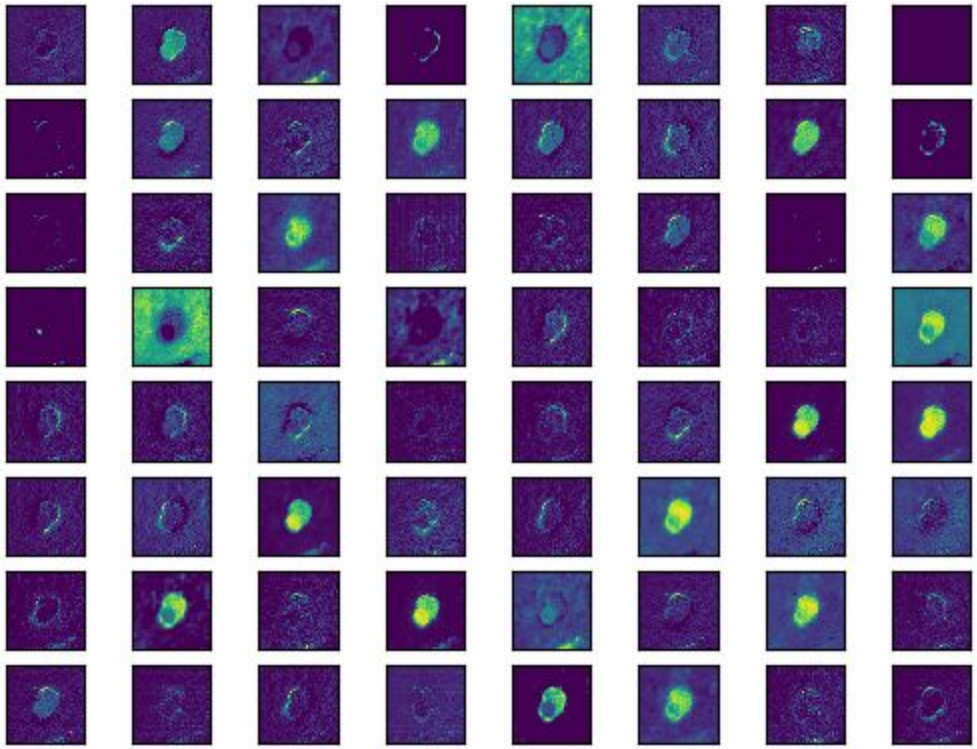
Example input through the network



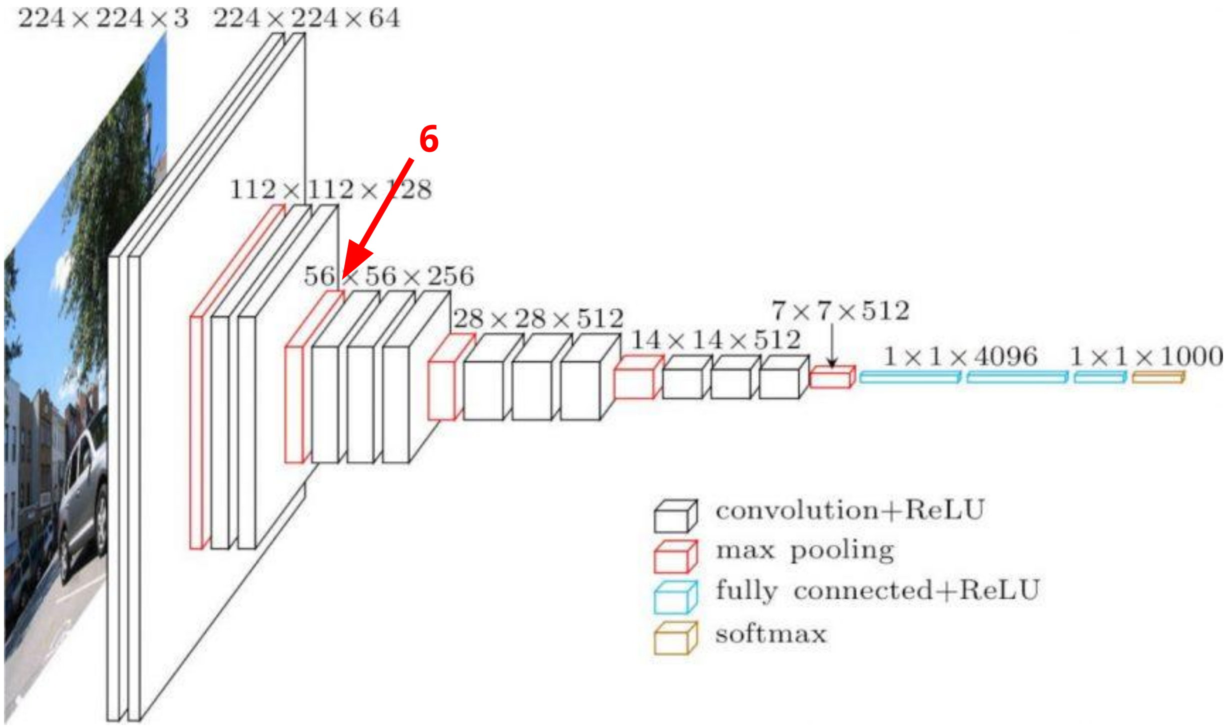
Example input through the network



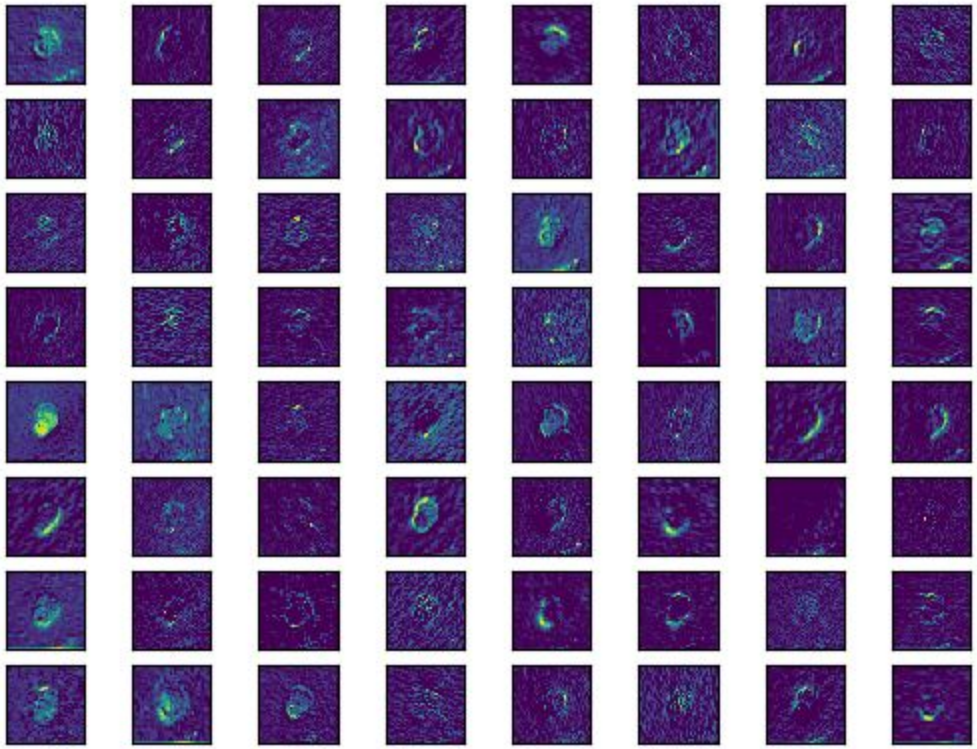
Example input through the network



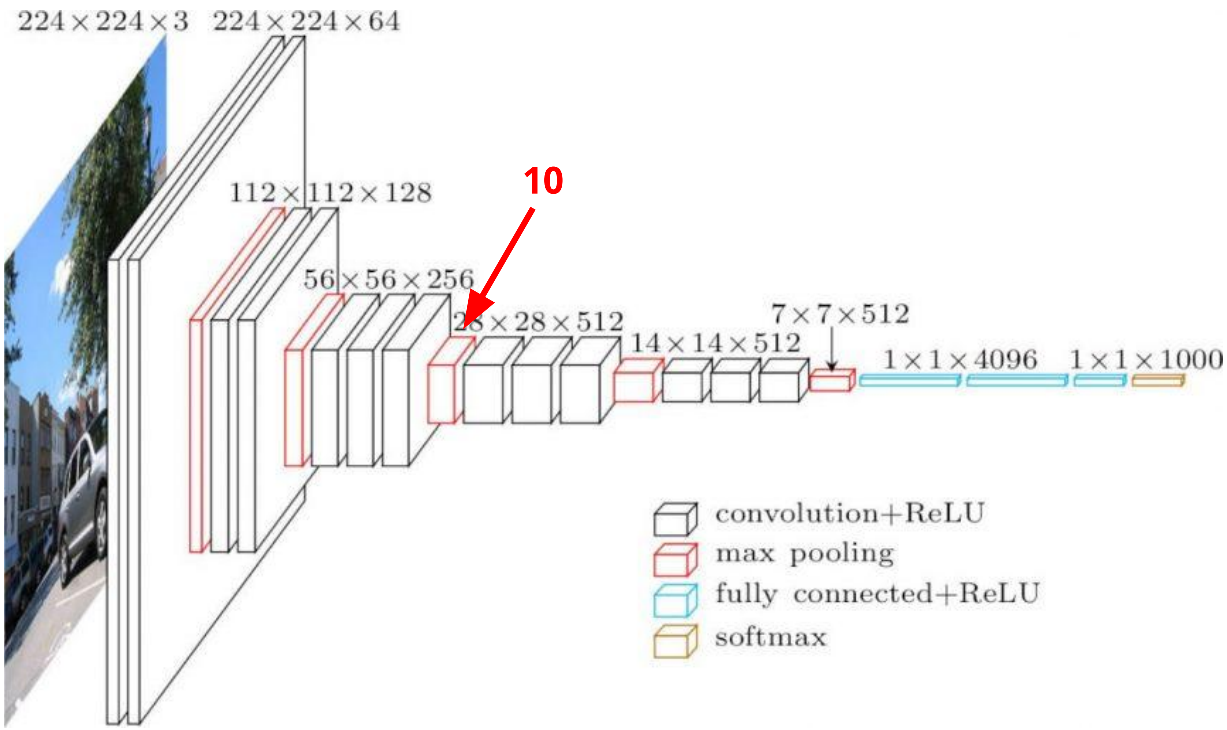
Example input through the network



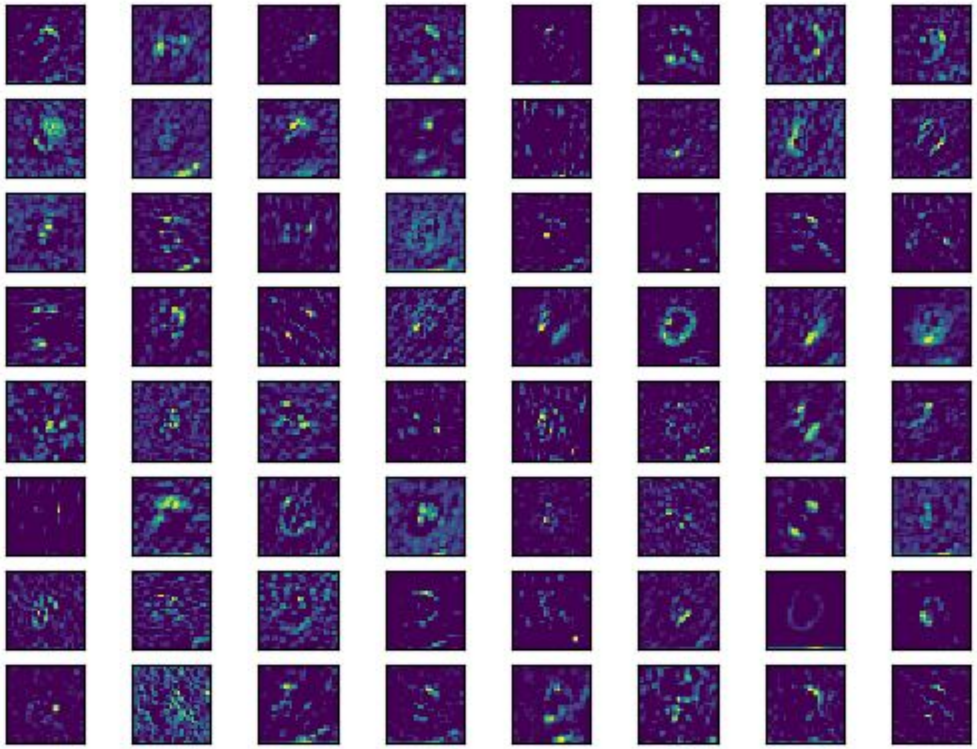
Example input through the network



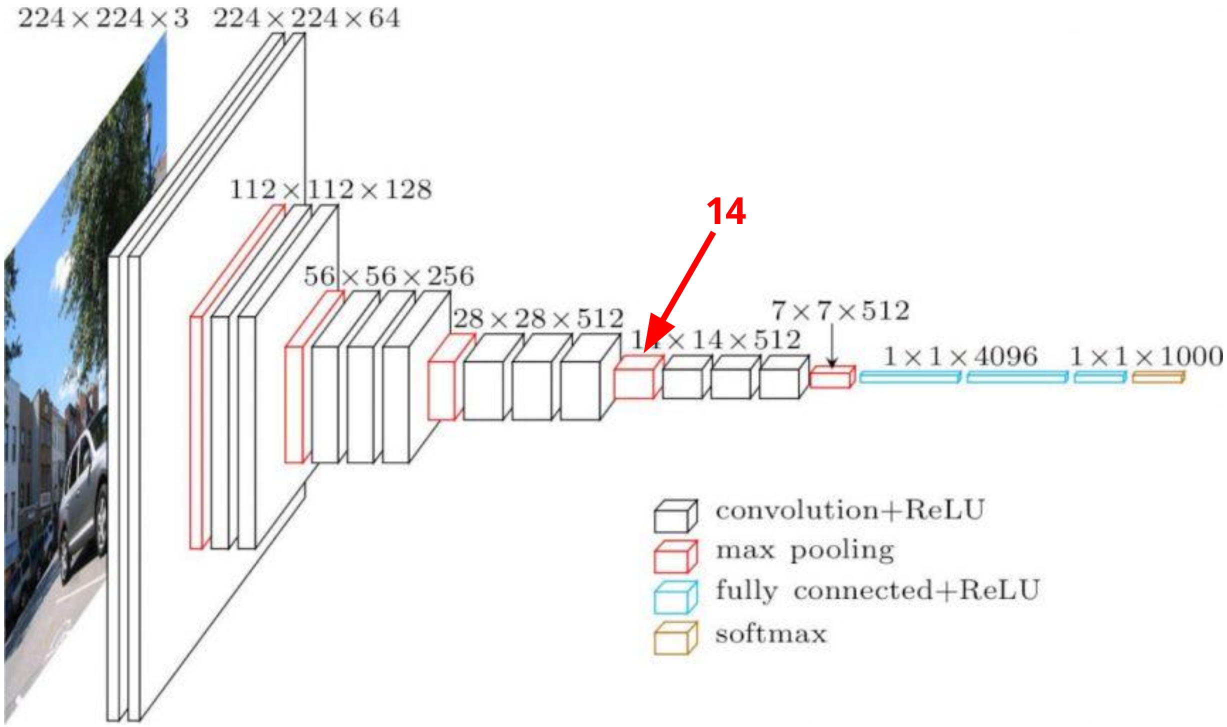
Example input through the network



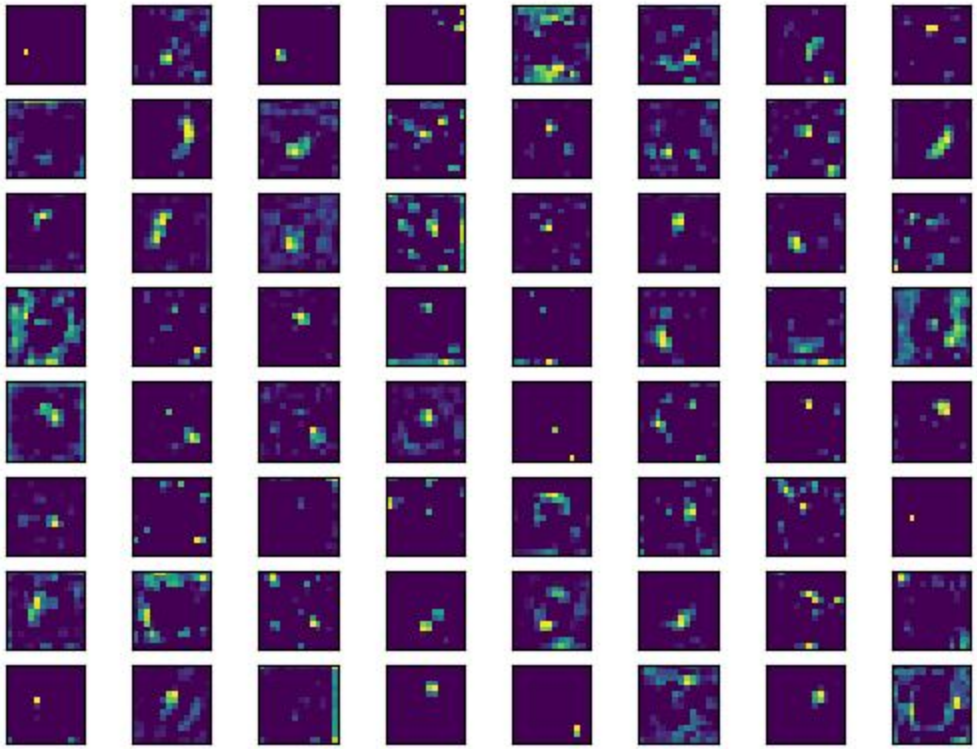
Example input through the network



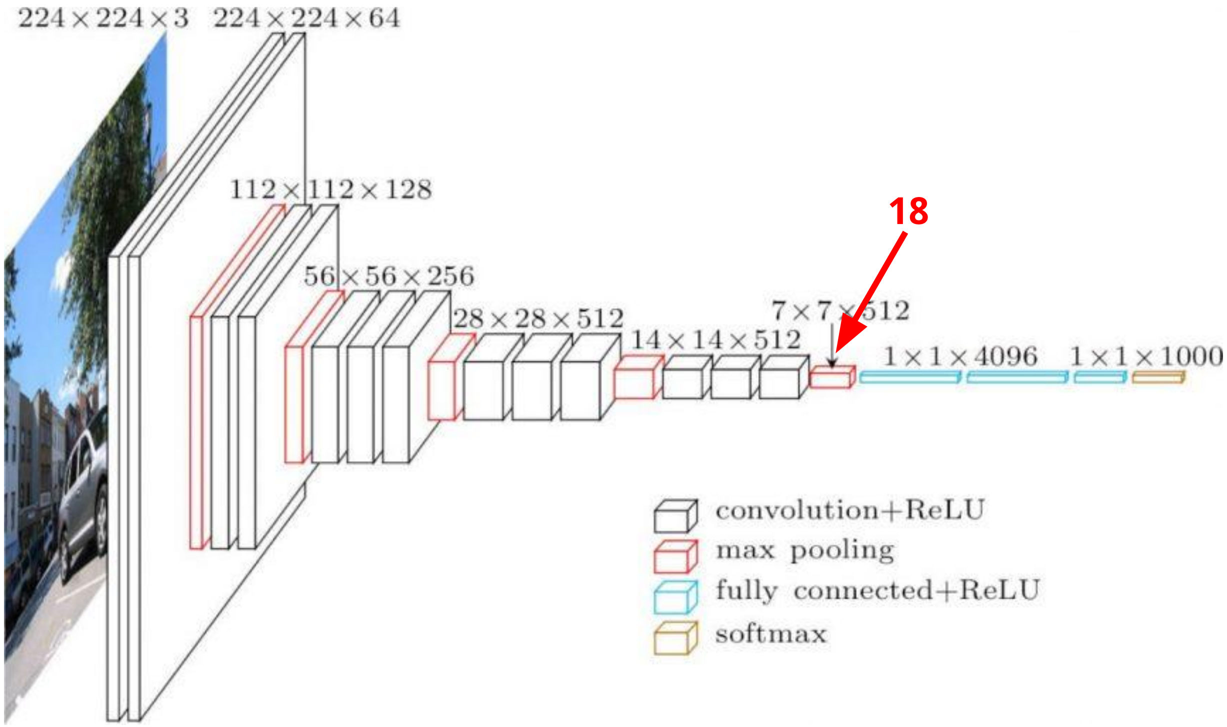
Example input through the network



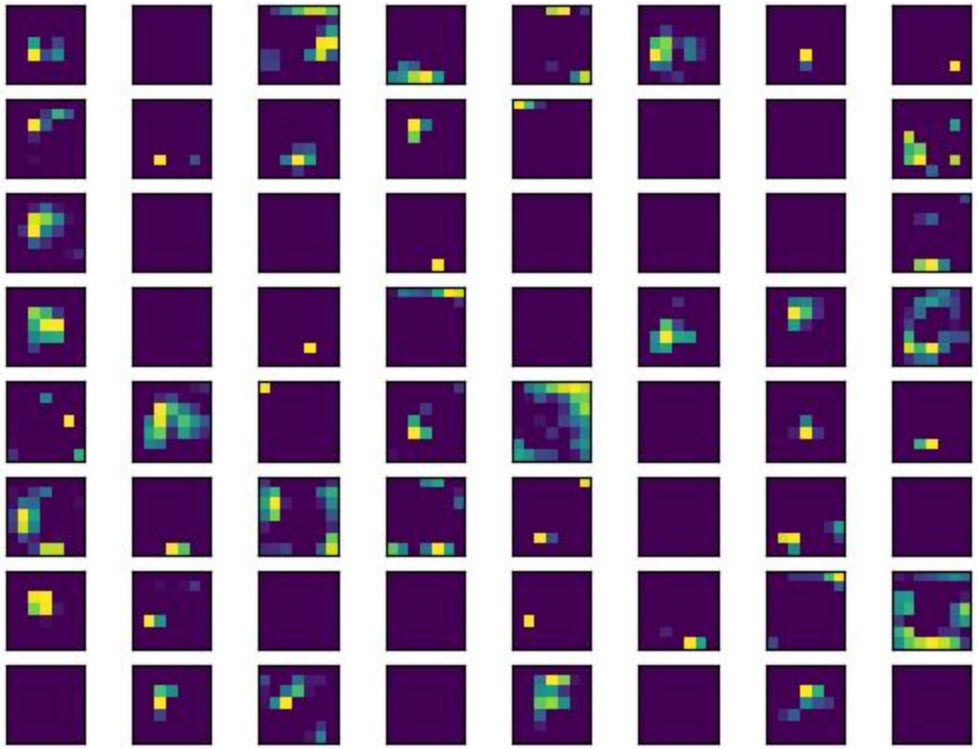
Example input through the network



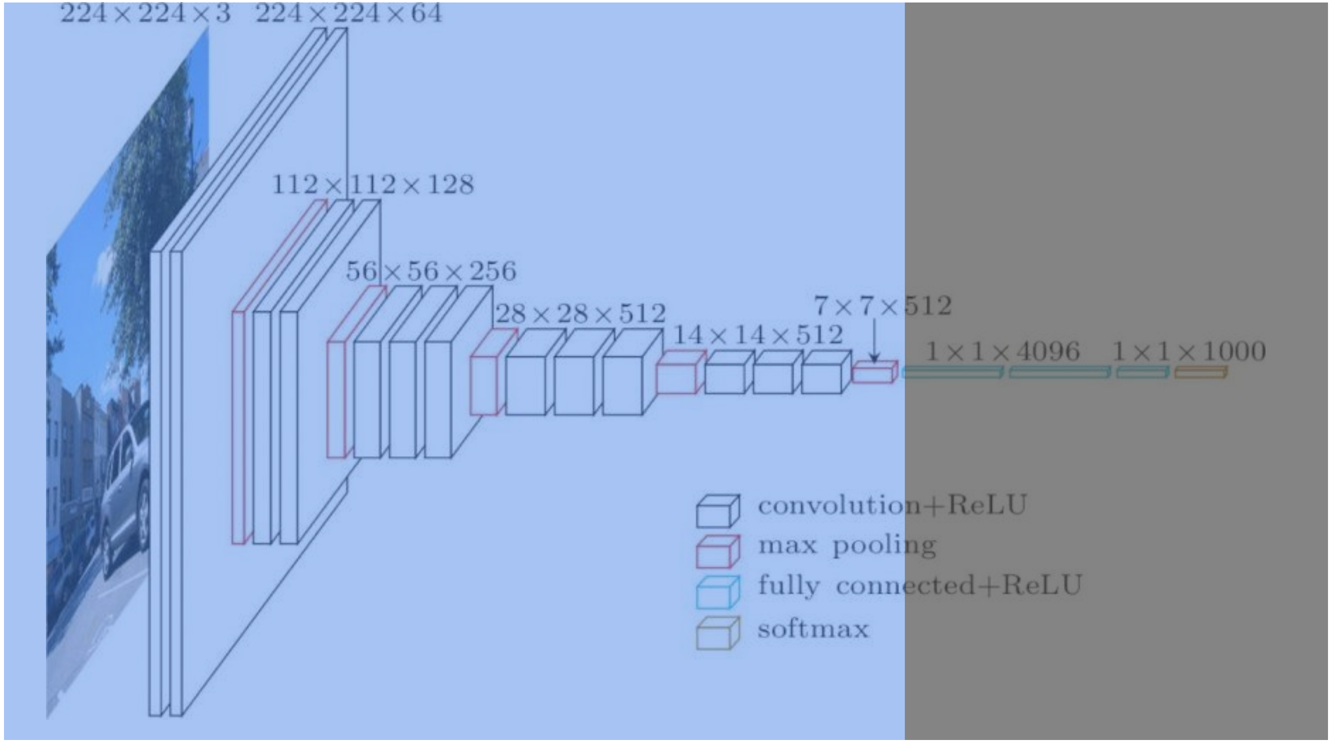
Example input through the network



Example input through the network

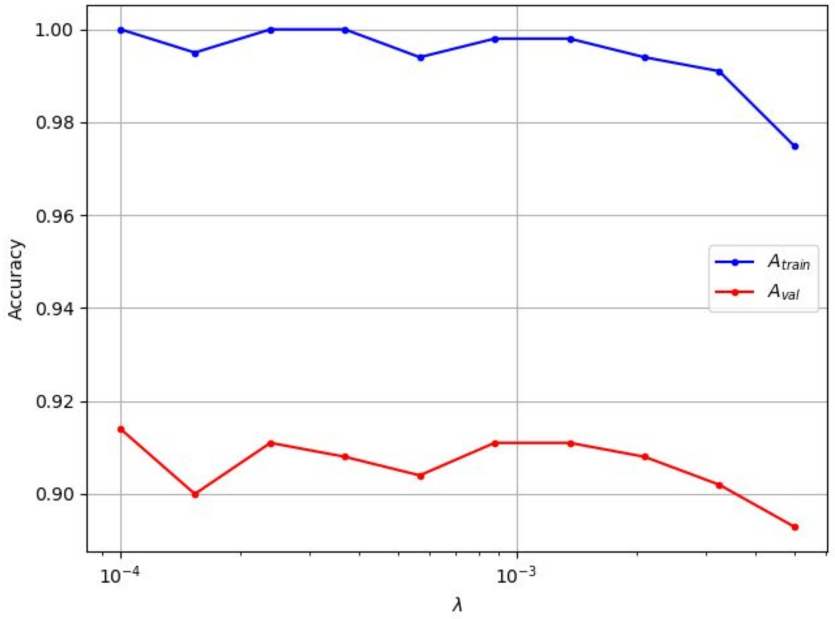


Total Parameter Extraction without Fine Tuning

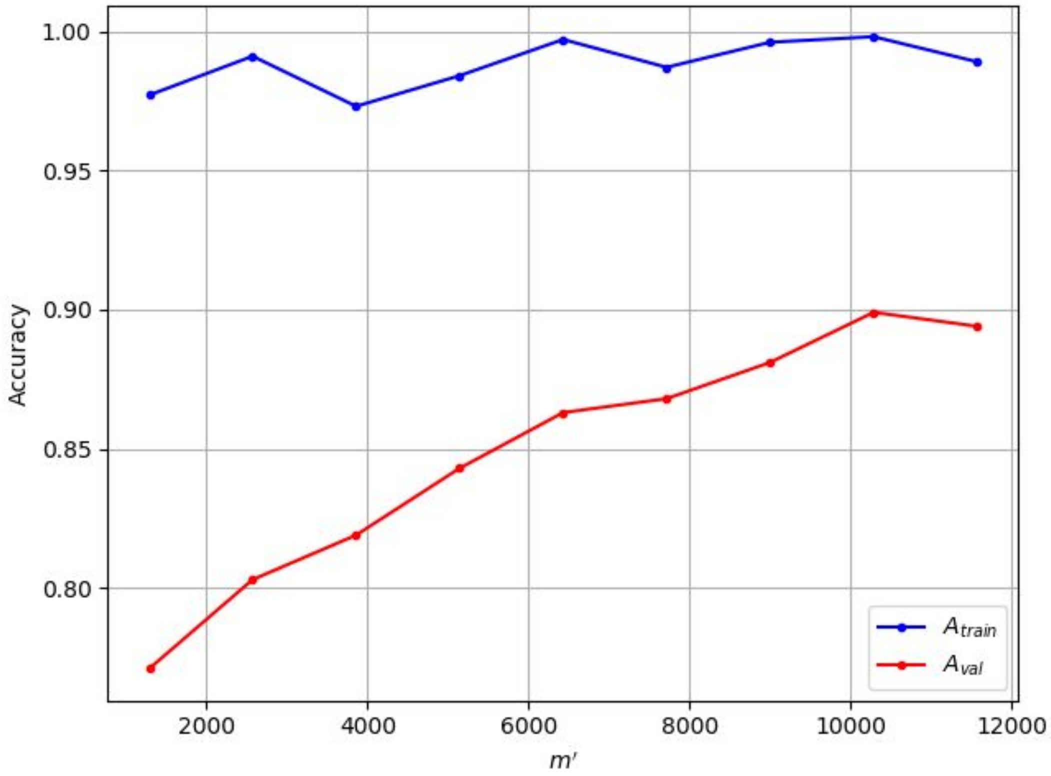


Good generalization performance on validation set

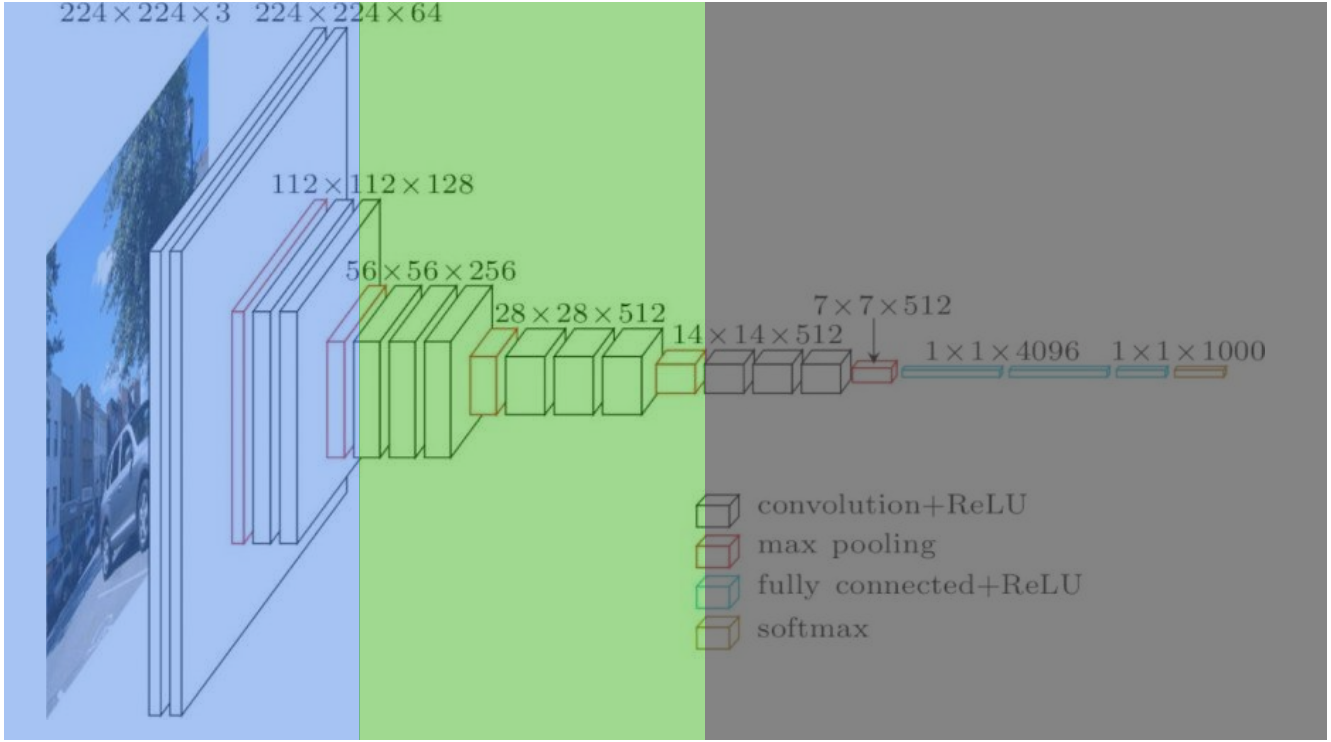
e	f	λ	A_{train}	A_{val}
18	18	0.0001	1.0	0.914
18	18	0.000154	0.995	0.9
18	18	0.000239	1.0	0.911
18	18	0.000368	1.0	0.908
18	18	0.000569	0.994	0.904
18	18	0.000879	0.998	0.911
18	18	0.00136	0.998	0.911
18	18	0.0021	0.994	0.908
18	18	0.00324	0.991	0.902
18	18	0.005	0.975	0.893
			0.995 ± 0.00713	0.906 ± 0.0061



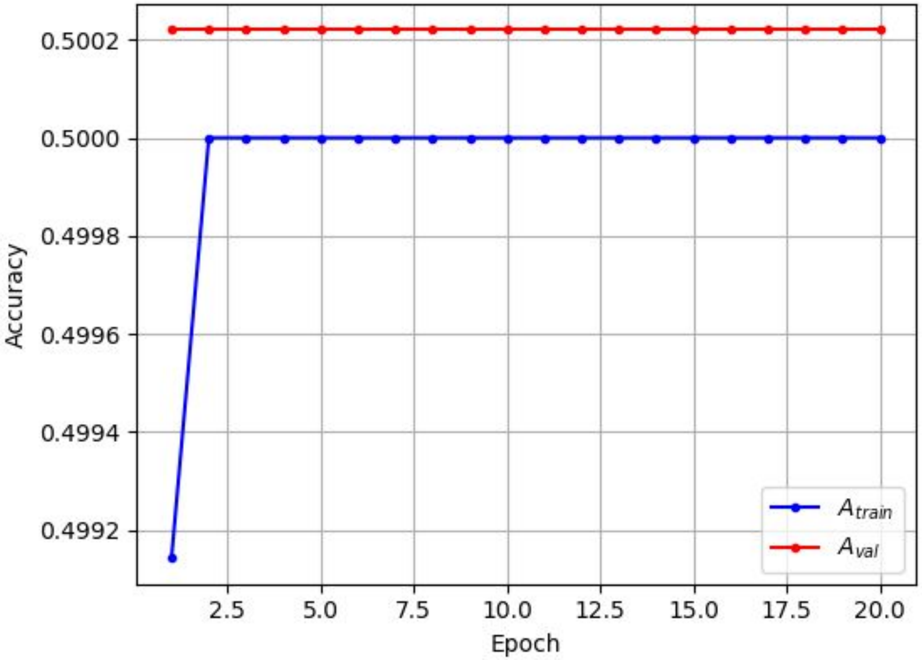
Data augmentation is very important

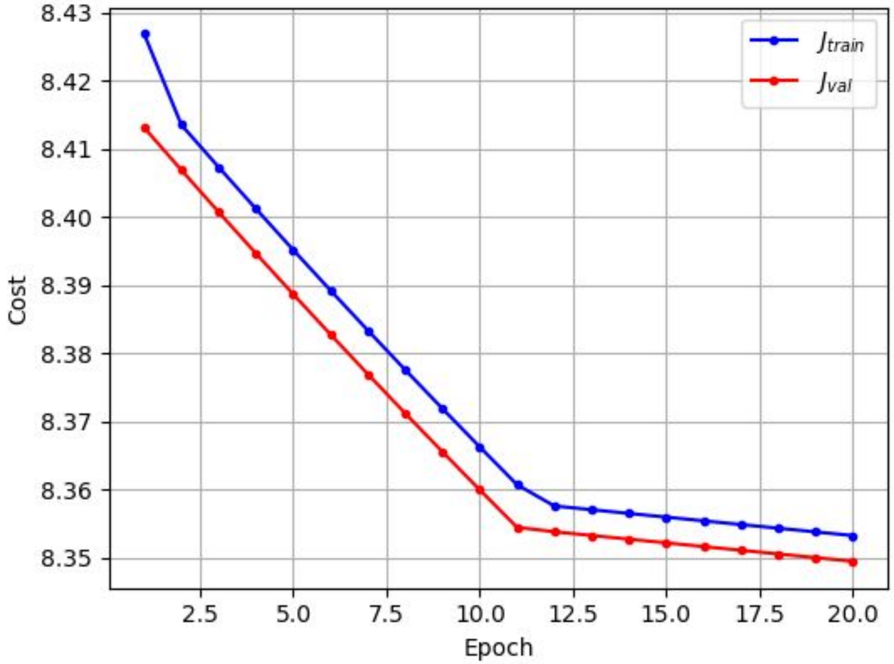


Partial Parameter Extraction w/ and w/o Finetuning

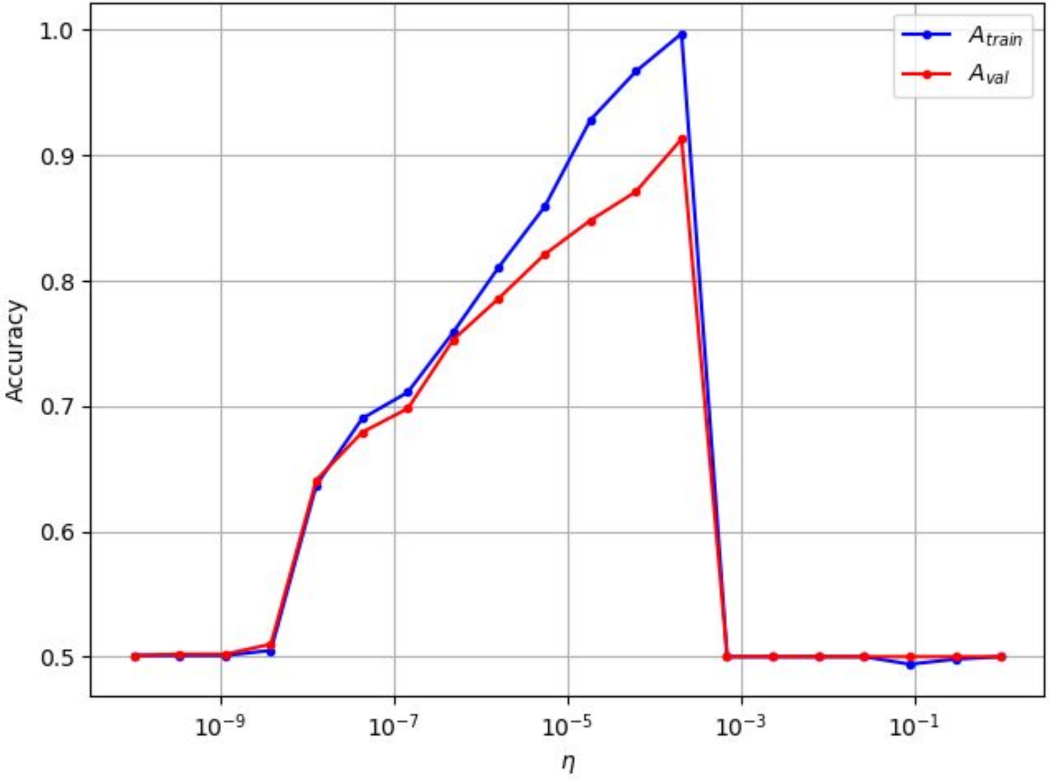


Models do not converge

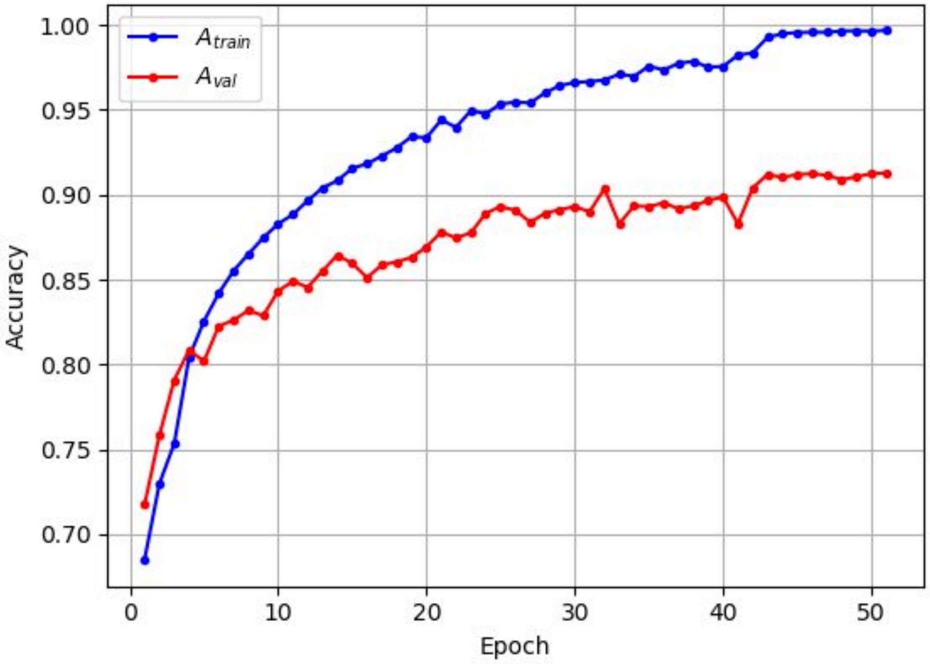


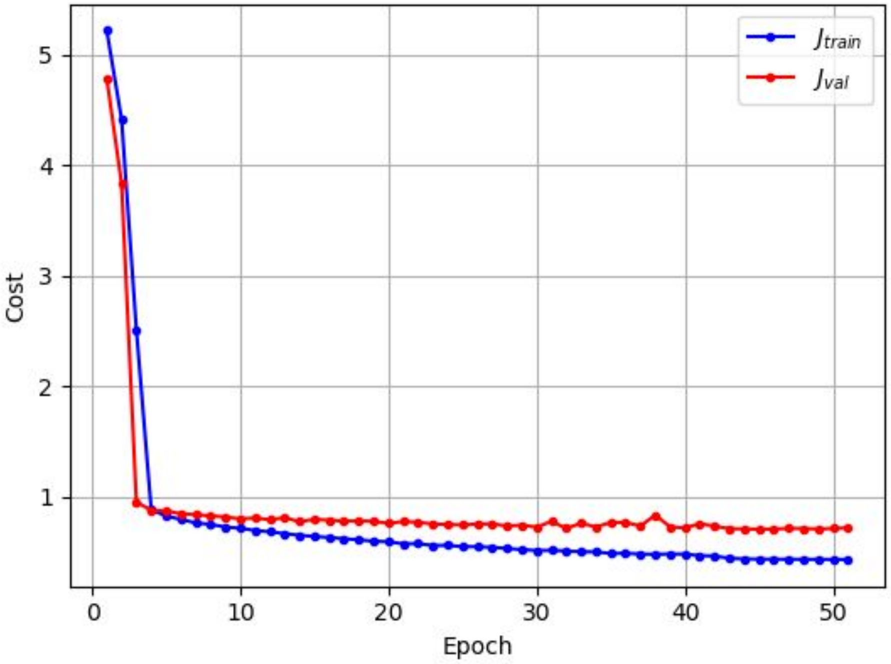


Cross-validating new learning rates

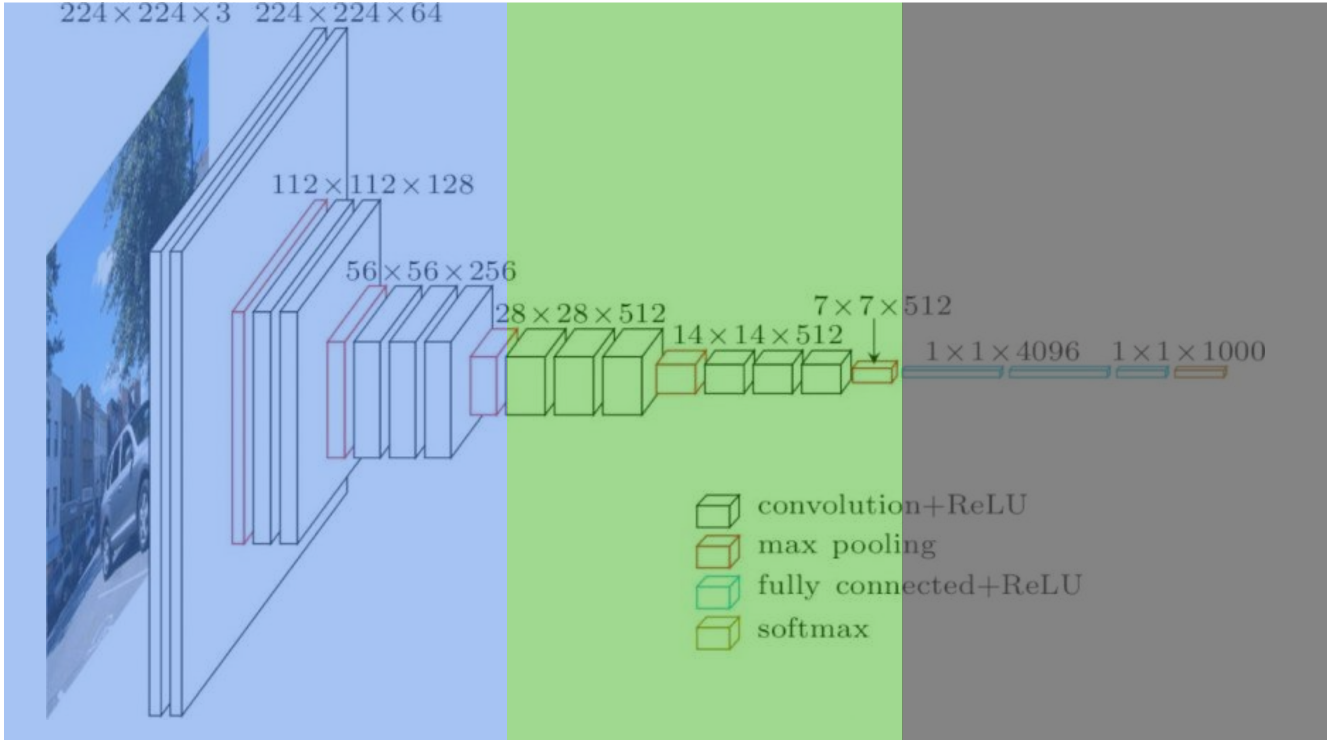


Appropriate learning rate allows convergence





Total Parameter Extraction with Finetuning



8% increase in performance

e	f	λ	A_{train}	A_{val}
18	14	0.0001	1.0	0.928
18	14	0.000154	1.0	0.923
18	14	0.000239	1.0	0.925
18	14	0.000368	1.0	0.926
18	14	0.000569	1.0	0.925
18	14	0.000879	1.0	0.921
18	14	0.00136	1.0	0.923
18	14	0.0021	1.0	0.927
18	14	0.00324	1.0	0.925
18	14	0.005	1.0	0.925
			1.0 ± 0.0	0.925 ± 0.00194

But as we unfreeze layers, models don't converge

e	f	λ	A_{train}	A_{val}
18	10	0.0001	0.5	0.5
18	10	0.000154	1.0	0.926
18	10	0.000239	1.0	0.928
18	10	0.000368	1.0	0.922
18	10	0.000569	1.0	0.933
18	10	0.000879	1.0	0.925
18	10	0.00136	1.0	0.935
18	10	0.0021	1.0	0.931
18	10	0.00324	1.0	0.929
18	10	0.005	1.0	0.929
			0.95 ± 0.15	0.886 ± 0.129

But as we unfreeze layers, models don't converge

e	f	λ	A_{train}	A_{val}
18	6	0.0001	0.5	0.5
18	6	0.000154	0.5	0.5
18	6	0.000239	1.0	0.93
18	6	0.000368	0.5	0.5
18	6	0.000569	1.0	0.925
18	6	0.000879	1.0	0.922
18	6	0.00136	0.5	0.501
18	6	0.0021	0.501	0.498
18	6	0.00324	1.0	0.917
18	6	0.005	1.0	0.922
			0.75 ± 0.25	0.711 ± 0.212

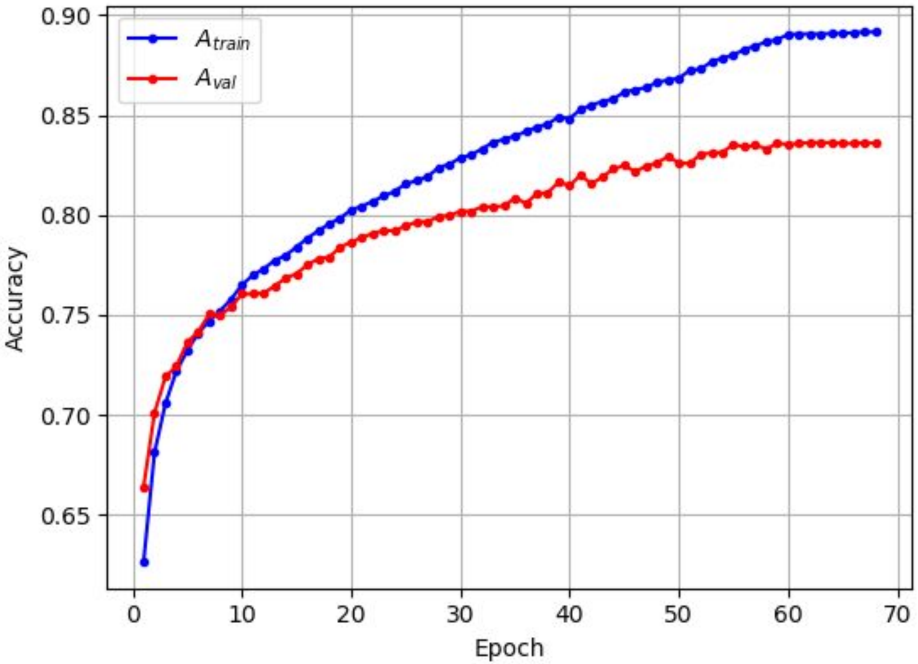
But as we unfreeze layers, models don’t converge

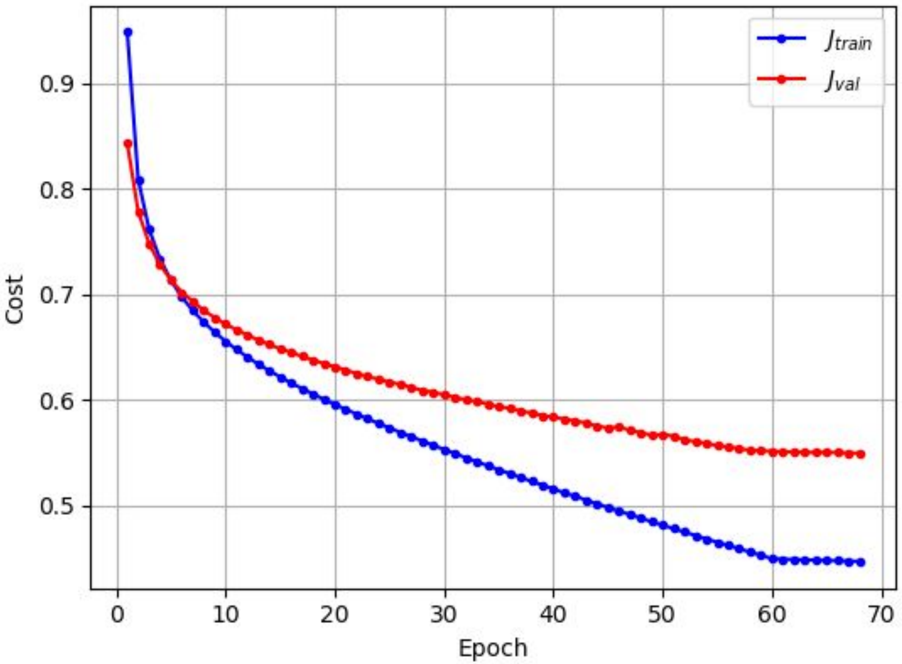
e	f	λ	A_{train}	A_{val}
18	3	0.0001	1.0	0.9
18	3	0.000154	0.5	0.5
18	3	0.000239	0.5	0.5
18	3	0.000368	1.0	0.915
18	3	0.000569	1.0	0.926
18	3	0.000879	1.0	0.927
18	3	0.00136	1.0	0.933
18	3	0.0021	1.0	0.931
18	3	0.00324	0.5	0.5
18	3	0.005	0.5	0.5
			0.8 ± 0.245	0.753 ± 0.207

But as we unfreeze layers, models don’t converge

e	f	λ	A_{train}	A_{val}
18	0	0.0001	0.5	0.5
18	0	0.000154	1.0	0.916
18	0	0.000239	1.0	0.9
18	0	0.000368	0.5	0.5
18	0	0.000569	0.5	0.5
18	0	0.000879	0.501	0.5
18	0	0.00136	0.5	0.5
18	0	0.0021	0.5	0.5
18	0	0.00324	0.5	0.5
18	0	0.005	0.5	0.5
			0.6 ± 0.2	0.582 ± 0.163

Adam optimizer allows such models to converge





Structure

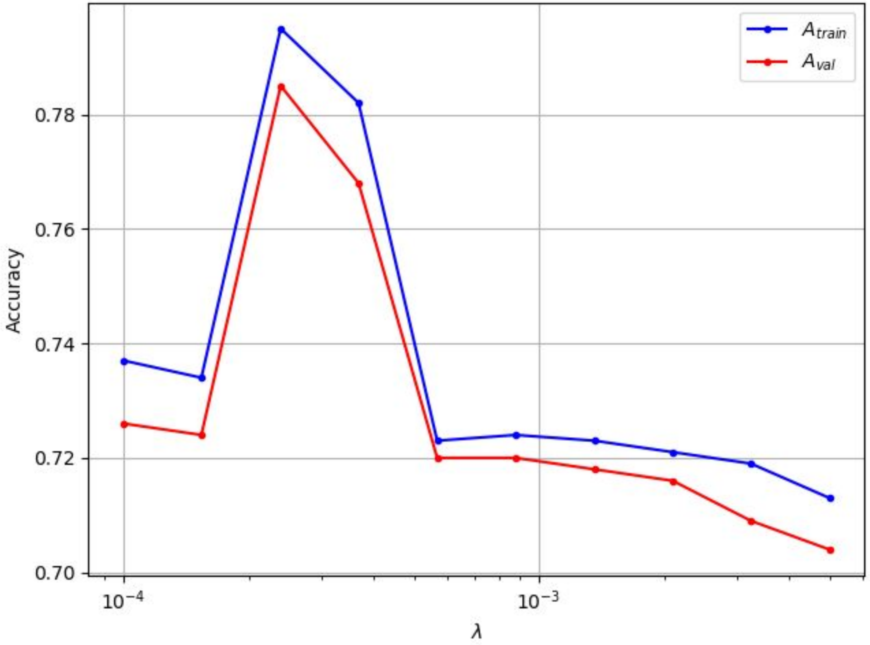
1. Motivation
2. Objectives
3. Background
4. Transfer Learning Experiments
5. **End-to-end Learning Experiments**
6. Conclusions

Custom Architecture 1

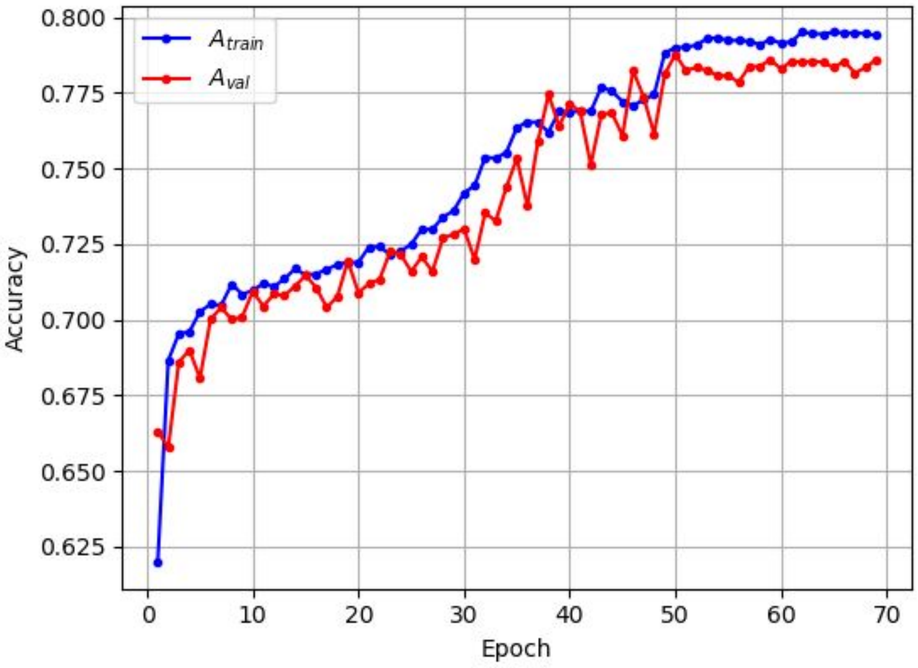
- 32 3×3 filters, stride of 1, zero padding, ReLU activated
- 32 3×3 filters, stride of 1, zero padding, ReLU activated
- 2×2 max pooling with stride 2
- 64 3×3 filters, stride of 1, zero padding, ReLU activated
- 64 3×3 filters, stride of 1, zero padding, ReLU activated
- 2×2 max pooling with stride 2
- 128 3×3 filters, stride of 1, zero padding, ReLU activated
- 128 3×3 filters, stride of 1, zero padding, ReLU activated
- 2×2 max pooling with stride 2
- 512 fully-connected ReLU-activated neurons
- 512 fully-connected ReLU-activated neurons
- 1 fully-connected sigmoid-activated neuron for binary classification

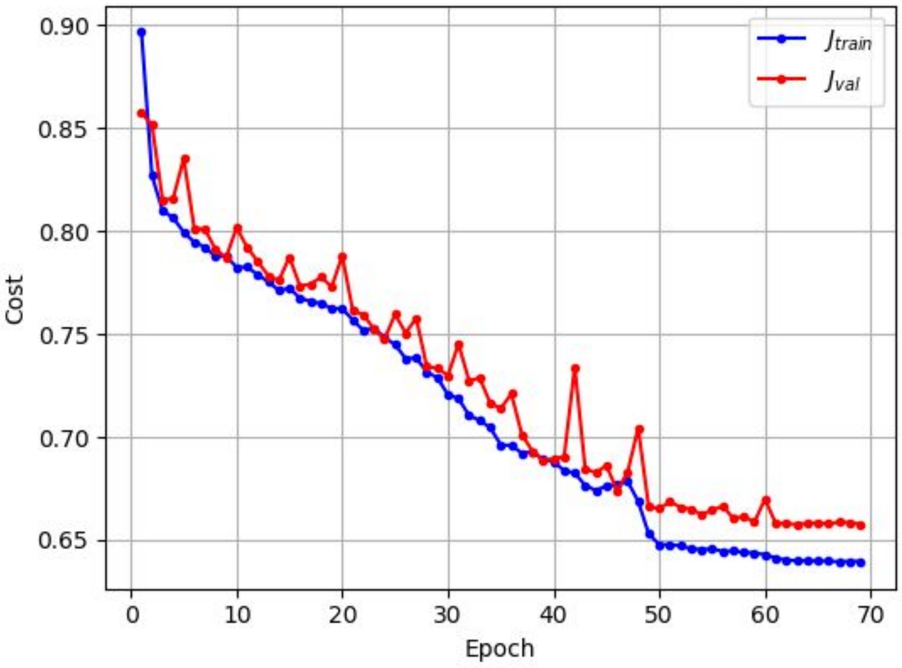
Custom Architecture 1

λ	A_{train}	A_{val}
0.0001	0.737	0.726
0.000154	0.734	0.724
0.000239	0.795	0.785
0.000368	0.782	0.768
0.000569	0.723	0.72
0.000879	0.724	0.72
0.00136	0.723	0.718
0.0021	0.721	0.716
0.00324	0.719	0.709
0.005	0.713	0.704
	0.737 ± 0.0267	0.729 ± 0.0248



Custom Architecture 1



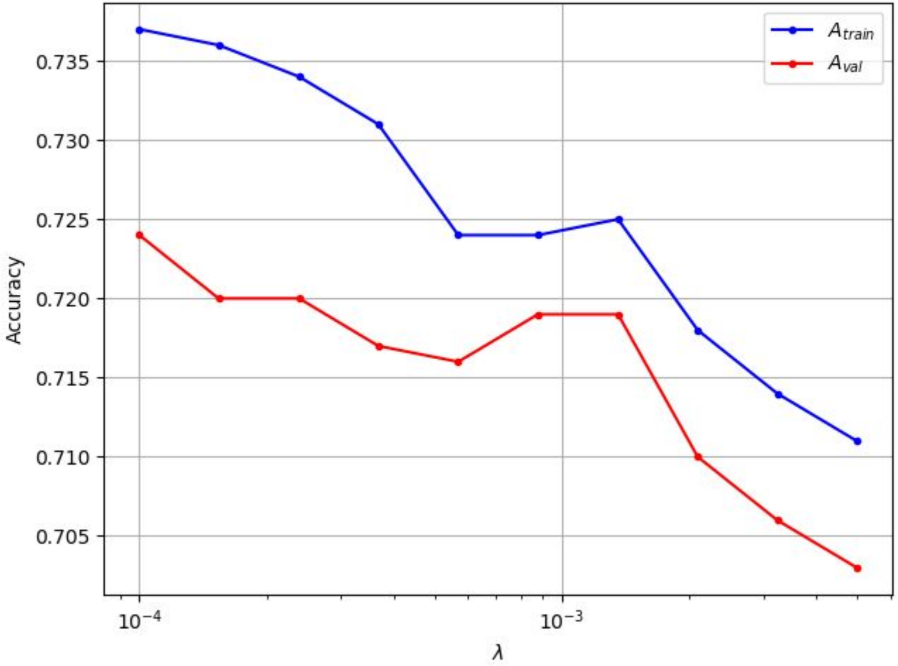


Custom Architecture 2

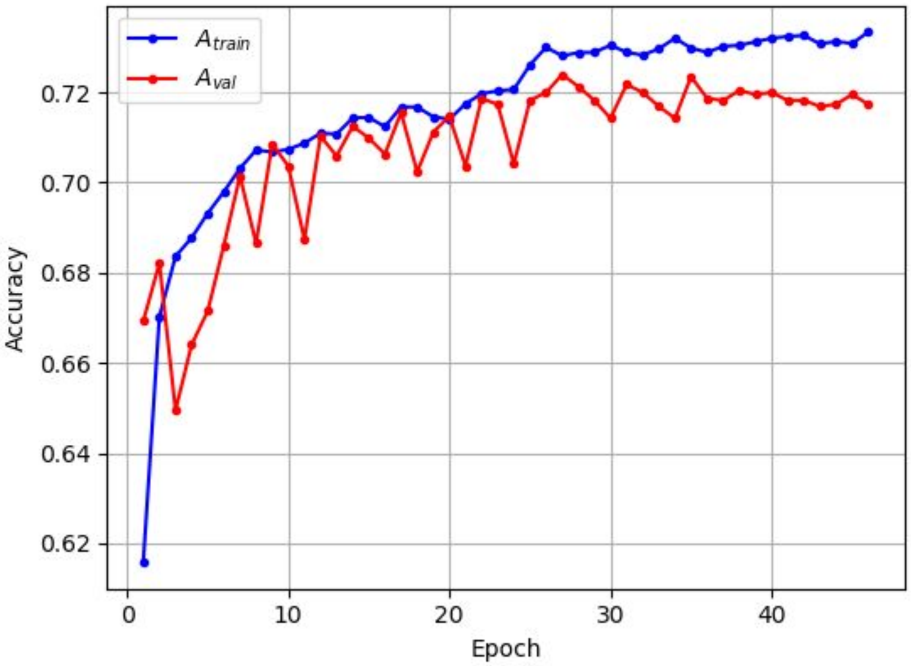
- 32 3×3 filters, stride of 1, zero padding, ReLU activated
- 2×2 max pooling with stride of 2
- 64 3×3 filters, stride of 1, zero padding, ReLU activated
- 2×2 max pooling with stride of 2
- 512 fully-connected ReLU-activated neurons
- 1 fully-connected sigmoid-activated neuron for binary classification

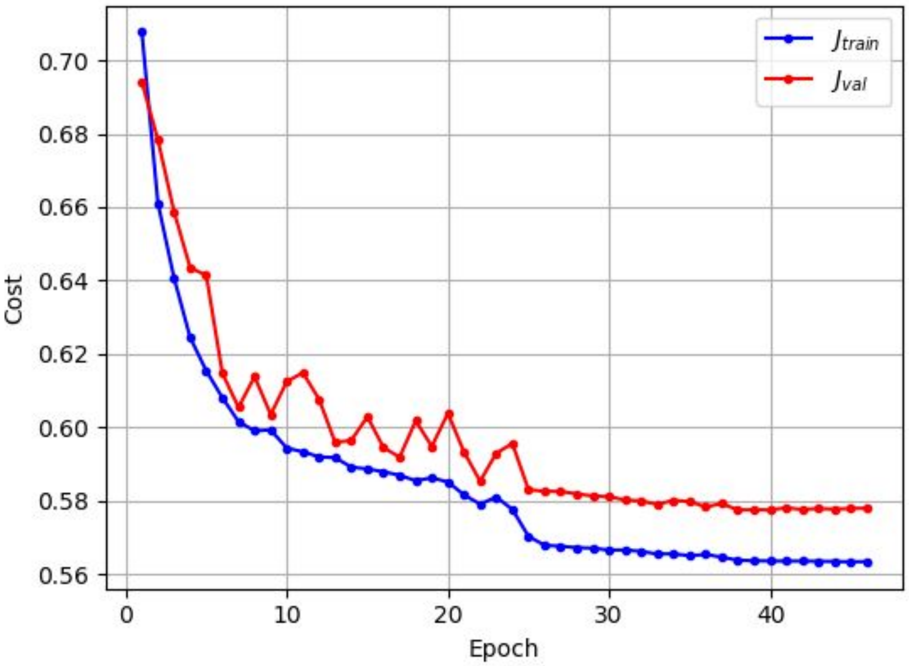
Custom Architecture 2

λ	A_{train}	A_{val}
0.0001	0.737	0.724
0.000154	0.736	0.72
0.000239	0.734	0.72
0.000368	0.731	0.717
0.000569	0.724	0.716
0.000879	0.724	0.719
0.00136	0.725	0.719
0.0021	0.718	0.71
0.00324	0.714	0.706
0.005	0.711	0.703
	0.725 ± 0.00865	0.715 ± 0.00645

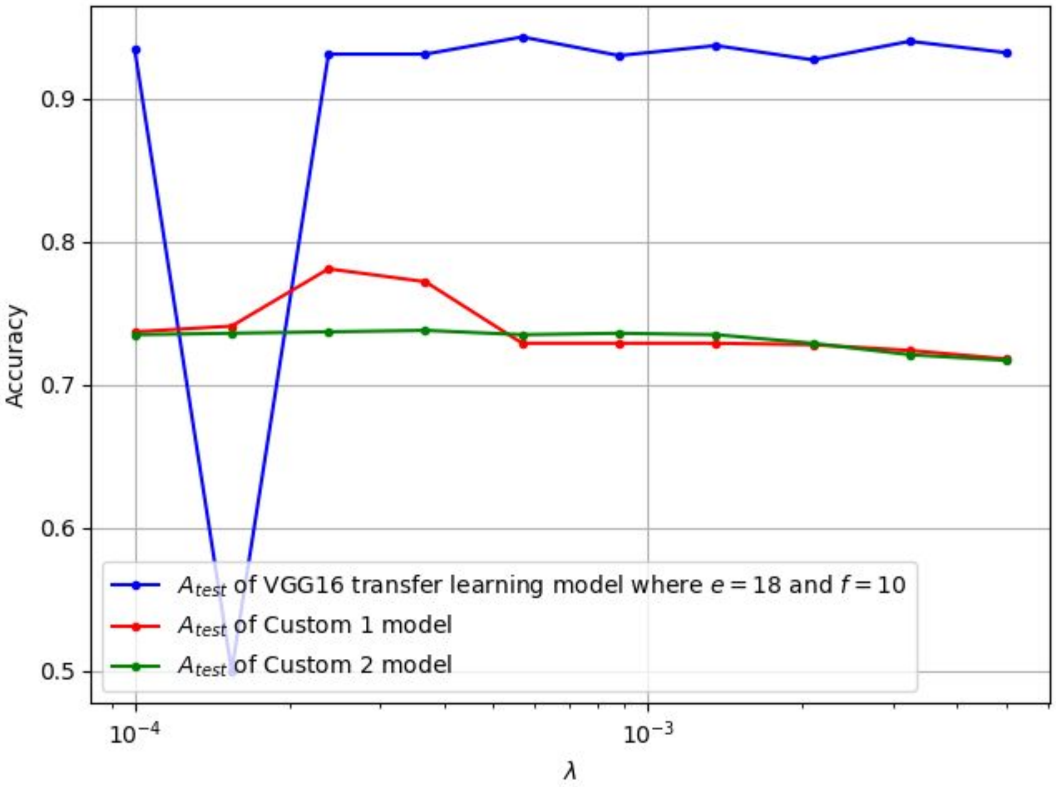


Custom Architecture 2





Comparison



Accuracy
0.9
0.8
0.7
0.6
0.5
10^{-4}
10^{-3}
λ
A_{test} of VGG16 transfer learning model where $e = 18$ and $f = 10$
A_{test} of Custom 1 model
A_{test} of Custom 2 model

Comparison

Model	A_{test}	AUC	P	R	F_1
Custom 2 $\lambda = 0.0003684$	0.738	0.737	0.761	0.738	0.732
Custom 1 $\lambda = 0.00023853$	0.781	0.781	0.785	0.781	0.78
VGG16 $e = 18,\ f = 10,\ \lambda = 0.00056898$	0.943	0.943	0.945	0.943	0.943

Structure

1. Motivation
2. Objectives
3. Background
4. Transfer Learning Experiments
5. End-to-end Learning Experiments
6. **Conclusions**

Takeaways

- Appropriate and reasonable data augmentation techniques are essential;
- Parameters from lower layers seem to be more sensitive to overshooting updates when compared to higher layers. Adaptive optimizers or carefully set learning rates solve the problem.
- Extracting only a subset of parameters from a pre-trained model can simultaneously provide good generalization performance as well as a more compact and computationally efficient model for certain applications;
- Extracting parameters up to the highest layer and fine-tuning the parameters up to the middle layers is the strategy that yielded the best generalization performance, a result which can be taken as a simple guiding heuristic;
- Designing a custom CNN and training it from scratch is difficult because it requires reasoning about and cross-validating many hyperparameters simultaneously
- Transfer learning emerges as the clear practical solution to a lot of problems where data is scarce

Future Work

- Study transfer learning applied to skin lesion classification with respect to more recent, state-of-the-art architectures;
- Study of optimizers and learning rates in transfer learning applied to skin lesion classification, perhaps even developing optimizers or learning rate schedules specific to transfer learning.

Thank you