Implement Classification with PyTorch

Tutorial of EE4146

TA: GUO Xiaoqing
Email: xiaoqiguo2-c@my.cityu.edu.hk

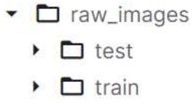
Department of Electrical Engineering
City University of Hong Kong

10/11/2021

General Pipeline

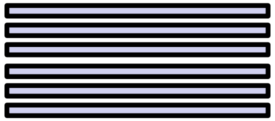
Raw Images

N × 3 ×W ×H



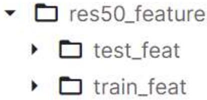
- raw_images
  - test
  - train

Feature Extractor

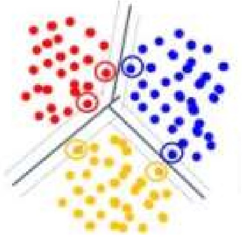


Feature Vectors

N × D (2048)



- res50_features
  - test_feat
  - train_feat



Classification Algorithms

buildings，
forest，
glacier，
mountain，
sea，
street？

Classification Results

N × 1

- Better features
  - Use some tricks, e.g. dimensionality reduction
  - Use stronger feature extractors (VGG/ ResNet/ DenseNet…)

- Better Classifiers
  - Linear Classifiers
    - SVM
    - Logistic Regression …
  - Non-linear Classifier:
    - Kernel-based methods
    - Boosting: Ada boosting…
    - Bagging: Random forest…

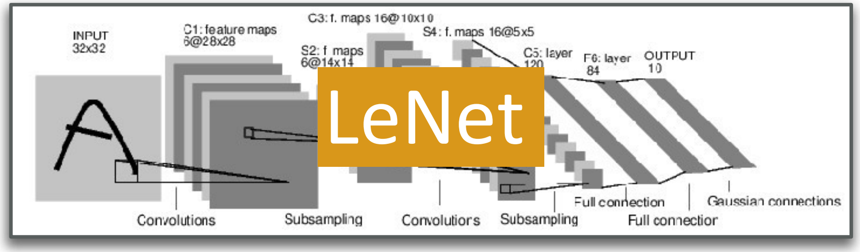
Outline

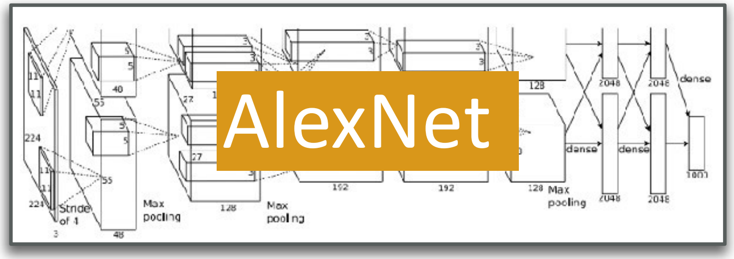
- ➢ Classification Network (DenseNet)
- ➢ Implement DenseNet with Pytorch
- ➢ Further Extensions
  - Data augmentation
  - Attention Mechanism
  - Contrastive Learning

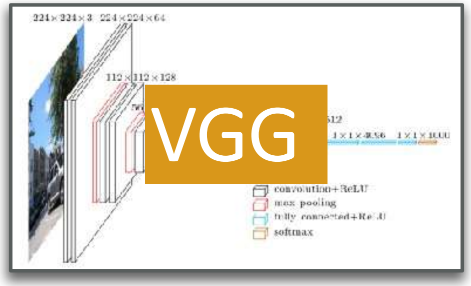
Outline

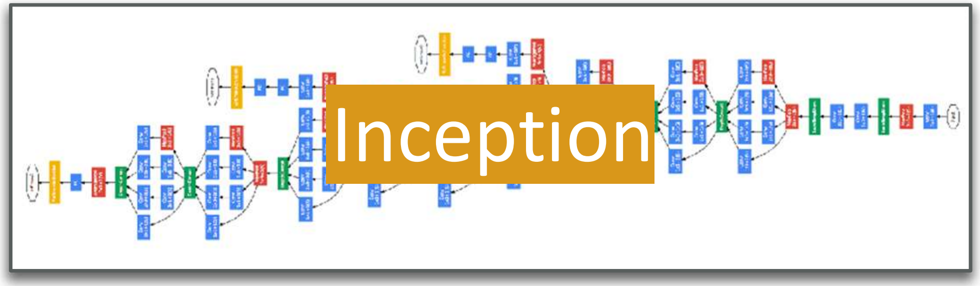
- Classification Network (DenseNet)
- Implement DenseNet with Pytorch
- Further Extensions
  - Data augmentation
  - Attention Mechanism
  - Contrastive Learning

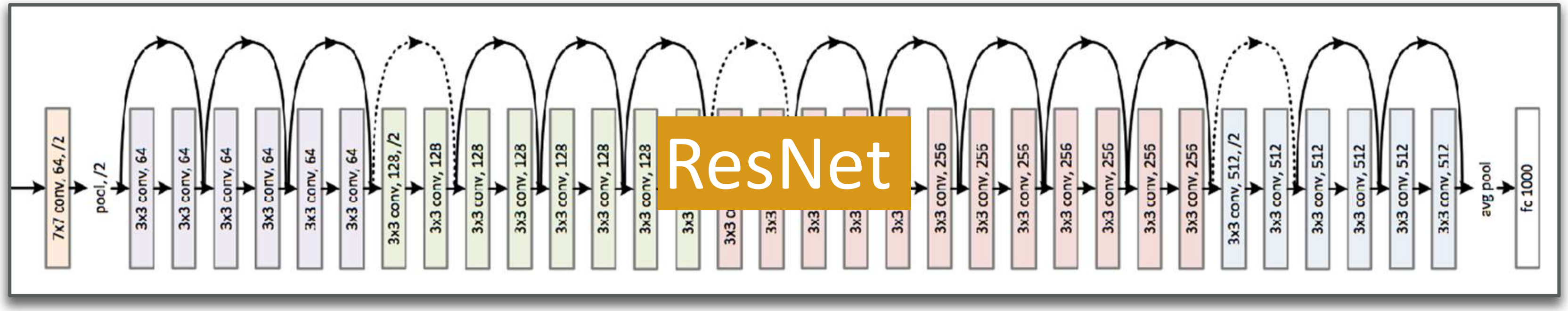
CONVOLUTIONAL NETWORKS



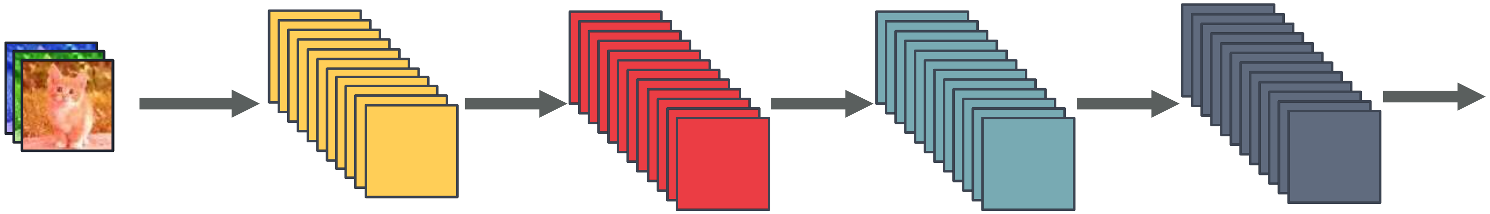






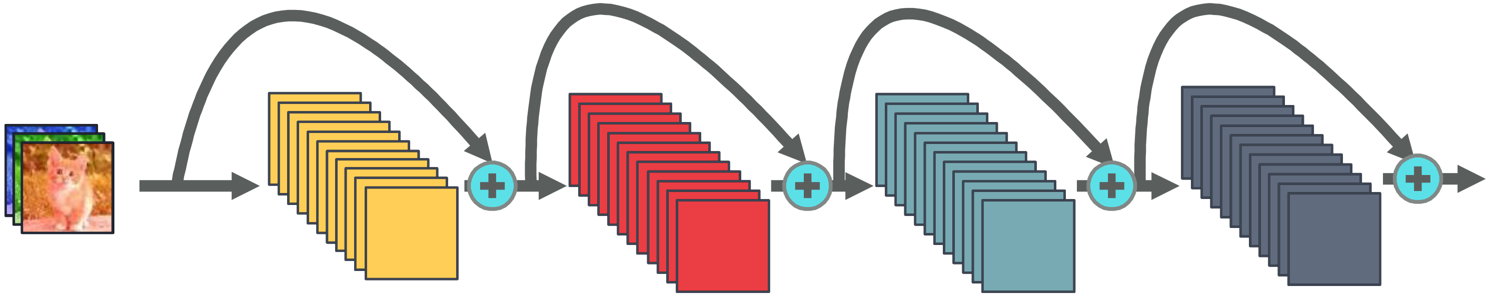


Standard Connectivity



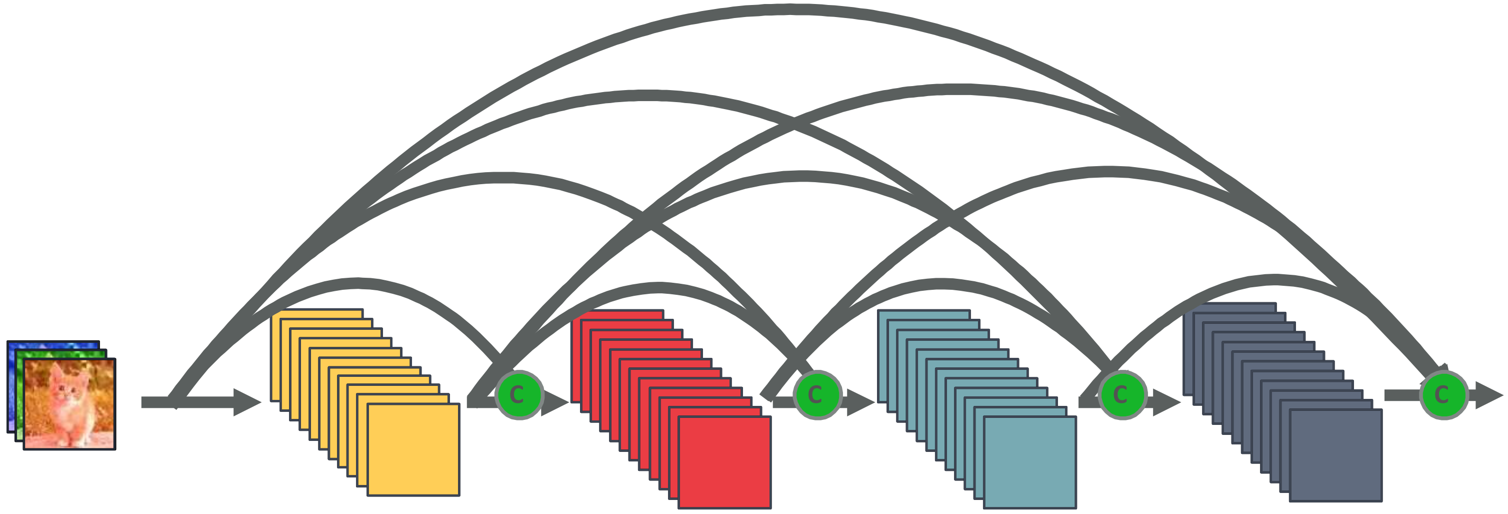
ResNet Connectivity

Identity mappings promote gradient propagation.



⊕ : Element-wise addition

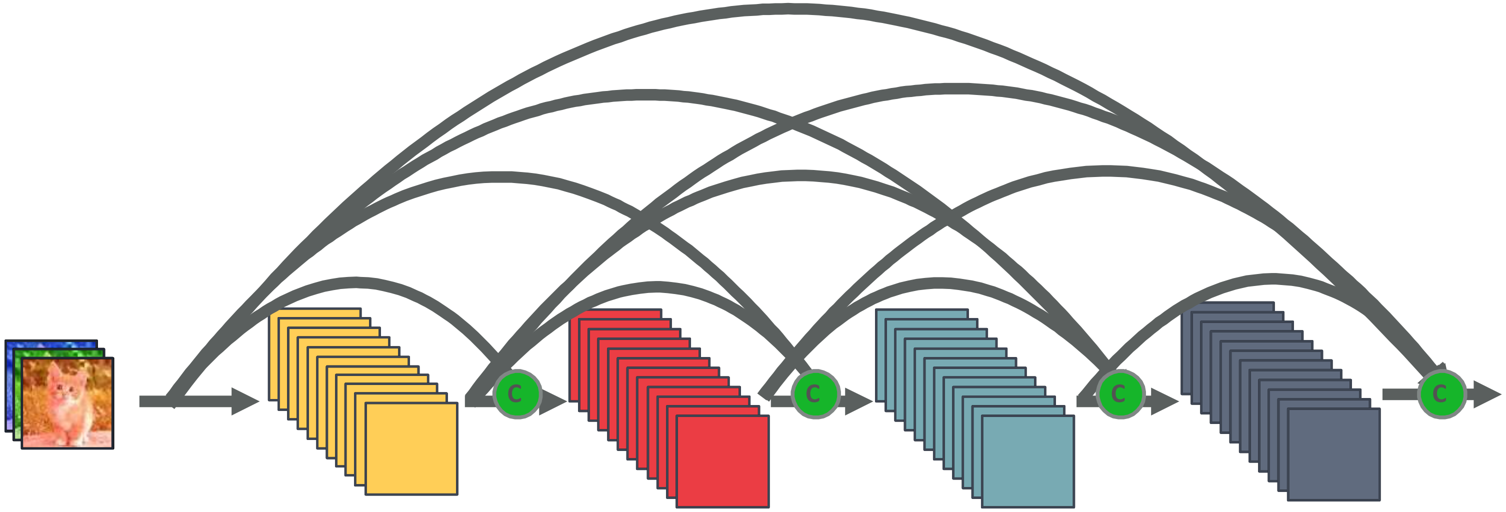
Dense Connectivity



C : Channel-wise concatenation

[1] Huang, Gao, et al. "Densely connected convolutional networks." CVPR, 2017.

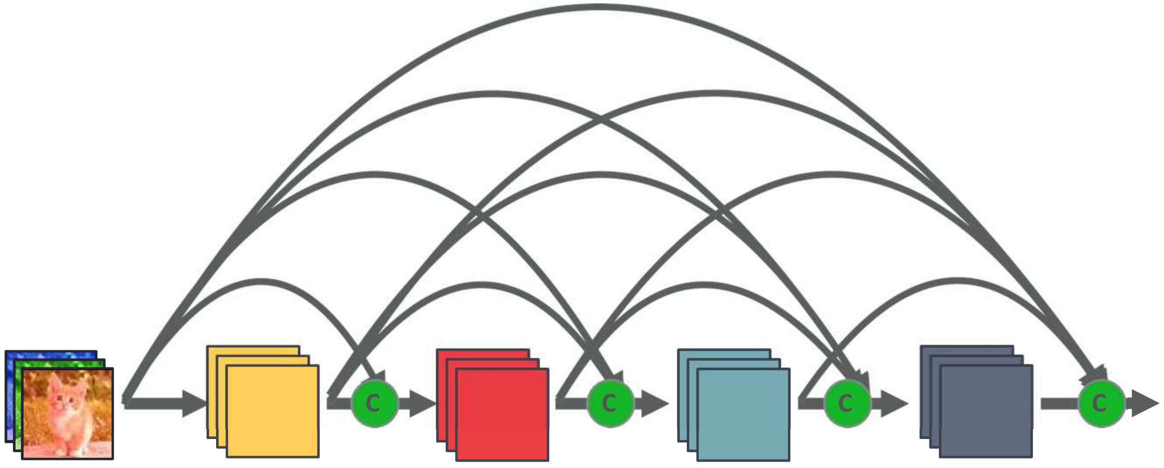
Dense and Slim



Ⓒ : Channel-wise concatenation

[1] Huang, Gao, et al. "Densely connected convolutional networks." CVPR, 2017.

Dense and Slim



k channels

k channels

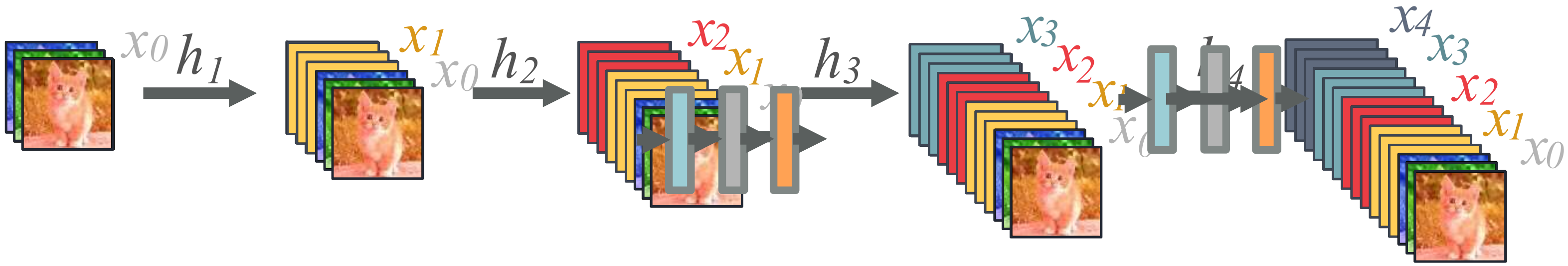
k channels

k channels

k : Growth Rate

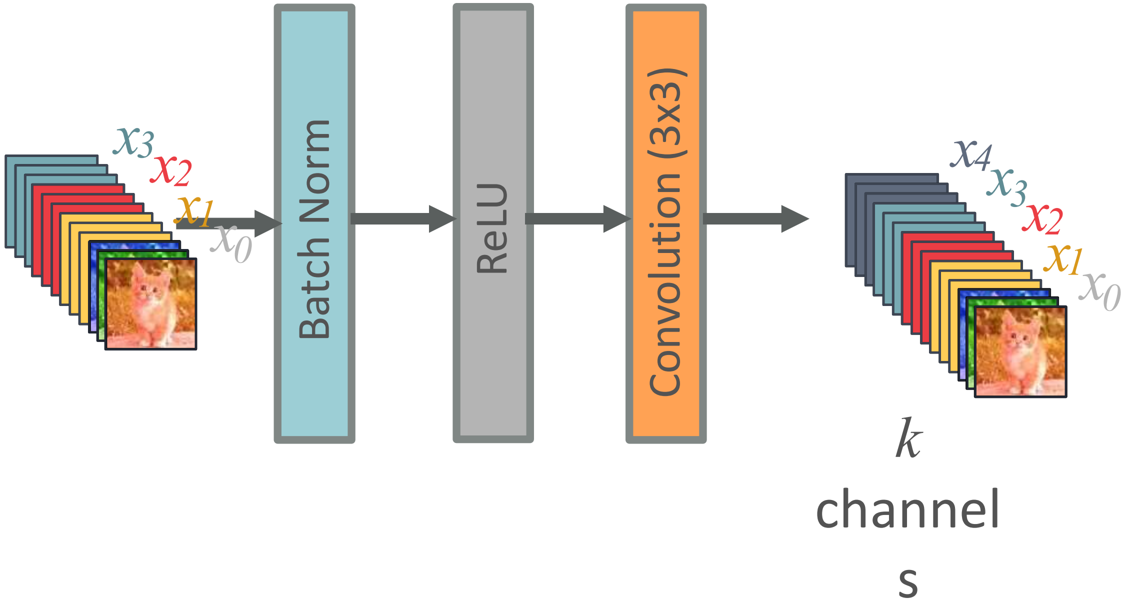
[1] Huang, Gao, et al. "Densely connected convolutional networks." CVPR, 2017.

Forward Propagation of Dense Block



[1] Huang, Gao, et al. "Densely connected convolutional networks." CVPR, 2017.

Composite Layer in DenseNet

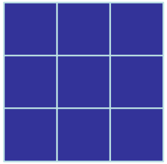


$$x_5 = h_5([x_0, \ldots, x_4])$$

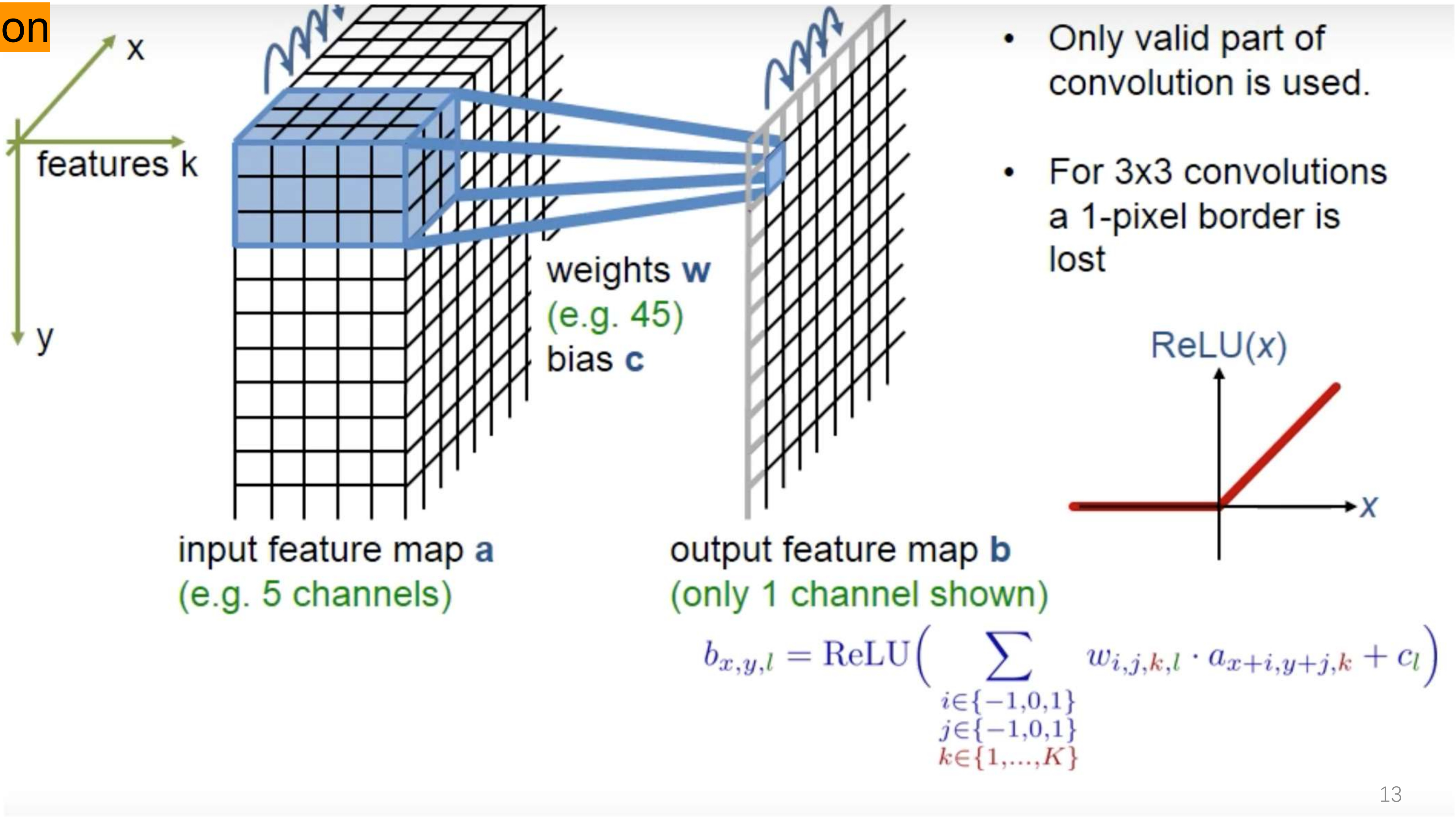
[1] Huang, Gao, et al. "Densely connected convolutional networks." CVPR, 2017.

DenseNet

Convolution



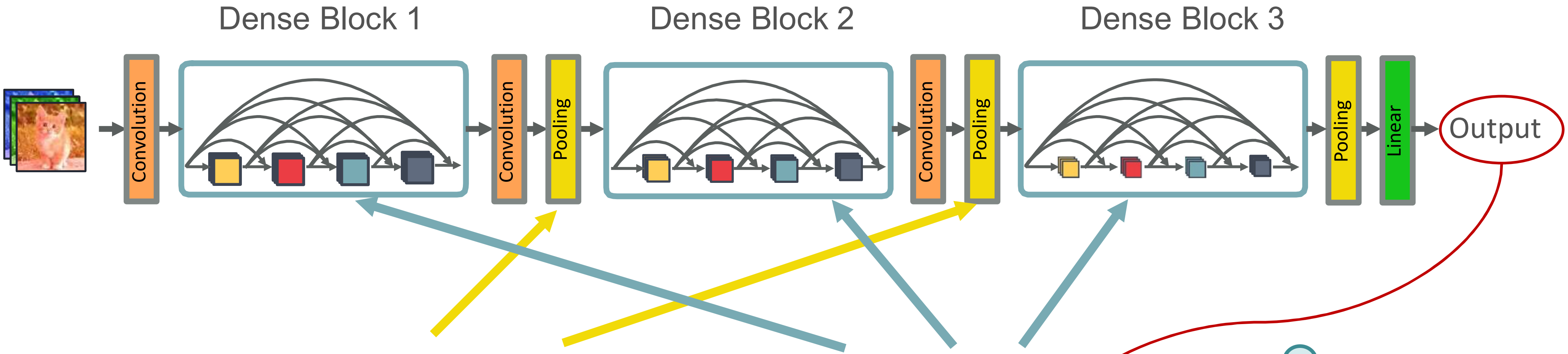
3x3 kernel



- Only valid part of convolution is used.
- For 3x3 convolutions a 1-pixel border is lost

$$b_{x,y,l} = \mathrm{ReLU}\Bigg(\sum_{\substack{i\in\{-1,0,1\}\\ j\in\{-1,0,1\}\\ k\in\{1,\ldots,K\}}} w_{i,j,k,l} \cdot a_{x+i,y+j,k} + c_l \Bigg)$$

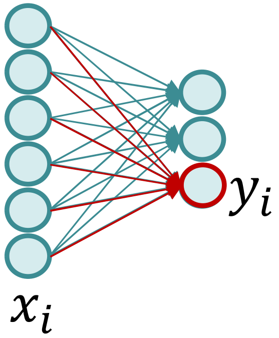
DenseNet [1]



Pooling reduces feature map sizes

Feature map sizes match within each block

Cross Entropy Loss $L_{softmax} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{W_{y_i}^{\top} x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^{\top} x_i + b_j}}$



[1] Huang, Gao, et al. "Densely connected convolutional networks." CVPR, 2017.

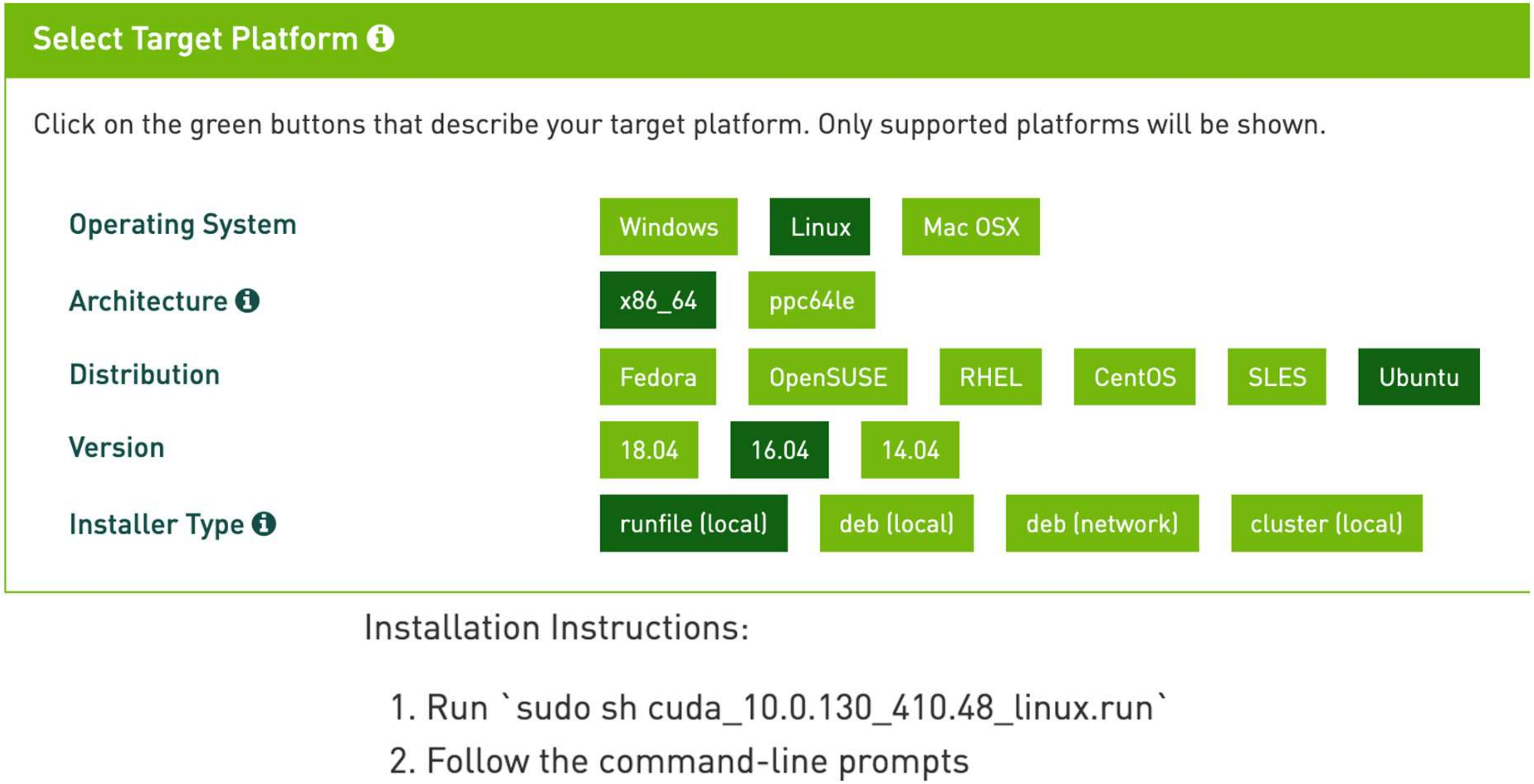
Outline

- Classification Network (DenseNet)
- Implement DenseNet with Pytorch
- Further Extensions
  - Data augmentation
  - Attention Mechanism
  - Contrastive Learning

Implement DenseNet with Pytorch -- Install CUDA

NVIDIA. DEVELOPER

- Download: https://developer.nvidia.com/cuda-10.0-download-archive



Select Target Platform

Click on the green buttons that describe your target platform. Only supported platforms will be shown.

Operating System: Windows [✓ Linux] Mac OSX

Architecture: [✓ x86_64] ppc64le

Distribution: Fedora OpenSUSE RHEL CentOS SLES [✓ Ubuntu]

Version: 18.04 [✓ 16.04] 14.04

Installer Type: [✓ runfile (local)] deb (local) deb (network) cluster (local)

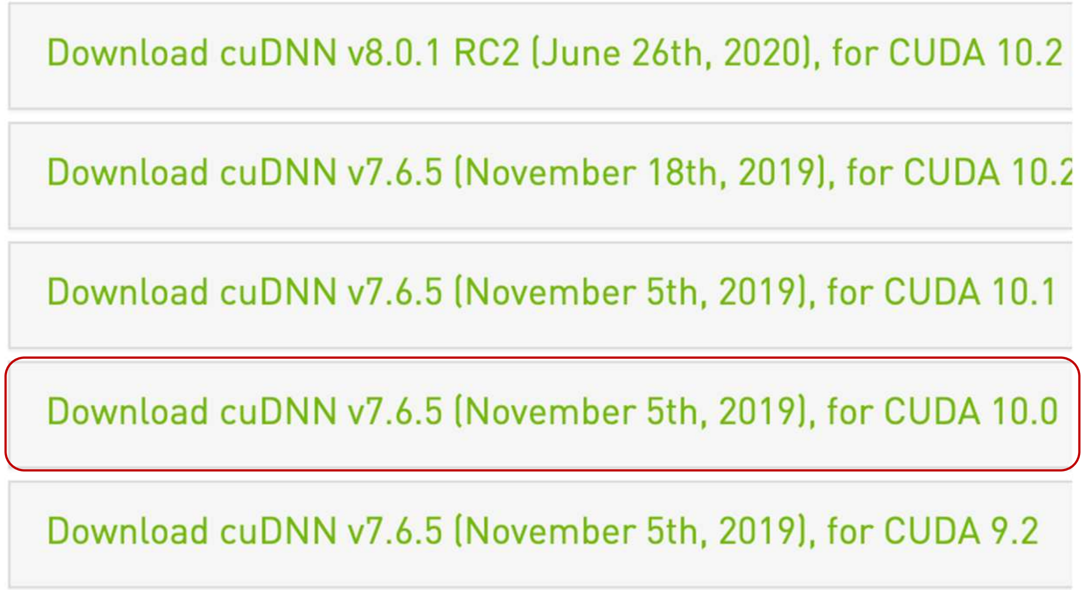
Installation Instructions:

1. Run `sudo sh cuda_10.0.130_410.48_linux.run`
2. Follow the command-line prompts

Implement DenseNet with Pytorch -- Install cuDNN

NVIDIA. DEVELOPER

- Download: https://developer.nvidia.com/rdp/cudnn-archive



Download cuDNN v8.0.1 RC2 (June 26th, 2020), for CUDA 10.2

Download cuDNN v7.6.5 (November 18th, 2019), for CUDA 10.2

Download cuDNN v7.6.5 (November 5th, 2019), for CUDA 10.1

Download cuDNN v7.6.5 (November 5th, 2019), for CUDA 10.0

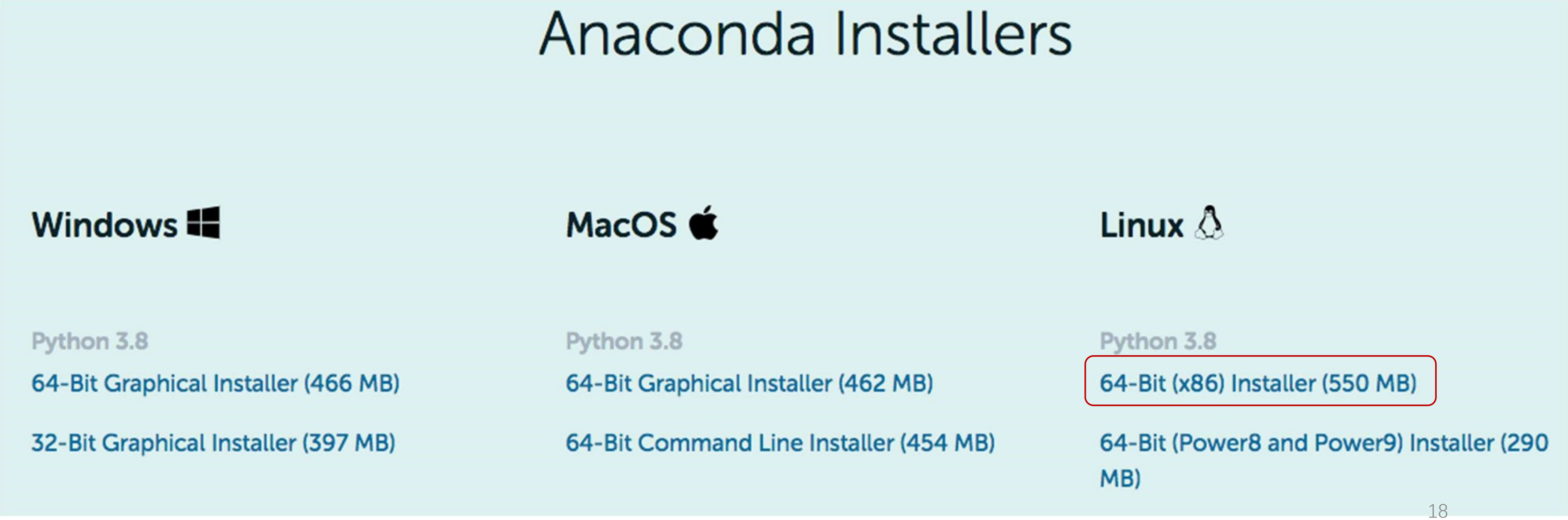
Download cuDNN v7.6.5 (November 5th, 2019), for CUDA 9.2

- You should choose the matched version of cuDNN with the installed CUDA
- Installation instruction: https://docs.nvidia.com/deeplearning/cudnn/install-guide/index.html#installlinux

Implement DenseNet with Pytorch -- Install Anaconda

ANACONDA

- Download: https://www.anaconda.com/products/individual#linux



Anaconda Installers

Windows

Python 3.8

64-Bit Graphical Installer (466 MB)

32-Bit Graphical Installer (397 MB)

MacOS

Python 3.8

64-Bit Graphical Installer (462 MB)

64-Bit Command Line Installer (454 MB)

Linux

Python 3.8

64-Bit (x86) Installer (550 MB)

64-Bit (Power8 and Power9) Installer (290 MB)

Implement DenseNet with Pytorch -- Install Anaconda

ANACONDA

- Install Steps:

```
cd /home  # The path you download Anaconda
bash Anaconda3-2020.07-Linux-x86_64.sh  # Enter,
          # Follow the command-line prompts and answer each question

# Once install successfully, test with
python  # If there is an error indicating ‘cannot find python’
sudo vim /etc/profile  # Configure and edit the environment variables
PATH=/home/user/anaconda3/bin:$PATH  # Add this line into the file
            # Save this configure file with ‘:wq!’
source /etc/profile  # Update the environment variables

# Test again
python
```

Implement DenseNet with Pytorch -- Install Pytorch

- Install torch, numpy, opencv-python, scikit-image, scipy

```
conda create -n torch020 python=3.6  # construct virtual
environment named pytorch
conda activate torch020  # activate virtual environment
Pip list  # show the installed library

pip install torch==1.7.0
pip install numpy
pip install opencv-python
pip install scikit-image
pip install scipy
# The resting required libraries can be installed
following the reported error when debugging
```

Implement DenseNet with Pytorch

- Dataset: 6-class image classification

buildings

forest

glacier

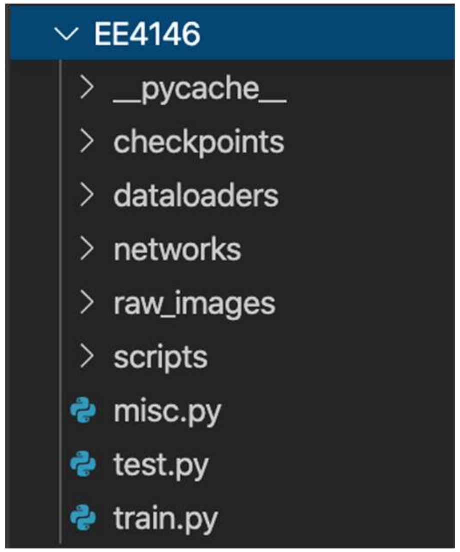
mountain

sea

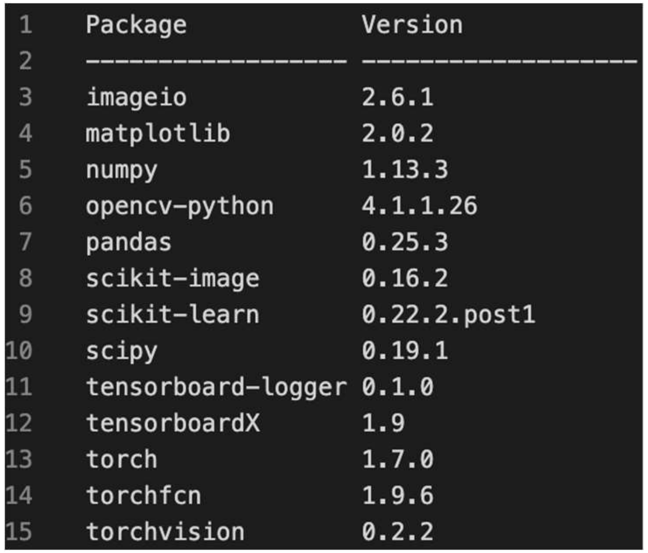
street

92.530%

- Code: https://drive.google.com/drive/folders/1j_FdcEOakzfZDVoPaGxiboNSYt961Acr?usp=sharing



```
⌄ EE4146
  > __pycache__
  > checkpoints
  > dataloaders
  > networks
  > raw_images
  > scripts
  misc.py
  test.py
  train.py
```



```
Package            Version
------------------ --------------------
imageio            2.6.1
matplotlib         2.0.2
numpy              1.13.3
opencv-python      4.1.1.26
pandas             0.25.3
scikit-image       0.16.2
scikit-learn       0.22.2.post1
scipy              0.19.1
tensorboard-logger 0.1.0
tensorboardX       1.9
torch              1.7.0
torchfcn           1.9.6
torchvision        0.2.2
```

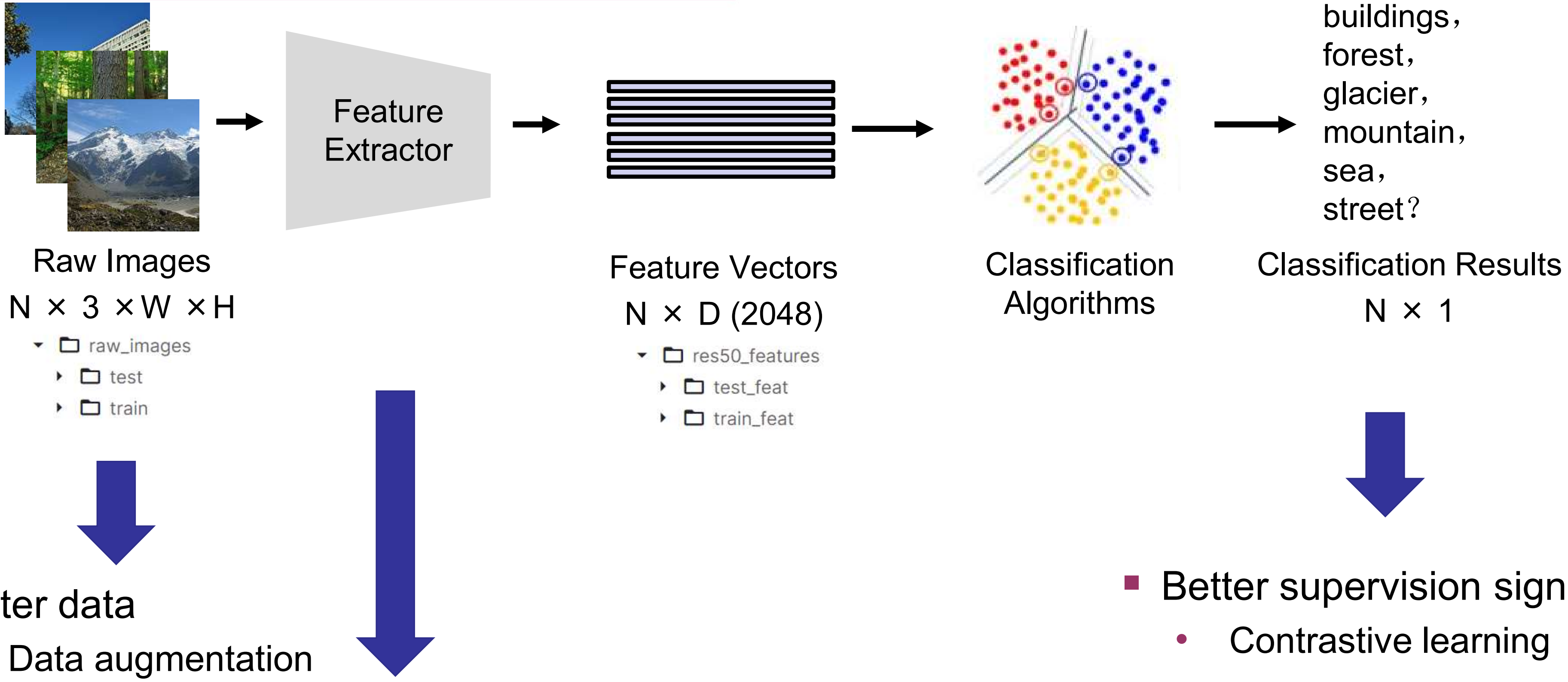
```
conda activate torch020  # activate virtual environment
pip install xxxx # DenseNet requires left dependent libraries
pip list  # show the installed library

# run training code:
cd ./scripts/
bash train.sh
# run testing code:
bash test.sh
```

Outline

- Classification Network (DenseNet)
- Implement DenseNet with Pytorch
- Further Extensions
  - Data augmentation
  - Attention Mechanism
  - Contrastive Learning

Further Extensions



Raw Images

N × 3 ×W ×H

```
▾ 🗀 raw_images
  ▸ 🗀 test
  ▸ 🗀 train
```

Feature Extractor

Feature Vectors

N × D (2048)

```
▾ 🗀 res50_features
  ▸ 🗀 test_feat
  ▸ 🗀 train_feat
```

Classification Algorithms

buildings，
forest，
glacier，
mountain，
sea，
street？

Classification Results

N × 1

- Better data
  - Data augmentation

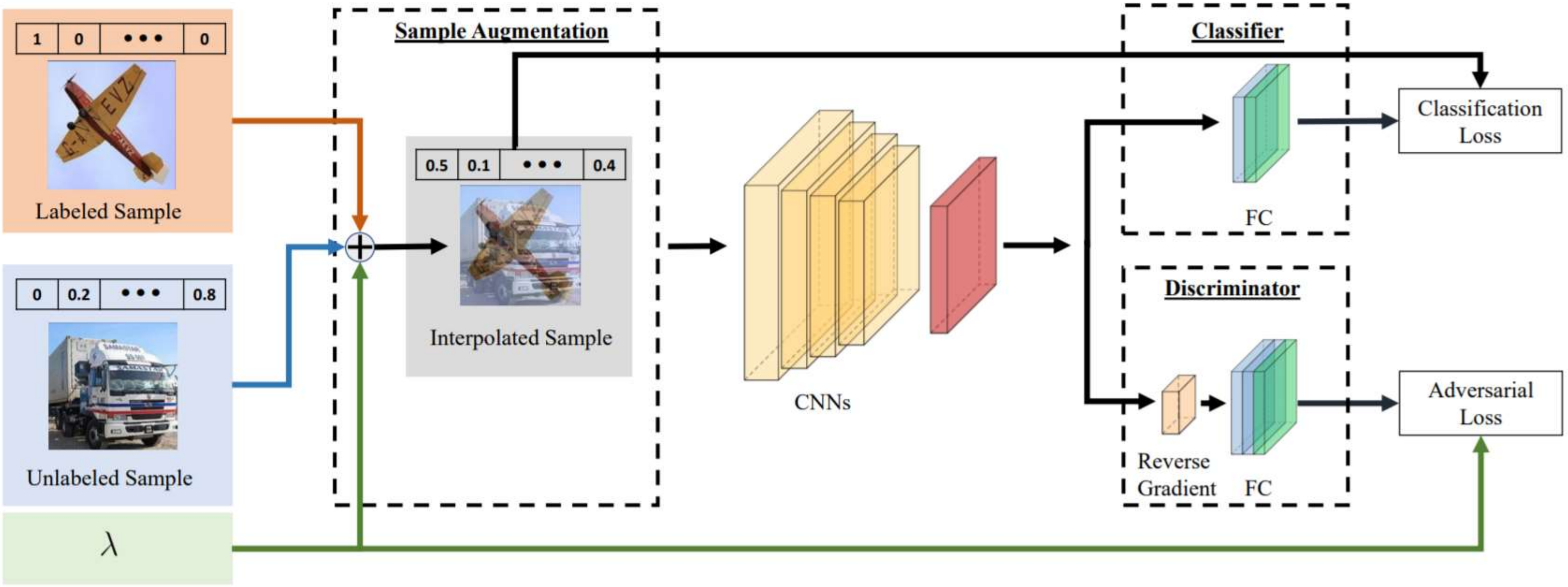
- Better classification network
  - Attention machanism

- Better supervision signals
  - Contrastive learning

Further Extensions

- Data Augmentation – Mixup [2]

Code: https://github.com/qinenergy/adanet

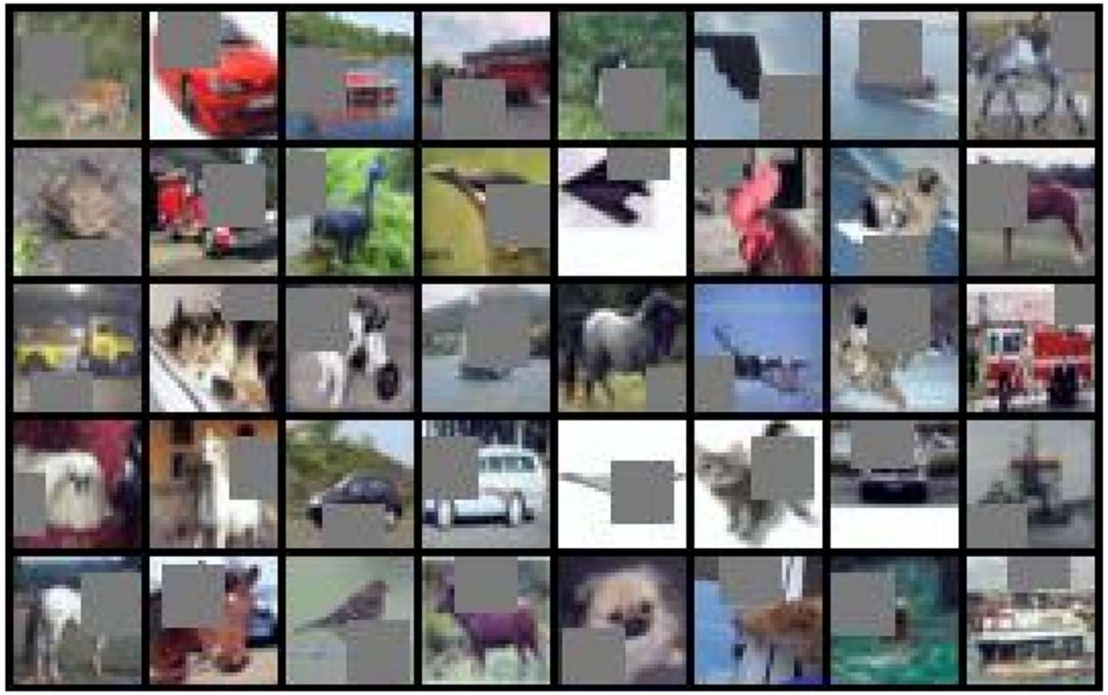


[2] Zhang, Hongyi, et al. "mixup: Beyond Empirical Risk Minimization." ICLR. 2018.

Further Extensions

- Data Augmentation – Cutout [3]

  Code: https://github.com/uoguelph-mlrg/Cutout

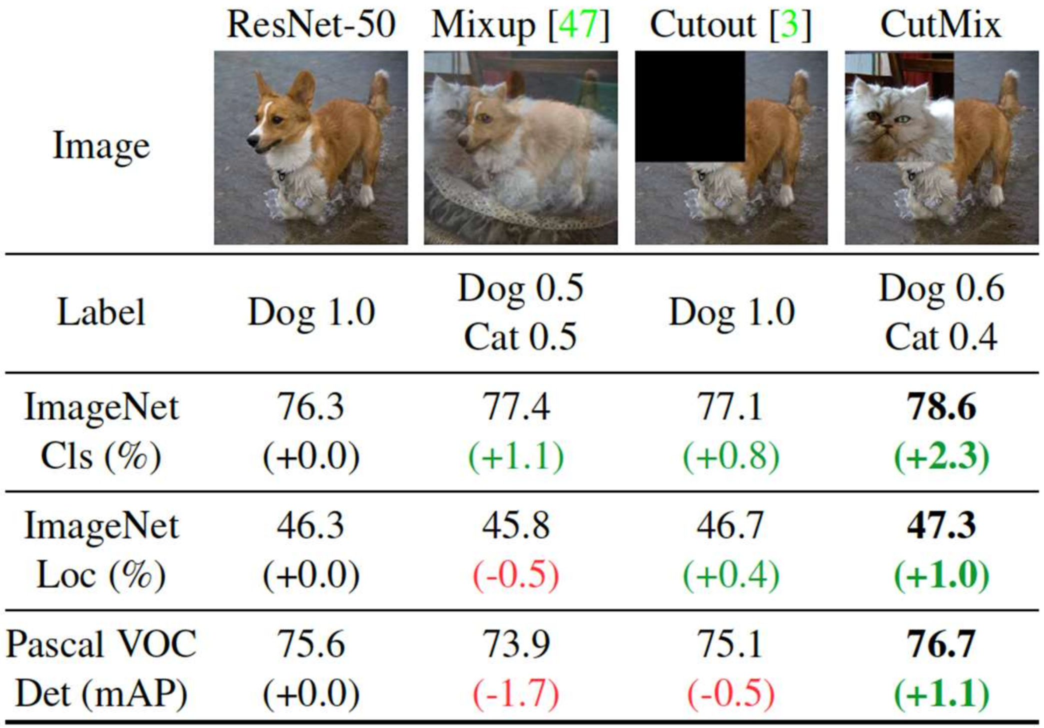


[3] DeVries, Terrance, and Graham W. Taylor. "Improved regularization of convolutional neural networks with cutout." arXiv preprint arXiv:1708.04552 (2017).

Further Extensions

- Data Augmentation – Cutmix [4]

Code: https://github.com/clovaai/CutMix-PyTorch



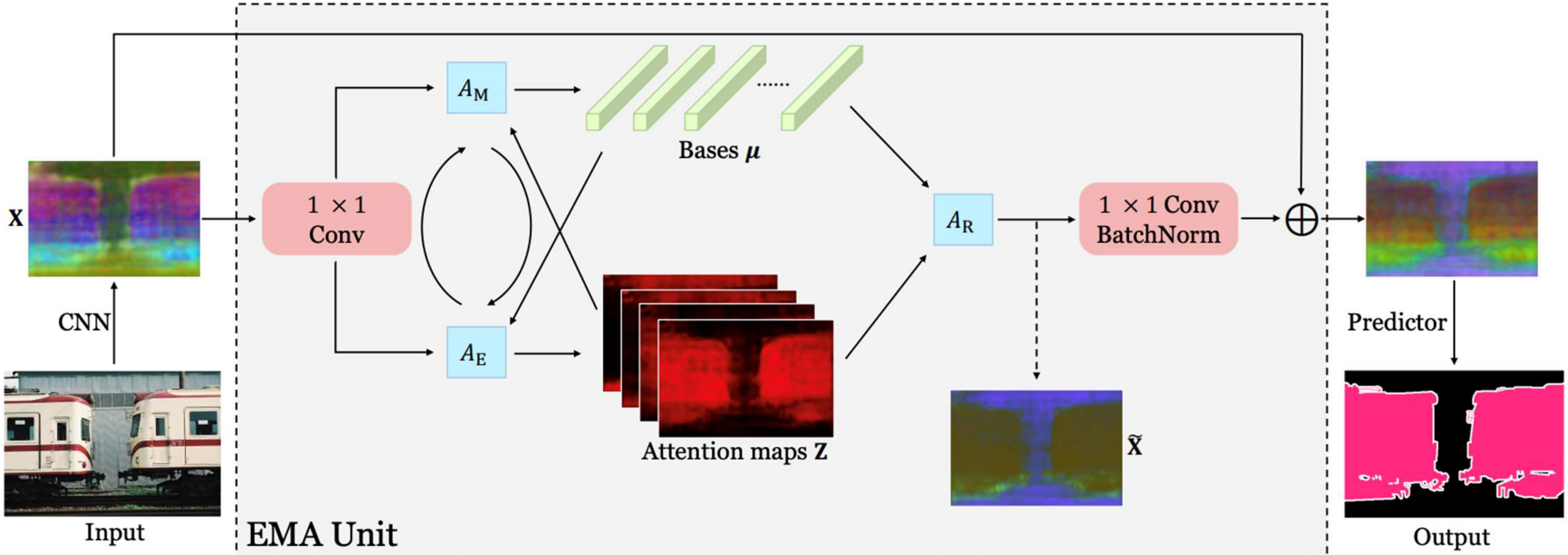
	ResNet-50	Mixup [47]	Cutout [3]	CutMix
Image				
Label	Dog 1.0	Dog 0.5 Cat 0.5	Dog 1.0	Dog 0.6 Cat 0.4
ImageNet Cls (%)	76.3 (+0.0)	77.4 (+1.1)	77.1 (+0.8)	**78.6** **(+2.3)**
ImageNet Loc (%)	46.3 (+0.0)	45.8 (-0.5)	46.7 (+0.4)	**47.3** **(+1.0)**
Pascal VOC Det (mAP)	75.6 (+0.0)	73.9 (-1.7)	75.1 (-0.5)	**76.7** **(+1.1)**

[4] Yun, Sangdoo, et al. "Cutmix: Regularization strategy to train strong classifiers with localizable features." ICCV. 2019.

Further Extensions

- Attention Mechanism -- Expectation-Maximization Attention [5]

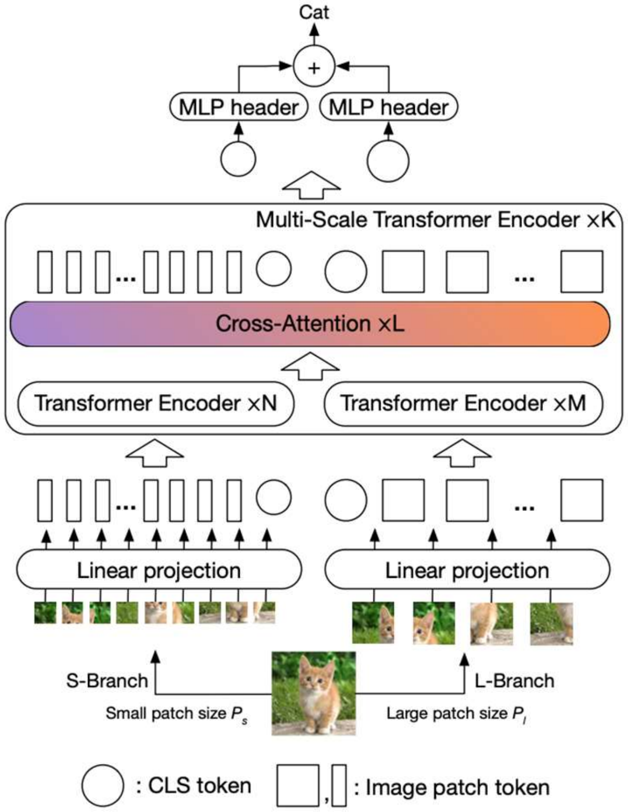
  Code: https://xialipku.github.io/EMANet/



[5] Li, Xia, et al. "Expectation-maximization attention networks for semantic segmentation." ICCV. 2019.

- Attention Mechanism – Transformer & Cross-attention module [6]

Code: https://github.com/IBM/CrossViT



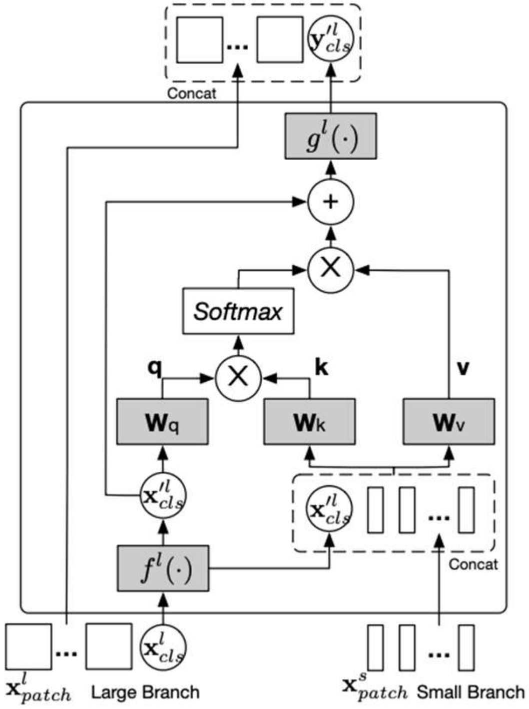


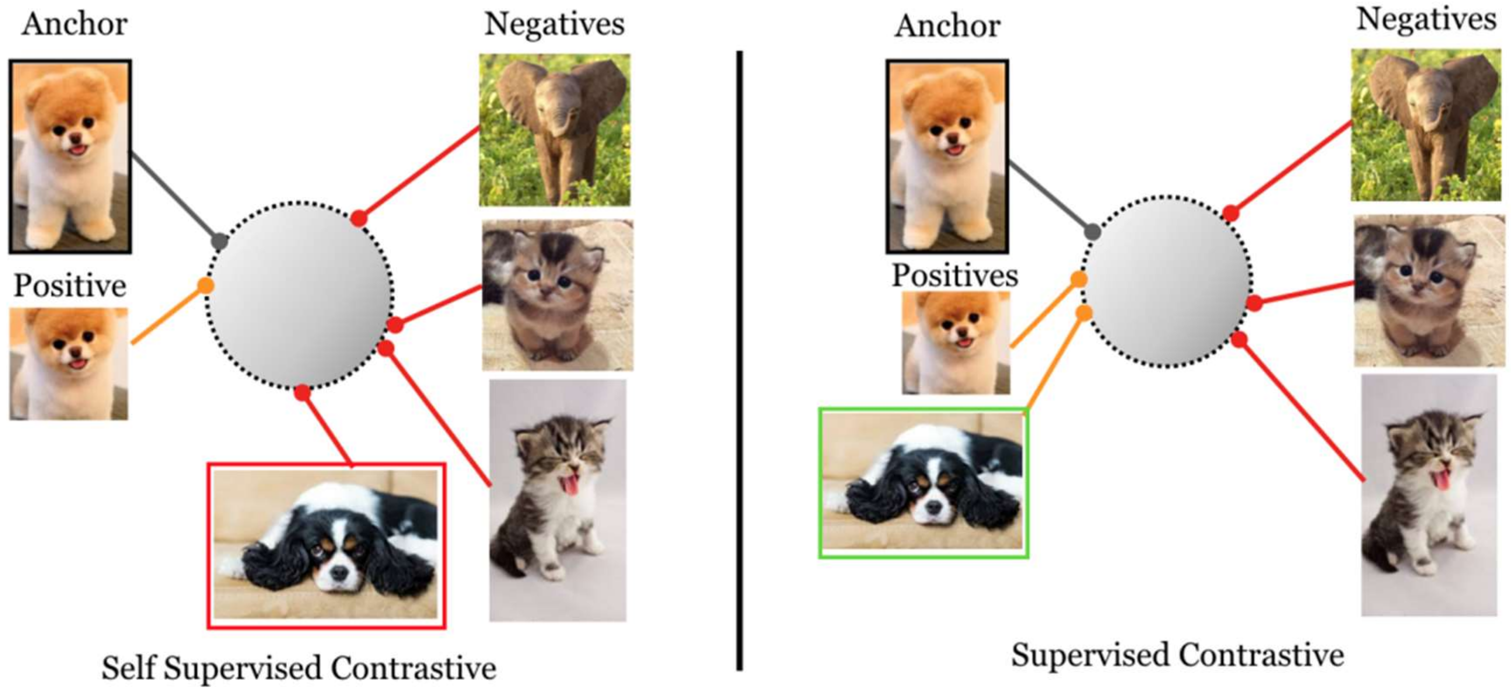
Figure 4: **Cross-attention module for Large branch**. The

[6] Chen, Chun-Fu, Quanfu Fan, and Rameswar Panda. "Crossvit: Cross-attention multi-scale vision transformer for image classification." ICCV (2021).

Further Extensions

- Contrastive Learning – Supervised Contrastive Learning [9]

  Code: https://github.com/HobbitLong/SupContrast



[9] Khosla, Prannay, et al. "Supervised Contrastive Learning." NeurIPS 33 (2020).

Thanks for listening!