ارائه نهایی درس شبکه عصبی

عنوان : شبکه های کانولوشنی

Convolutional Neural Networks

ارائه دهنده : محمد سینا حسن نیا

استاد درس : دکتر عسکری

تاریخ : ۱۴۰۳/۰۴/۱۴

Outline

- Definition & History
- Why not just use an MLP for images?
- CNN Architecture
- Convolutional Layer
- Non-linear Layer
- Pooling layers
- Fully Connected Layer
- Casdcsdcsdcsd
- V fvsdvsfvfsvfd
- References

Definition & History

- A convolutional neural network (CNN) is a regularized type of feed-forward neural network that learns relevant features from images, being capable of performing several tasks like object classification, detection, and segmentation.

- The modern concept of Convolutional Neural Networks comes from the work of Yann LeCun published in 1998. LeCun proposed a CNN called LeNet for hand-write recognition.

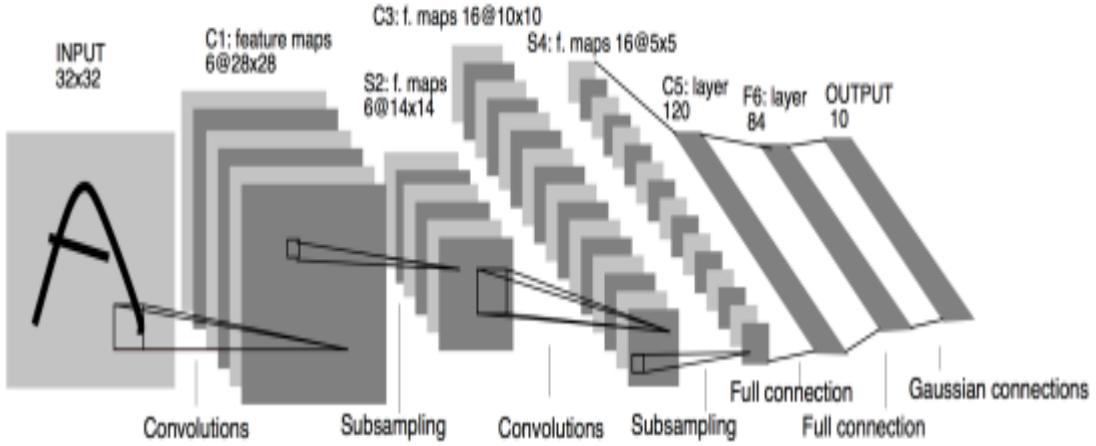


Fig 1. Old CNN (LeNet)

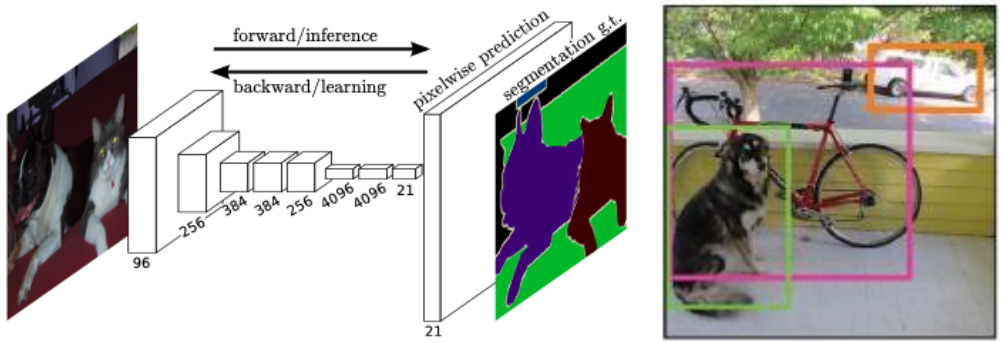


Fig 2. Applications of CNNs

Why not just use an MLP for images?

- MLP connects each pixel in an image to each neuron and suffers from the curse of dimensionality, so it does not scale well to higher resolution images

- Local Connectivity
  - In FC layers, every output unit interacts with every input unit.
  - Store fewer parameters ➔ Sparse Connection

- Parameter Sharing
  - In a convolutional neural network, each member
  - of the kernel is used at every position of the input

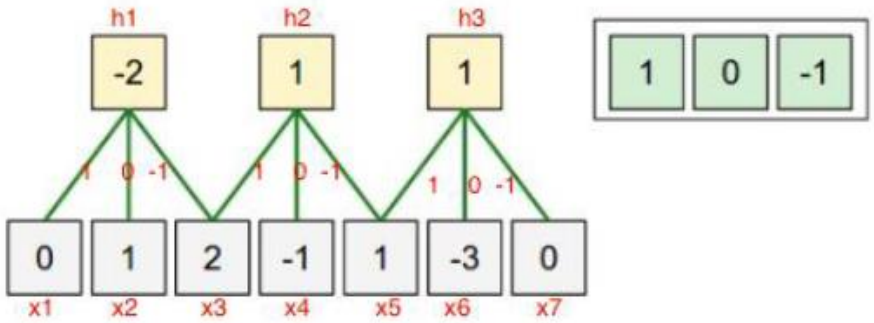


Fig 3. Parameter sharing

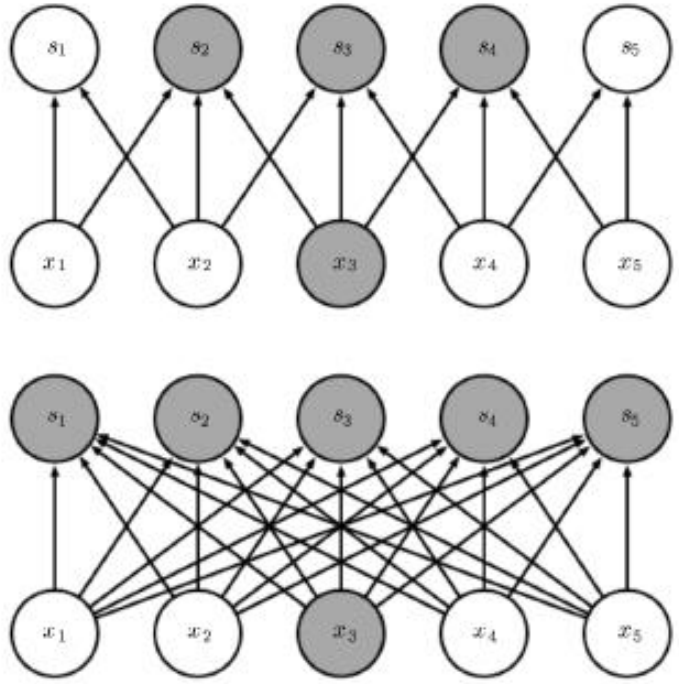


Fig 4. Local connectivity

CNN Architecture

- A CNN is composed by four types of layers:
  - Convolutional layers
  - Pooling layers
  - Non-linear layers
  - Fully Connected layers

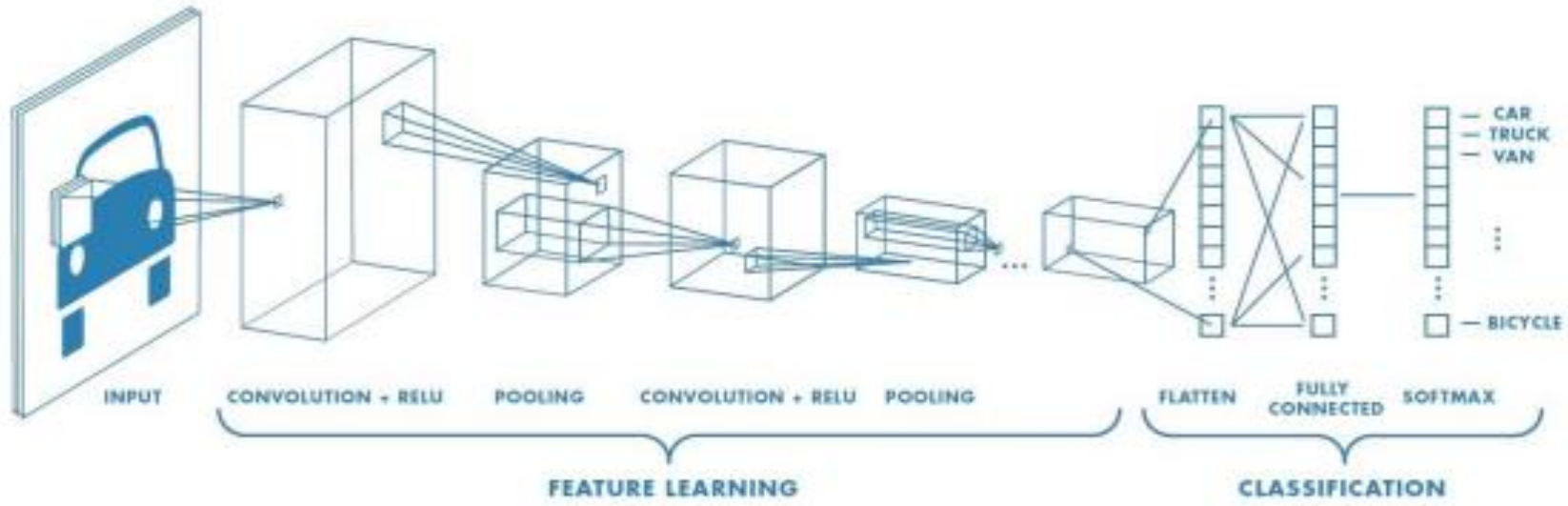


Fig 5. CNN Architecture

Convolutional Layer

- A Cconvolutional Layer is composed by a set of filters, also called kernels, that slides over the input data
- Each kernel has a width, a height and width × height weights utilized to extract features from the input.
- In the training step, the weights in the kernel starts with random values, and will be learning based on the training set.
- Kernel Effect:

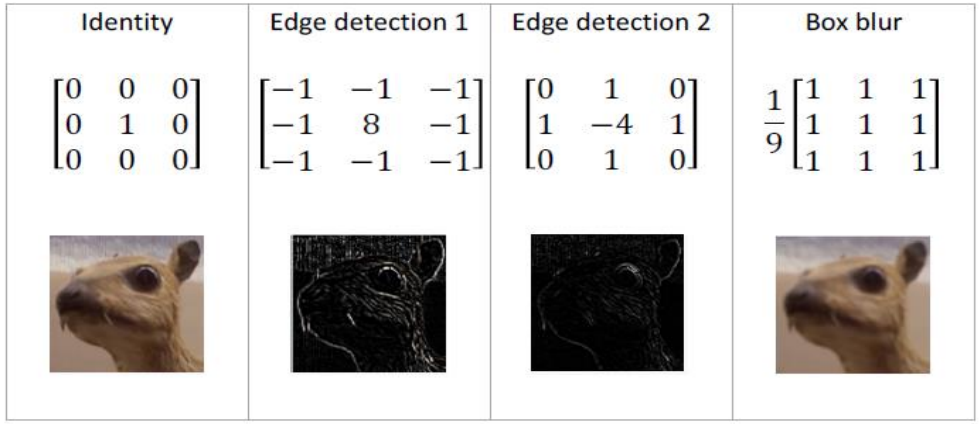


Fig 6. Kernel Effect

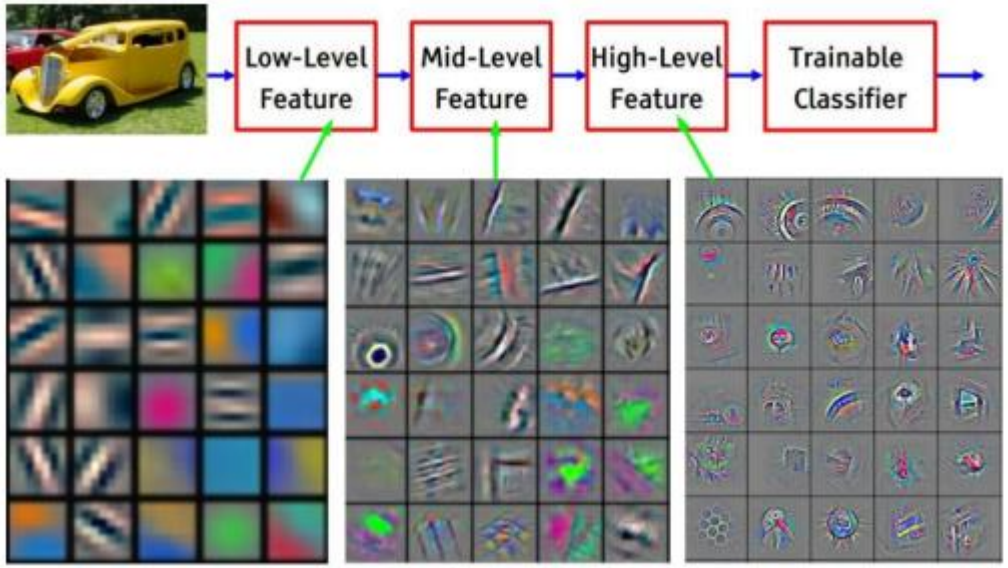


Fig 7.Features

Non-linear Layer

- Tthis layer, is an activation layer linked after a convolutional layer to generate non-linearity in the network.

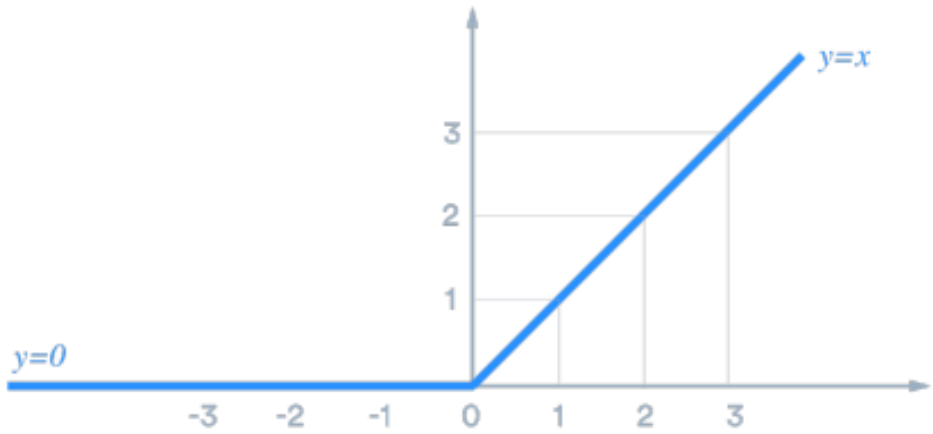


Fig 7.Example of Non-linear function

Pooling layers

- The pooling layer, or down-sampling layer, is applied to reduce the dimensionality of the feature maps in a way to save the most relevant information from the feature maps.
- In the pooling layer, a filter slides over the input data and applies the pooling operation (max, min, avg)
- The max pooling is the most used in the literature.
- Pooling increases the generalizability of a model.
  - Dimensionality Reduction
  - Robustness to Noise and Distortion

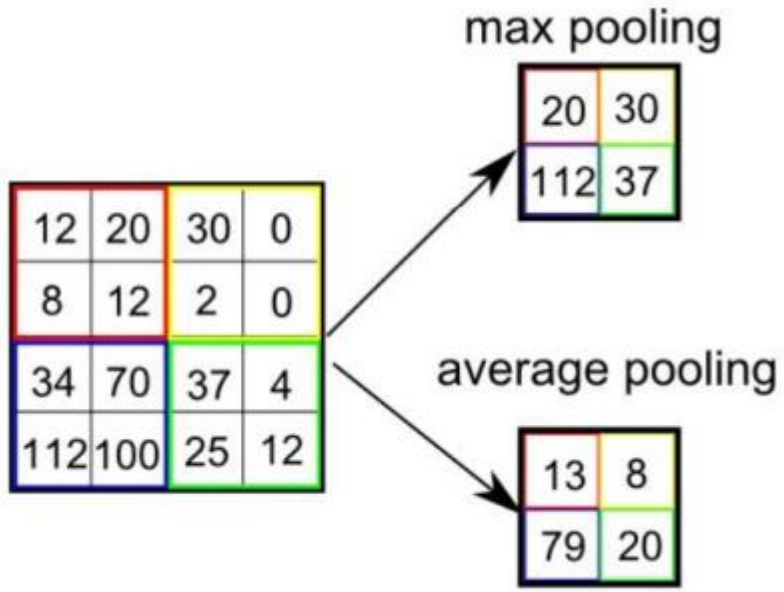


Fig 9.Max pooling and average Pooling

Fully Connected Layer

- A CNN is tyapically divided into two parts:
-
  - the convolutional steps: Learns the best features to extract from the images
  - Dense steps: Learns how to classify the features in different categories.
- The Fully Connected layer is a MLP, composed by three types of layers: input, hidden, and output layer
- The input layer receives the features generated by the CNN .
- The hidden layer is a sequence of neurons with weights that will be learned in the training step.
- The output layer is also a sequence of neurons. However, it has a different activation function.
  - Usually, the softmax function is used to generate the probabilities of each category in the problem scope.

ResNet

- Problems with Deeper Networks:
  - 56-layer model performs worse on both training and test error
  - It is not caused by overfitting

- This is a optimization problem, deeper models are harder to optimize

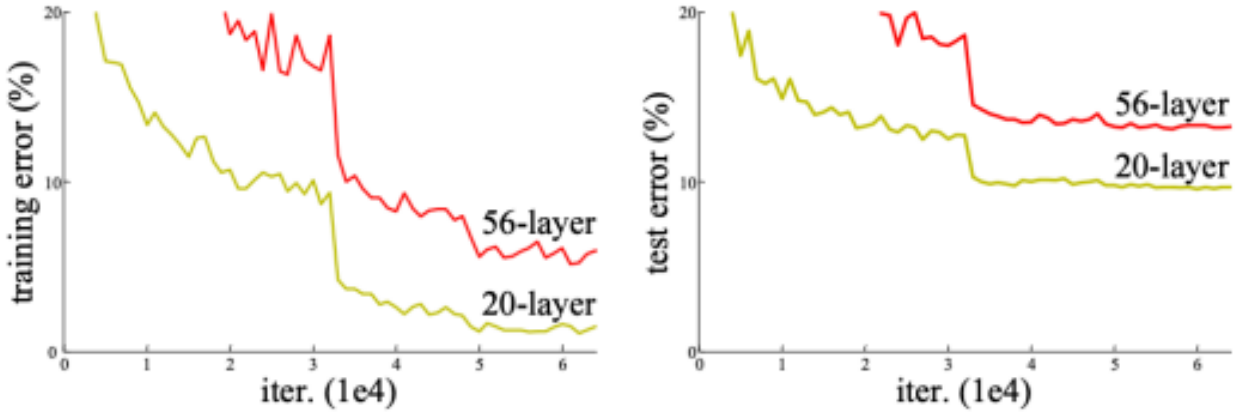


Fig 10. Training and Validation error for two different net works

ResNet

- Problems with Deeper Networks:
  - 56-layer model performs worse on both training and test error
  - It is not caused by overfitting

- This is a optimization problem, deeper models are harder to optimize

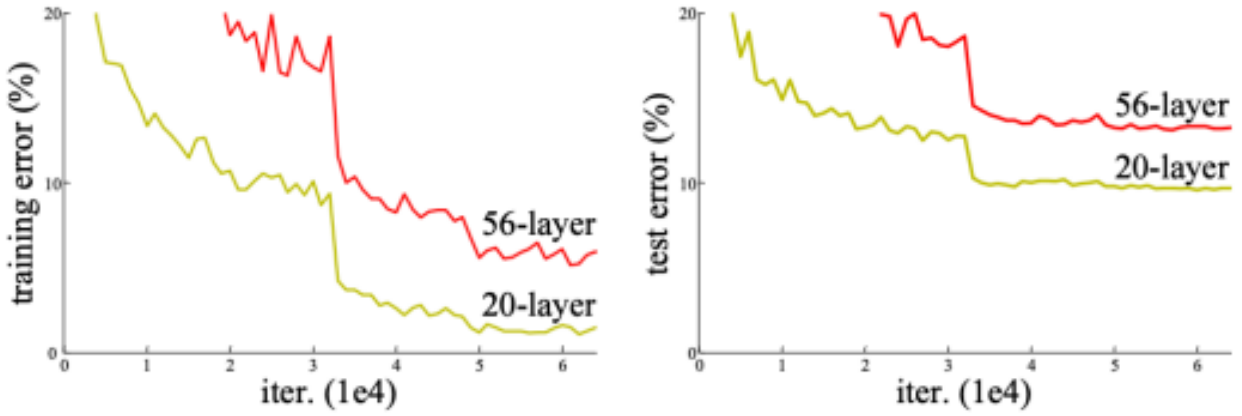


Fig 10. Training and Validation error for two different net works

ResNet

- Aiming to build deeper neural networks:
  - Idea: New Block (***Shortcut Connection*/*Skip Connections***)

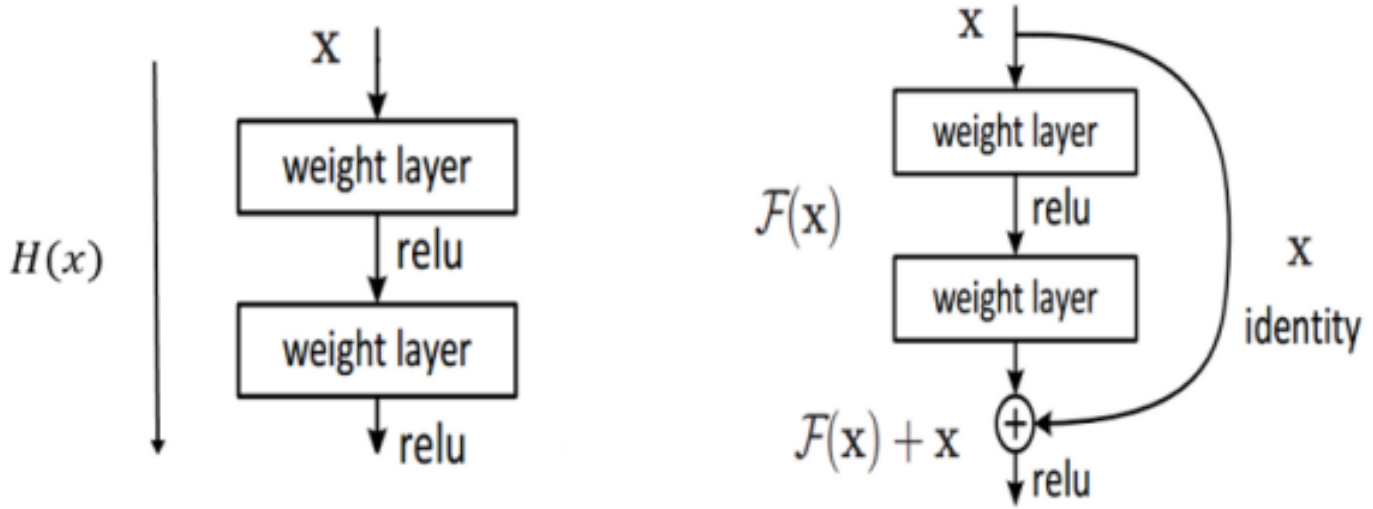


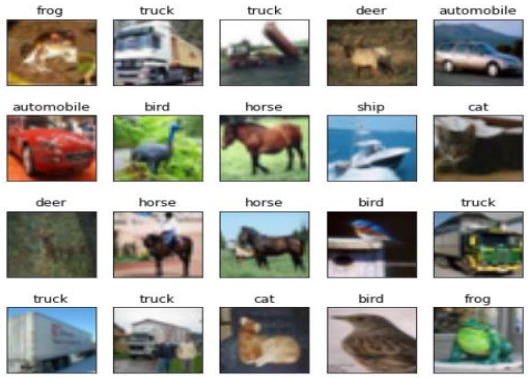
Fig 11. : Difference between a convolution block without skip and with skip Connection

Example

Multi-Layer Perceptron & Convolutional Neural Network on CIFAR-10 dataset

General Information

1- Dataset consists of 60×10^3 images
2- 50×10^3 images are used for training
3- 10×10^3 images are used for validation



First hyperparameters

Layers: 3 (L1: 256 , L2 = 128)
Batch size: 64
Epochs: 15
Algorithm: SGD
Loss Function: Cross Entropy

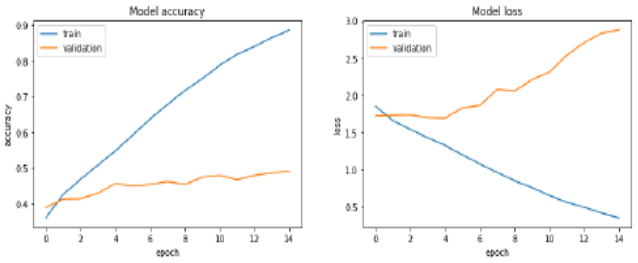
Pre-processing

1-Make the images normal using mean & std
2- One-hot the labels

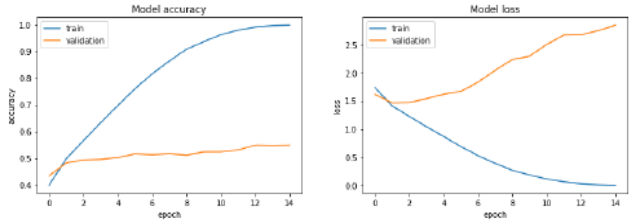
MLP hyperparameters

Batch size

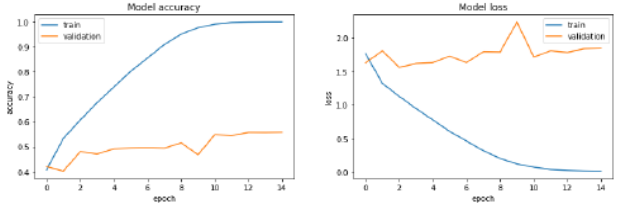
32



64

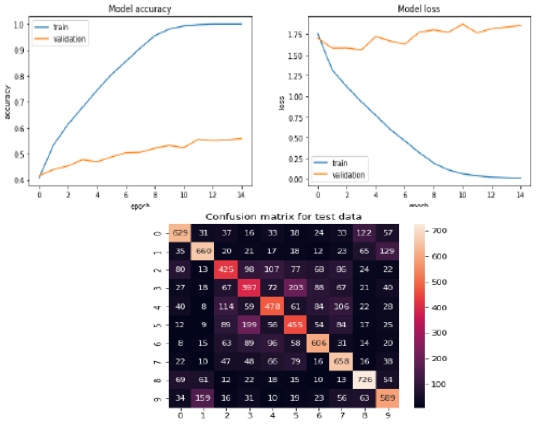


256

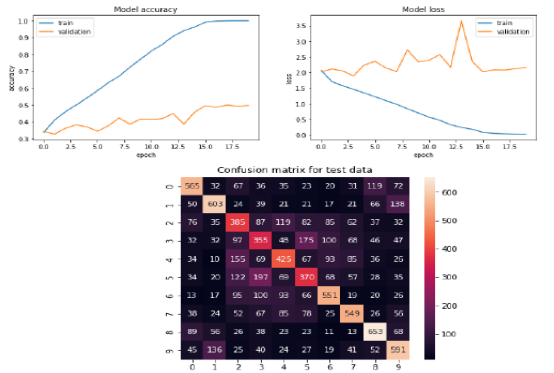


Activation Function

ReLU

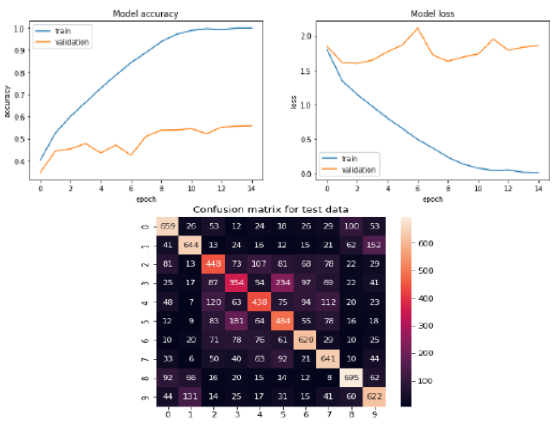


TanH

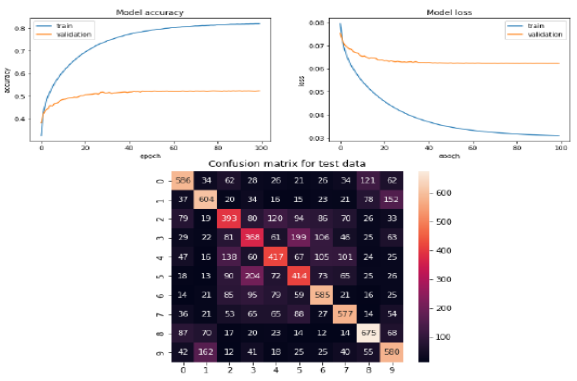


Loss Function

Categorical



MSE

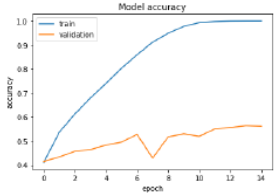


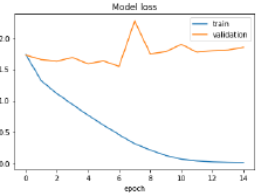
Example

Multi-Layer Perceptron & Convolutional Neural Network on CIFAR-10 dataset

Best MLP

Layers: 3(L1: 2048 , L2 = 2048)
Batch size: 256
Epochs: 15
Algorithm: SGD + momentum
Loss Function: Cross Entropy



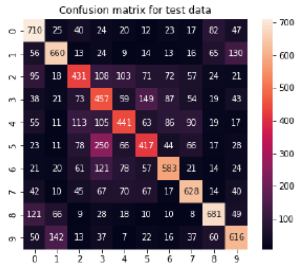



```
The Number of Epochs: 15
Elapsed Time: 28.9s

Train Results:
 - loss: 0.0159
 - accuracy: 0.9997

Validation Results:
 - loss: 1.8485
 - accuracy: 0.5619

Test Results:
 - loss: 1.8312
 - accuracy: 0.5624
```

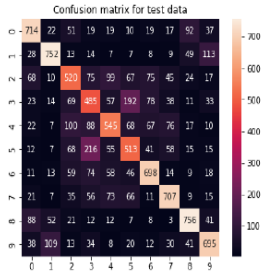


	precision	recall	f1-score	support
airplane	0.6724	0.5050	0.5768	1000
automobile	0.6630	0.6590	0.6610	1000
bird	0.5126	0.2840	0.3655	1000
cat	0.3762	0.3980	0.3868	1000
deer	0.4546	0.5210	0.4856	1000
dog	0.4309	0.4990	0.4625	1000
frog	0.5747	0.6270	0.5997	1000
horse	0.6253	0.6090	0.6170	1000
ship	0.6084	0.7380	0.6670	1000
truck	0.5617	0.5960	0.5784	1000
accuracy			0.5436	10000
macro avg	0.5480	0.5436	0.5400	10000
weighted avg	0.5480	0.5436	0.5400	10000

CNN

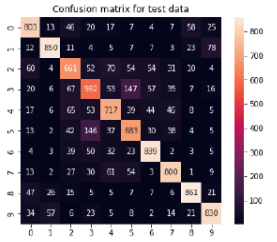
Fist CNN

	precision	recall	f1-score	support
airplane	0.6966	0.7140	0.7052	1000
automobile	0.7573	0.7520	0.7546	1000
bird	0.5479	0.5200	0.5336	1000
cat	0.4520	0.4850	0.4679	1000
deer	0.5841	0.5450	0.5639	1000
dog	0.5151	0.5130	0.5140	1000
frog	0.6863	0.6980	0.6921	1000
horse	0.7091	0.7070	0.7081	1000
ship	0.7390	0.7560	0.7474	1000
truck	0.6992	0.6950	0.6971	1000
accuracy			0.6385	10000
macro avg	0.6387	0.6385	0.6384	10000
weighted avg	0.6387	0.6385	0.6384	10000



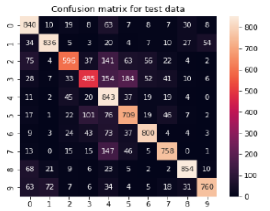
Second CNN: Fist + pooling + Batch normalization

	precision	recall	f1-score	support
airplane	0.7849	0.8030	0.7939	1000
automobile	0.8772	0.8500	0.8634	1000
bird	0.6752	0.6610	0.6680	1000
cat	0.6072	0.5920	0.5995	1000
deer	0.7156	0.7170	0.7163	1000
dog	0.6638	0.6830	0.6732	1000
frog	0.8013	0.8390	0.8197	1000
horse	0.8147	0.8000	0.8073	1000
ship	0.8645	0.8610	0.8627	1000
truck	0.8317	0.8300	0.8308	1000
accuracy			0.7636	10000
macro avg	0.7636	0.7636	0.7635	10000
weighted avg	0.7636	0.7636	0.7635	10000



Third CNN: Second + Dropout

	precision	recall	f1-score	support
airplane	0.7254	0.8400	0.7785	1000
automobile	0.8745	0.8360	0.8548	1000
bird	0.7690	0.5960	0.6715	1000
cat	0.6699	0.4850	0.5626	1000
deer	0.5356	0.8430	0.6550	1000
dog	0.6469	0.7090	0.6765	1000
frog	0.8222	0.8000	0.8109	1000
horse	0.8177	0.7580	0.7867	1000
ship	0.8795	0.8540	0.8666	1000
truck	0.8983	0.7600	0.8234	1000
accuracy			0.7481	10000
macro avg	0.7639	0.7481	0.7487	10000
weighted avg	0.7639	0.7481	0.7487	10000

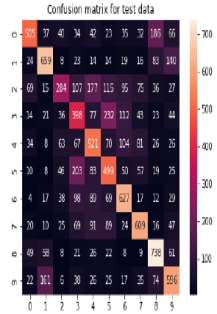


Imbalanced Dataset

Downsampling

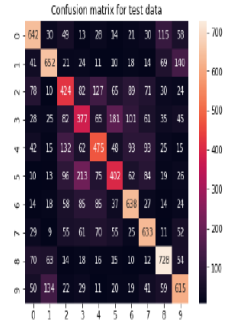
Before

	precision	recall	f1-score	support
airplane	0.6724	0.5050	0.5768	1000
automobile	0.6630	0.6590	0.6610	1000
bird	0.5126	0.2840	0.3655	1000
cat	0.3762	0.3980	0.3868	1000
deer	0.4546	0.5210	0.4856	1000
dog	0.4309	0.4990	0.4625	1000
frog	0.5747	0.6270	0.5997	1000
horse	0.6253	0.6090	0.6170	1000
ship	0.6084	0.7380	0.6670	1000
truck	0.5617	0.5960	0.5784	1000
accuracy			0.5436	10000
macro avg	0.5480	0.5436	0.5400	10000
weighted avg	0.5480	0.5436	0.5400	10000



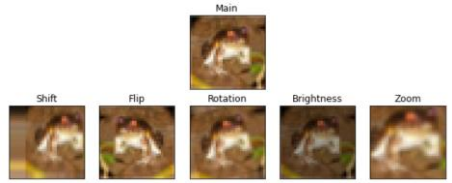
After

	precision	recall	f1-score	support
airplane	0.6394	0.6420	0.6407	1000
automobile	0.6729	0.6520	0.6623	1000
bird	0.4449	0.4240	0.4342	1000
cat	0.3911	0.3770	0.3839	1000
deer	0.4933	0.4750	0.4840	1000
dog	0.4746	0.4020	0.4353	1000
frog	0.5929	0.6380	0.6146	1000
horse	0.5938	0.6330	0.6128	1000
ship	0.6588	0.7280	0.6917	1000
truck	0.5840	0.6150	0.5991	1000
accuracy			0.5586	10000
macro avg	0.5546	0.5586	0.5559	10000
weighted avg	0.5546	0.5586	0.5559	10000

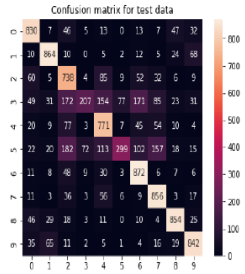


Data Augmentation

Before

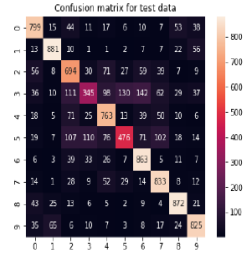


	precision	recall	f1-score	support
airplane	0.7587	0.8300	0.7927	1000
automobile	0.8300	0.8640	0.8466	1000
bird	0.5516	0.7380	0.6313	1000
cat	0.6721	0.2070	0.3165	1000
deer	0.6203	0.7710	0.6875	1000
dog	0.7401	0.2990	0.4259	1000
frog	0.6760	0.8720	0.7616	1000
horse	0.7005	0.8560	0.7705	1000
ship	0.8447	0.8540	0.8493	1000
truck	0.8027	0.8420	0.8219	1000
accuracy			0.7133	10000
macro avg	0.7197	0.7133	0.6904	10000
weighted avg	0.7197	0.7133	0.6904	10000



After

	precision	recall	f1-score	support
airplane	0.7690	0.7990	0.7837	1000
automobile	0.8637	0.8810	0.8723	1000
bird	0.6180	0.6940	0.6538	1000
cat	0.5948	0.3450	0.4367	1000
deer	0.6837	0.7630	0.7212	1000
dog	0.6849	0.4760	0.5617	1000
frog	0.7062	0.8630	0.7768	1000
horse	0.7398	0.8330	0.7836	1000
ship	0.8273	0.8720	0.8491	1000
truck	0.8049	0.8250	0.8148	1000
accuracy			0.7351	10000
macro avg	0.7292	0.7351	0.7254	10000
weighted avg	0.7292	0.7351	0.7254	10000



References

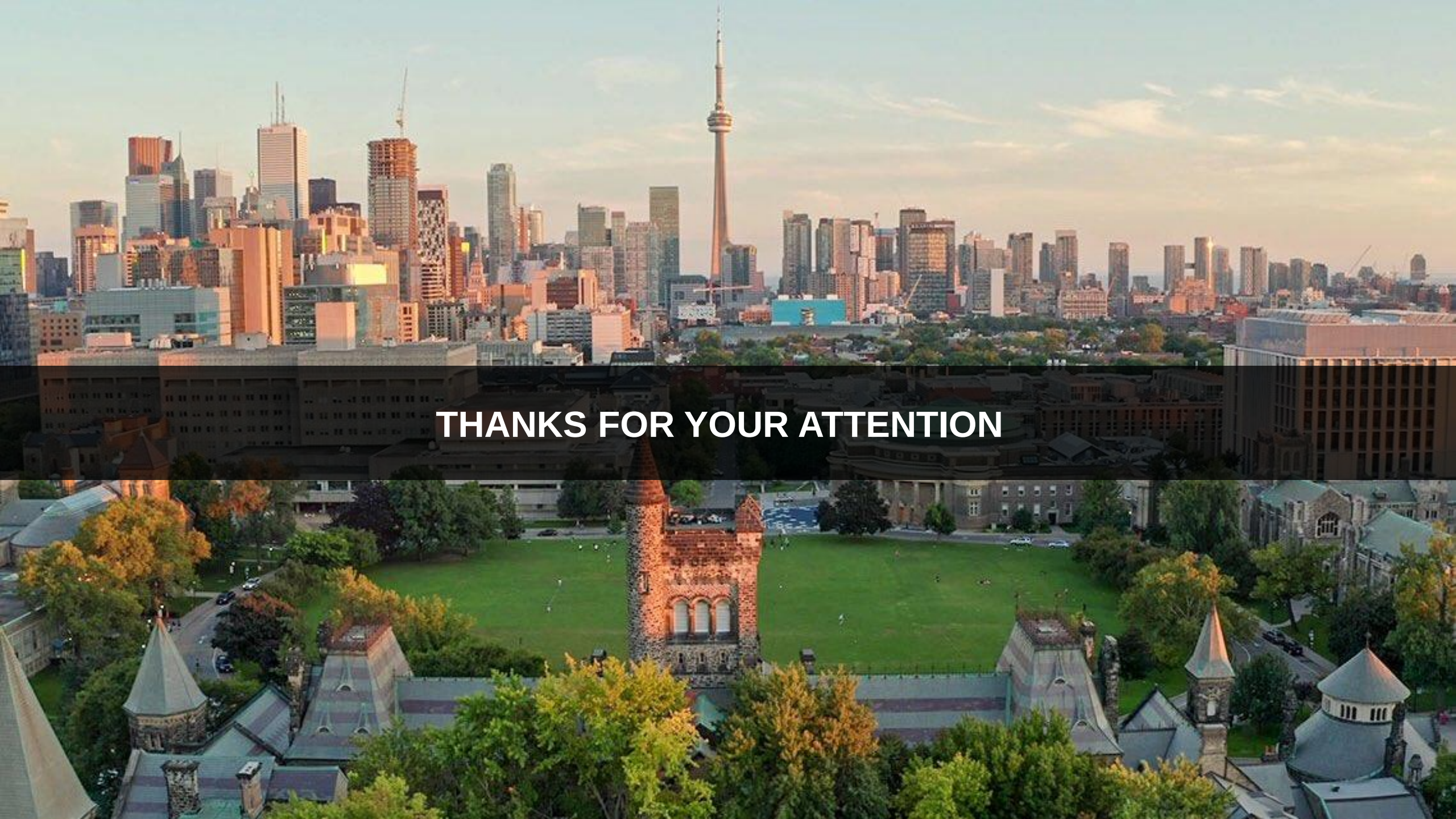
• Gradient-Based Learning Applied to Document Recognition, LeCun, Bottou, Bengio, Haffner, IEEE 1998

• ImageNet Classification with Deep Convolutional Neural Networks, Krizhevsky, Sutskever, Hinton, NIPS 2012

• Very Deep Convolutional Networks For Large-Scale Image Recognition, Simonyan, Zisserman, ICLR 2015

• Deep Residual Learning for Image Recognition, He, Zhang, Ren, and Sun, CVPR 2015

• Good Practices for Deep Feature Fusion, Shao et al. 2016



THANKS FOR YOUR ATTENTION