HW4

TUTORIAL

NTHU Computer Vision Lab

Alex Lin

Outline

Introduction

Transfer Learning

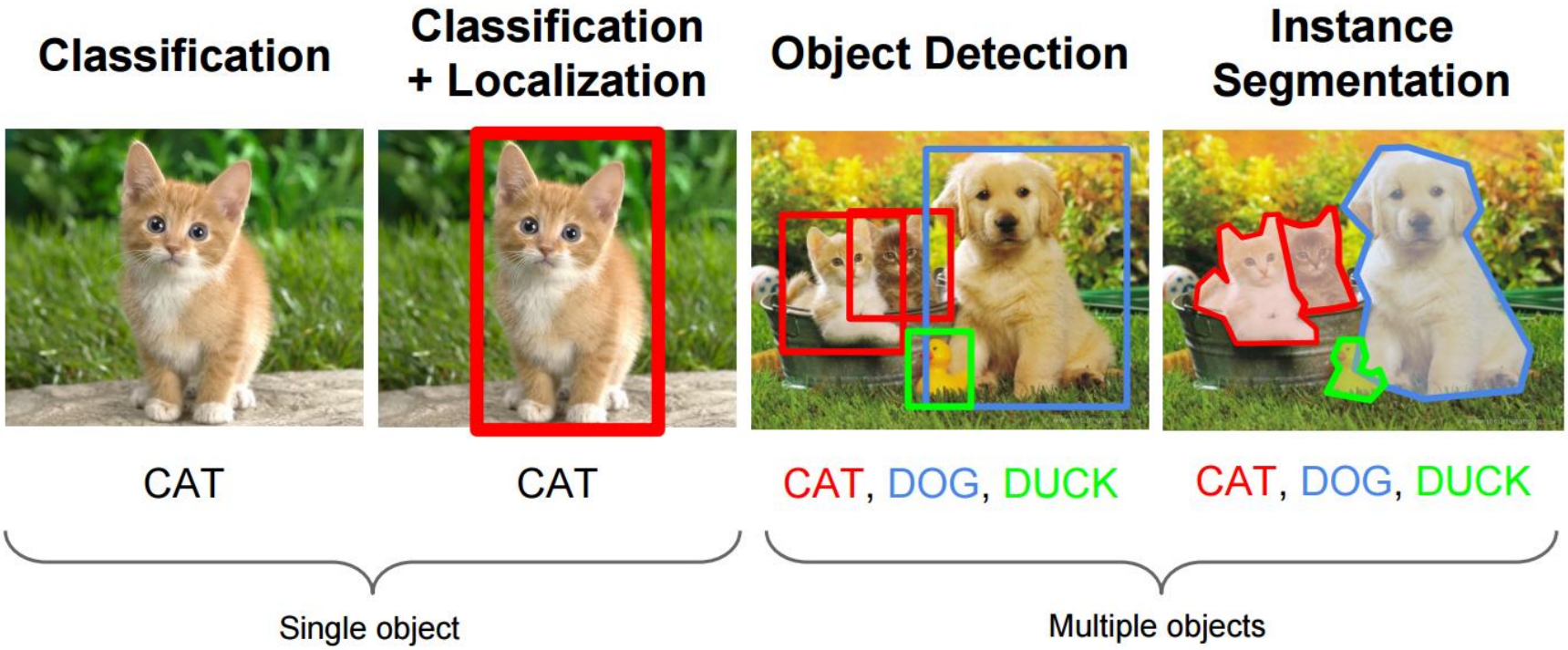
Two-Class Classification

Details of Q1

Semantic Segmentation

Details of Q2

Definition of CV tasks



Semantic Segmentation vs Instance Segmentation

Semantic Segmentation

Instance Segmentation

Transfer Learning

“You need a lot of a data if you want to train/use CNNs”

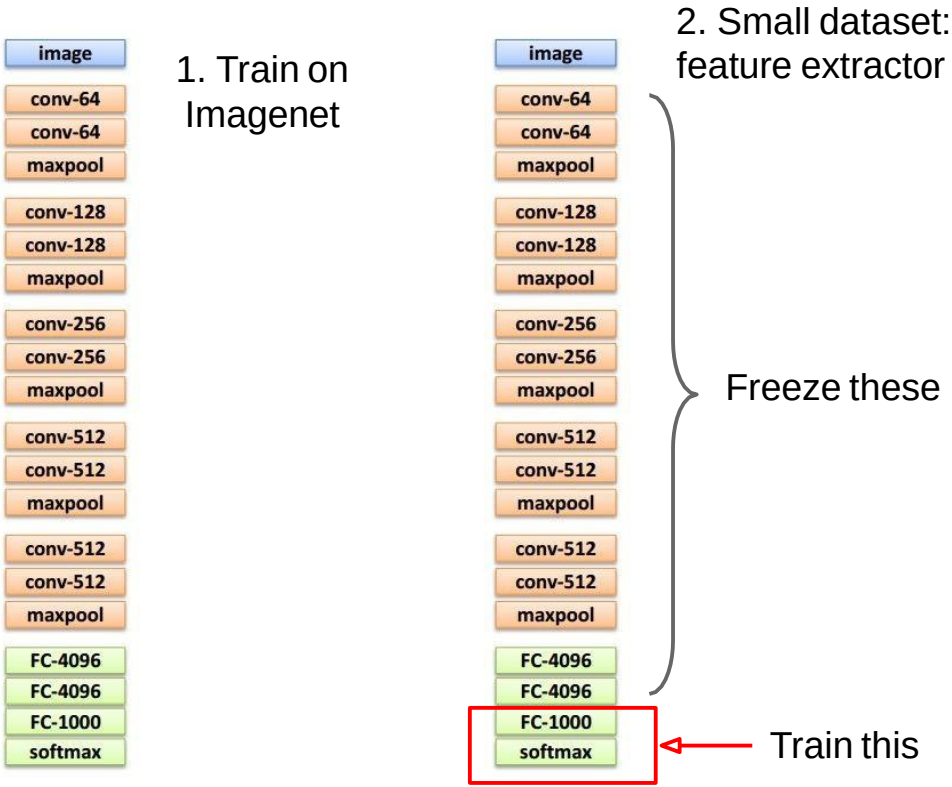
Transfer Learning

“You need a lot of data if you want to train/use CNNs”

Transfer Learning with CNNs

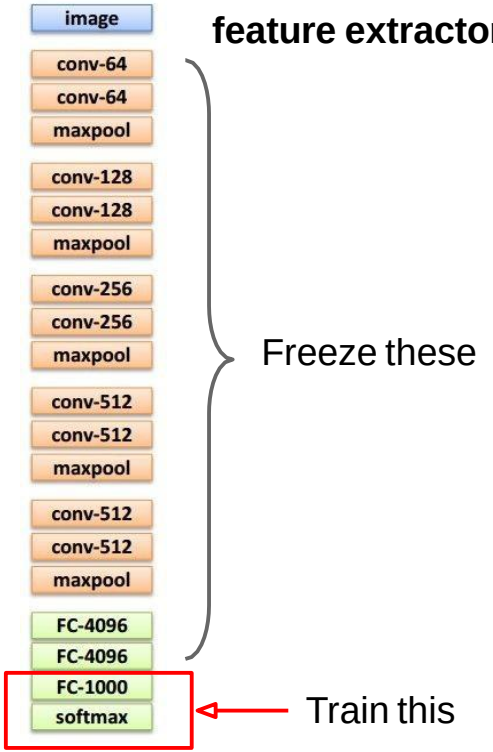
1. Train on Imagenet

Transfer Learning with CNNs

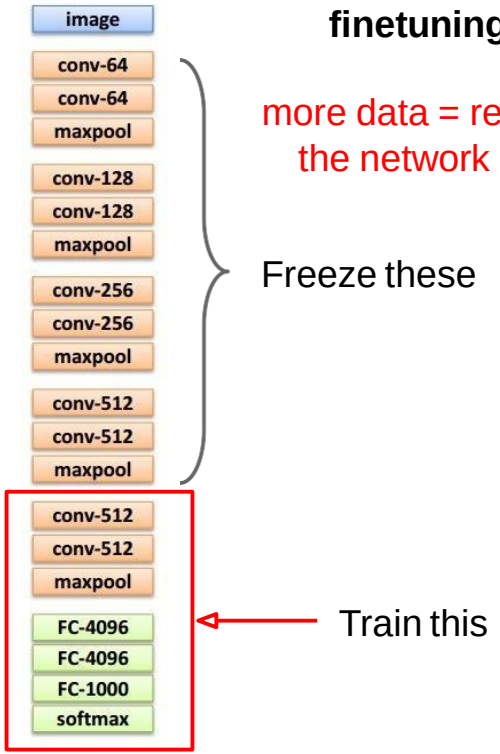


Transfer Learning with CNNs

1. Train on Imagenet



2. Small dataset: **feature extractor**



3. Medium dataset: **finetuning**

more data = retrain more of the network (or all of it)

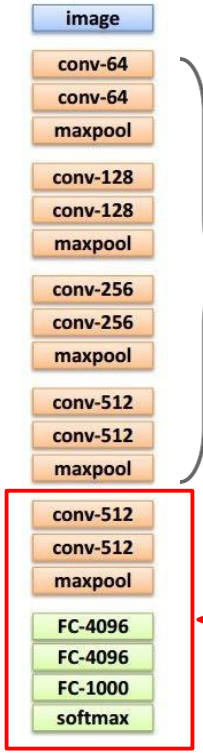
Transfer Learning with CNNs

1. Train on Imagenet

2. Small dataset: **feature extractor**

Freeze these

Train this



3. Medium dataset: **finetuning**

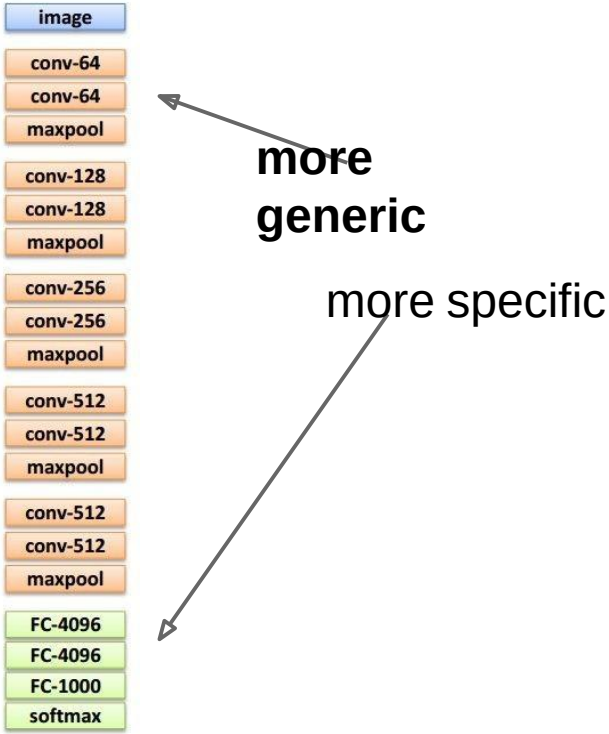
more data = retrain more of the network (or all of it)

Freeze these

tip: use only ~1/10th of the original learning rate in finetuning top layer, and ~1/100th on intermediate layers

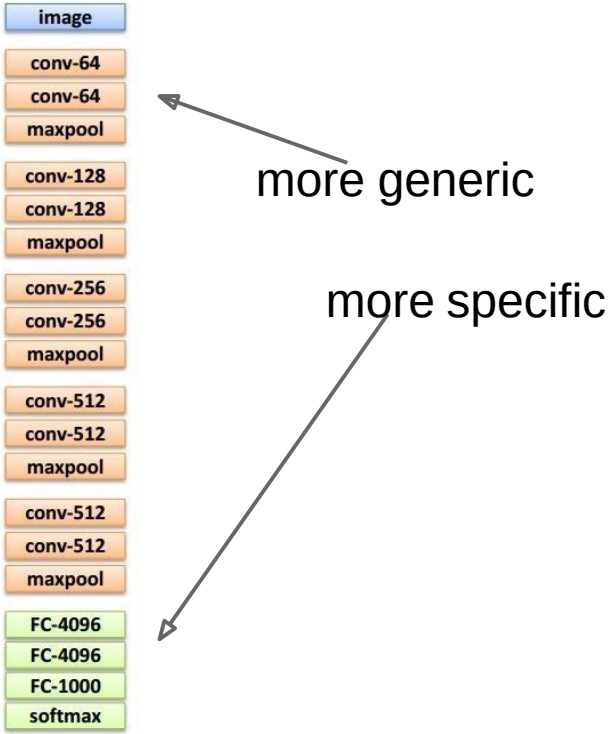
Train this

Transfer Learning with CNNs



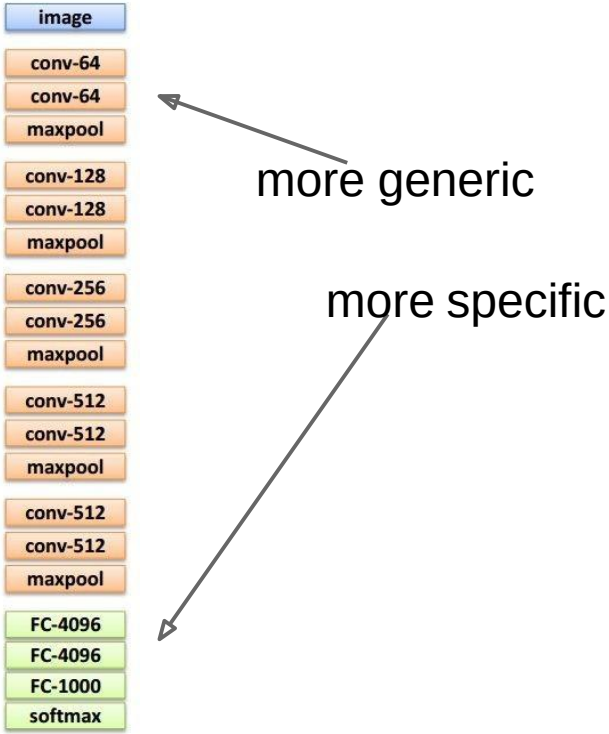
	very similar dataset	very different dataset
very little data	?	?
quite a lot of data	?	?

Transfer Learning with CNNs



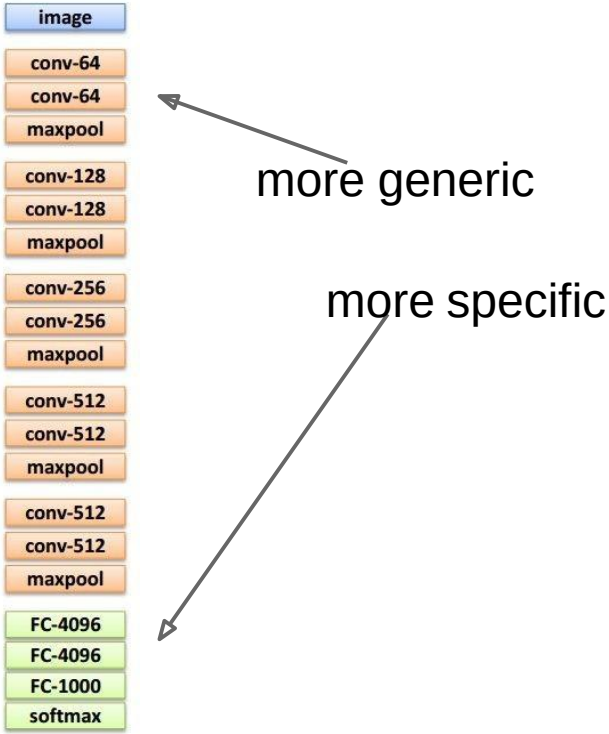
	very similar dataset	very different dataset
very little data	Use Linear Classifier on top layer	?
quite a lot of data	?	?

Transfer Learning with CNNs



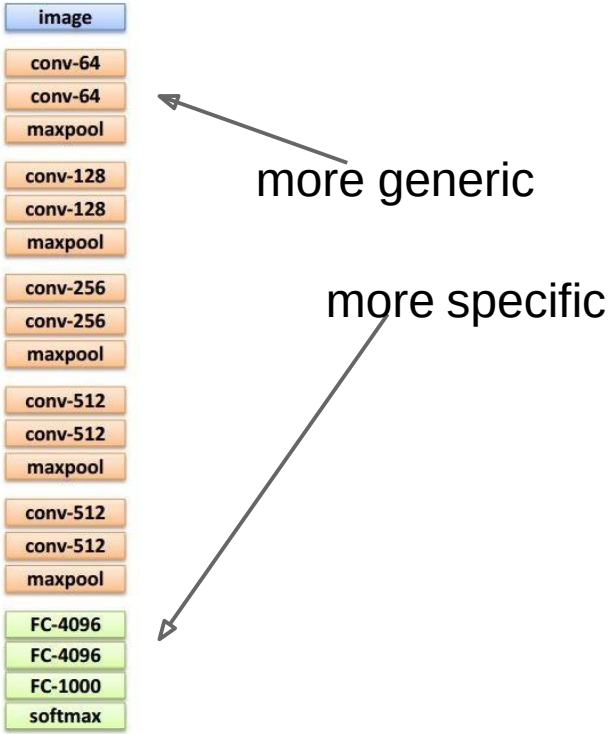
	very similar dataset	very different dataset
very little data	Use Linear Classifier on top layer	?
quite a lot of data	Finetune a few layers	?

Transfer Learning with CNNs



	very similar dataset	**very different dataset**
very little data	Use Linear Classifier on top layer	?
quite a lot of data	Finetune a few layers	Finetune a larger number of layers

Transfer Learning with CNNs



	very similar dataset	very different dataset
very little data	Use Linear Classifier on top layer	You're in trouble!
quite a lot of data	Finetune a few layers	Finetune a larger number of layers

Two-Class Classification Example

Takeshi Kaneshiro

Your Portrait

Two-Class Classification is actually not that easy!

Case Study: Pokémon v.s. Digimon

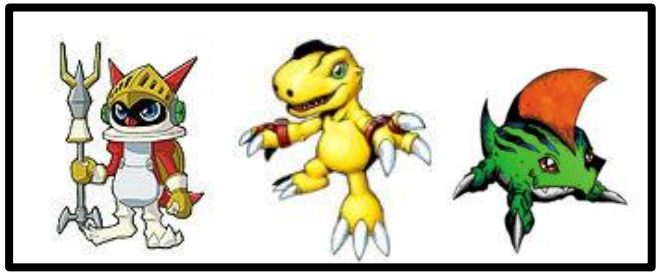
https://medium.com/@tyreeostevenson/teaching-a-computer-to-classify-anime-8c77bc89b881

Task

Pokémon images: https://www.Kaggle.com/kvpratama/pokemon-images-dataset/data

Digimon images: https://github.com/DeathReaper0965/Digimon-Generator-GAN

Pokémon



Digimon

Testing Images:

Experimental Results

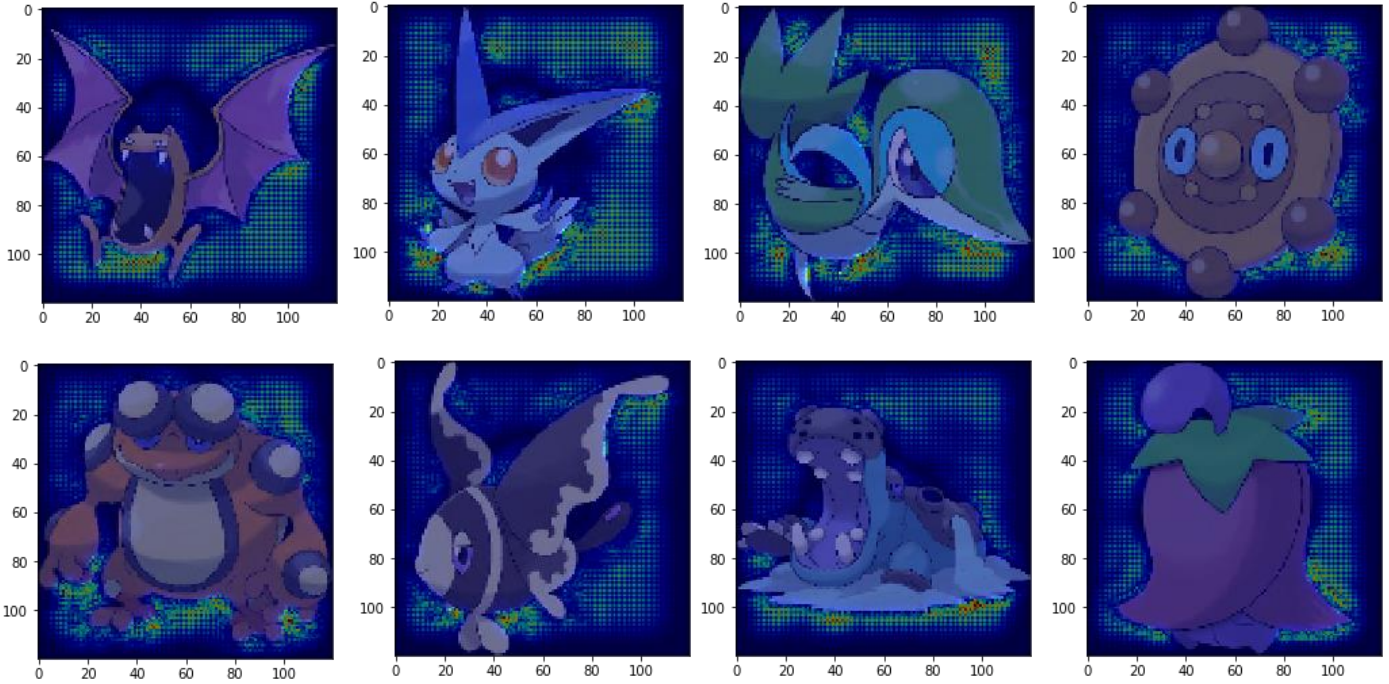
```
model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same', input_shape=(120,120,3)))
model.add(Activation('relu'))
model.add(Conv2D(32, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(256, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(256, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())
model.add(Dense(1024))
model.add(Activation('relu'))
model.add(Dense(2))
model.add(Activation('softmax'))
```

Training Accuracy: 98.9%

Testing Accuracy: 98.4%

太神啦!!!!!!

Saliency Map



What Happened?

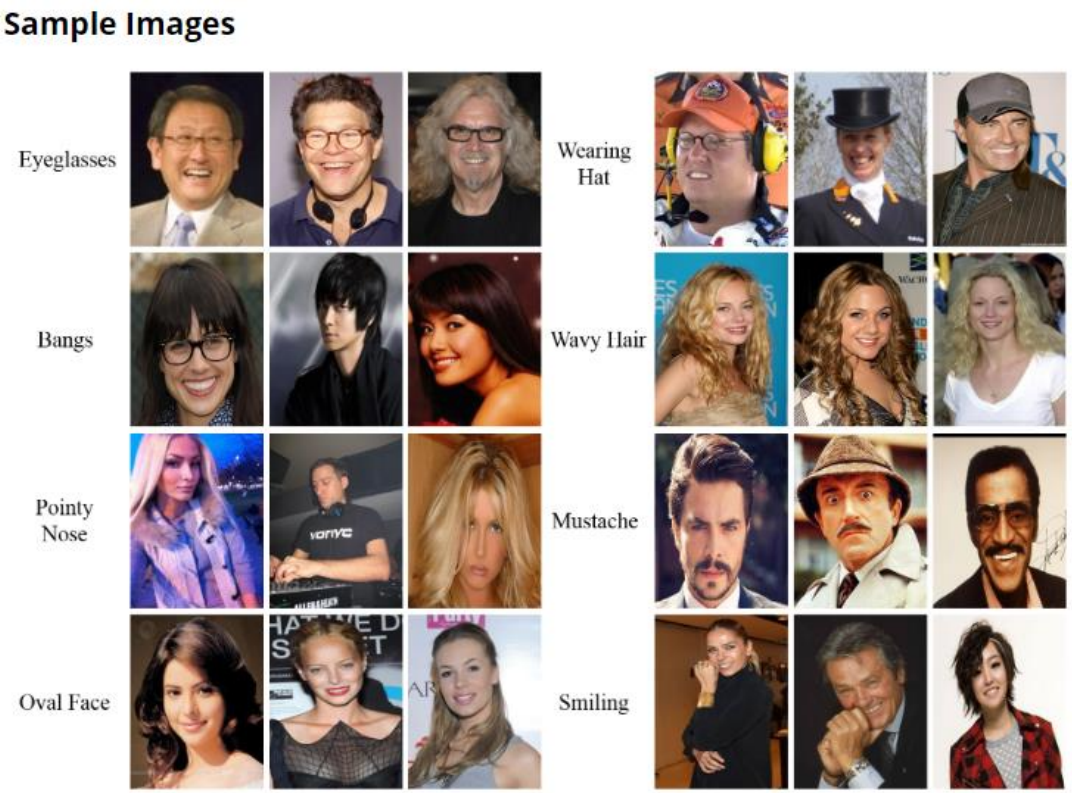
- All the images of Pokémon are PNG, while most images of Digimon are JPEG.

PNG 檔透明背景

讀檔後背景是黑的!

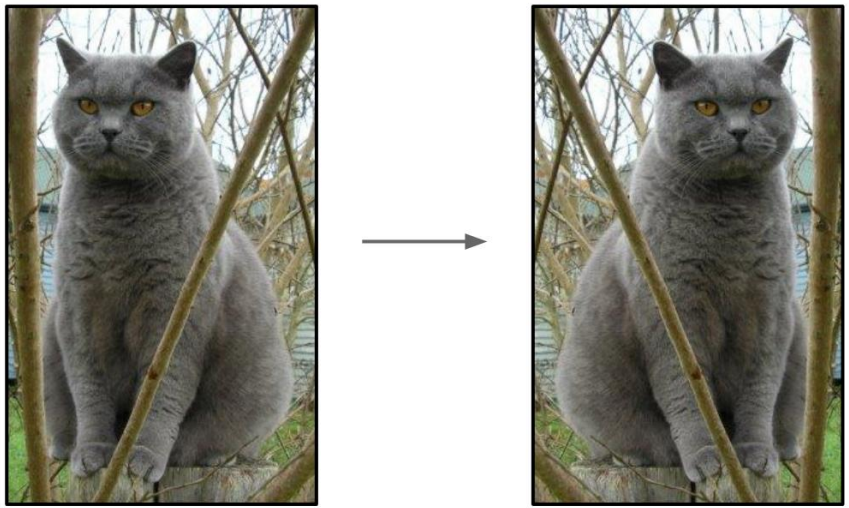
Machine discriminate Pokémon and Digimon based on Background color.

Dataset of our homework Q1: CelebA



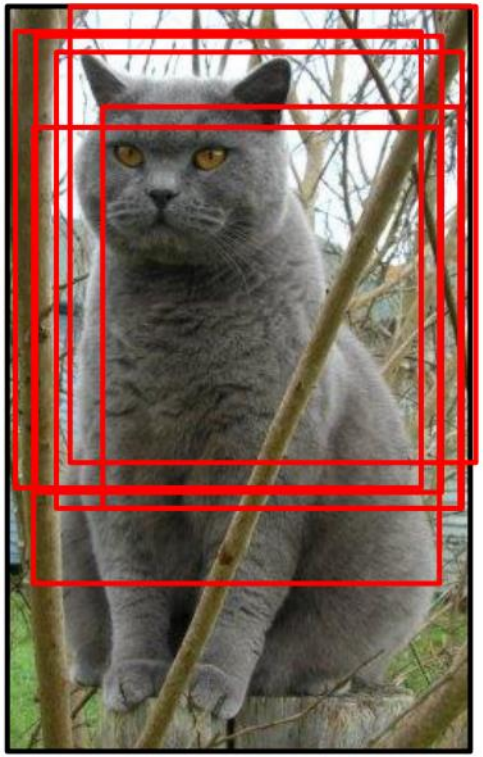
Do Better: Data Augmentation

- **Simulating “fake” data**
- **Explicitly encoding image transformations that shouldn’t change object identity.**
  - **Flip horizontally**

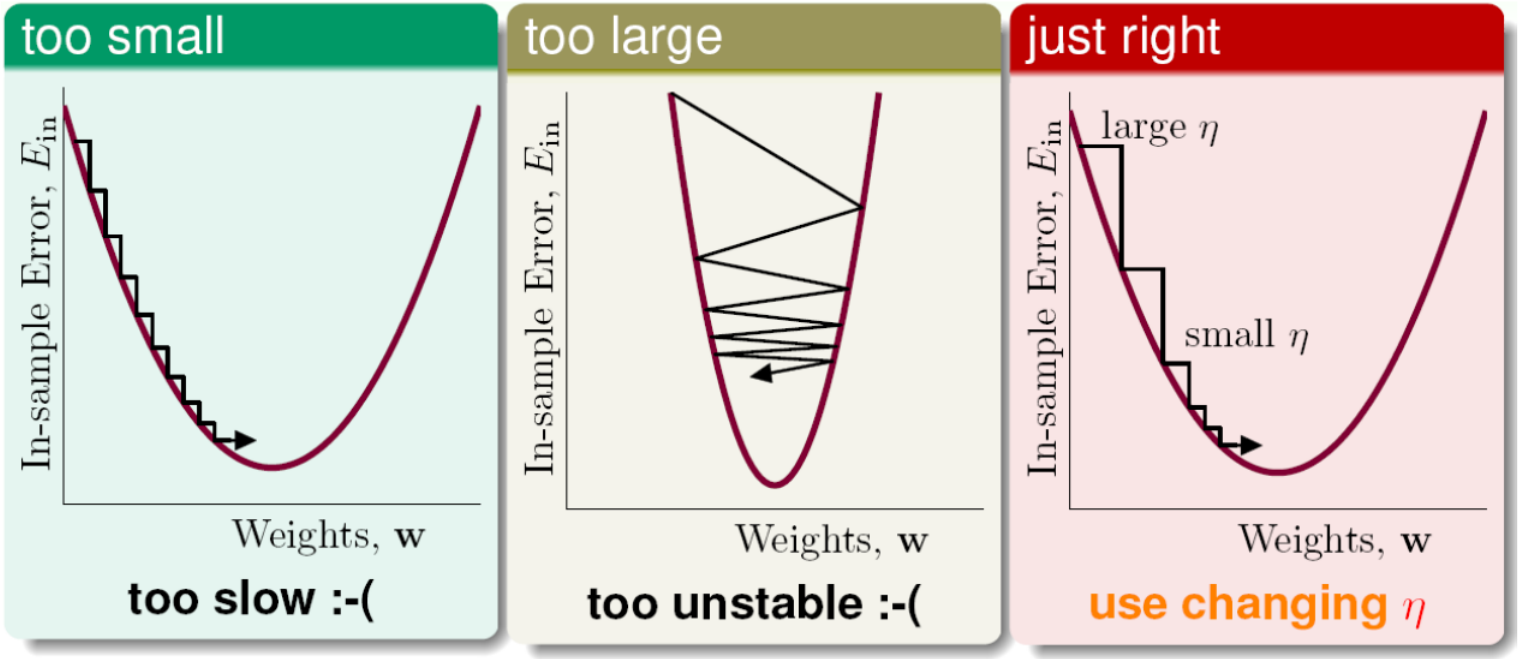


Do Better: Data Augmentation

- **Random/Multiple crops/scales**

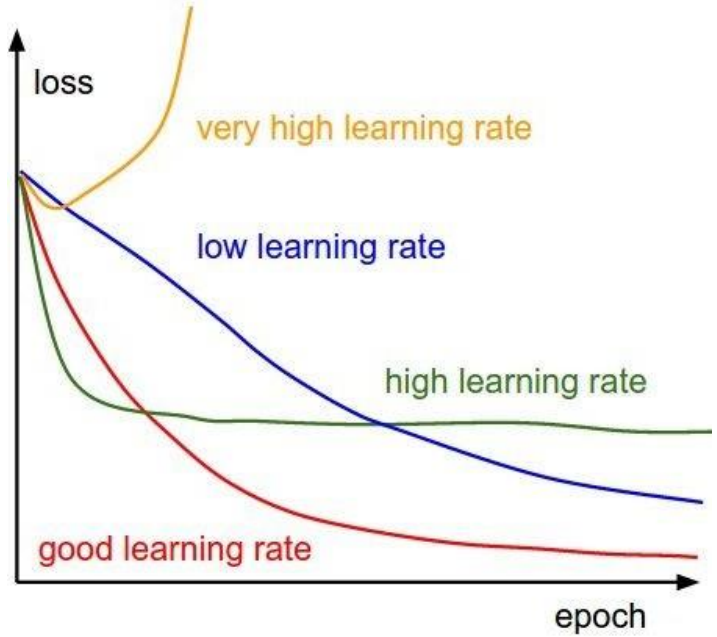


Choice of η



η better be **monotonic of** $\|\nabla E_{\text{in}}(\mathbf{w}_t)\|$

Current learning rate of the initial version



=> Learning rate decay over time!

step decay:
e.g. decay learning rate by 1/10 every few epochs.

Q1: Two-class classification for portraits with or without heavy makeup

Heavy makeup

No Heavy makeup

Dataset size

- Training set: 1000 imgs for heavy-makeup case; 1000 imgs for non-heavy-makeup case;
- Validation set: 200 imgs for heavy-makeup case; 200 imgs for non-heavy-makeup case;

Q1 initial version

- Backbone: Alexnet
- Data augmentation strategy: too heavy

- Quantitative metric: top-1 accuracy (no top-5 accuracy in this case)
- Accuracy of initial version:

Requirements of Q1

- Q1-1. Please report the validation accuracy of a pretrained Alexnet used as a feature extractor in the two-class classification problem. (5 pts)

  ps. Only 4096x2 layer would be finetuned

- Q1-2 Please report the validation accuracy of a pretrained Alexnet after it is fine-tuned in the two-class classification problem. (5 pts)

  ps. Please try to finetune every layer

- Q1-3 Please report the validation accuracy of a non-pretrained Alexnet after it is trained in the two-class classification problem. (5 pts)

  ps. Alexnet is trained from scratch

- Q1-4 Please discuss the results of Q1-1, Q1-2, & Q1-3. (5 pts)

Requirements of Q1

- Q1-5. Please try to correct the data augmentation strategy in order to let the entire face of each image be seen and report the validation accuracy of a pre-trained Alexnet as a feature extractor in the two-class classification problem. (5 pts)

- Q1-6. Please try to correct the data augmentation strategy in order to let the entire face of each image be seen and report the validation accuracy of a pretrained Alexnet after it is fine-tuned in the two-class classification problem. (5 pts)

- Q1-7. Please discuss the results of Q1-5 & Q1-6. (5pts)

Requirements of Q1

- Q1-8. Please try to achieve validation accuracy higher than 89.5% using a CNN other than Alexnet & ResNet-18 in the fine-tuning case. (20pts)

  ps. Please use the correct data augmentation strategy to achieve the best results.

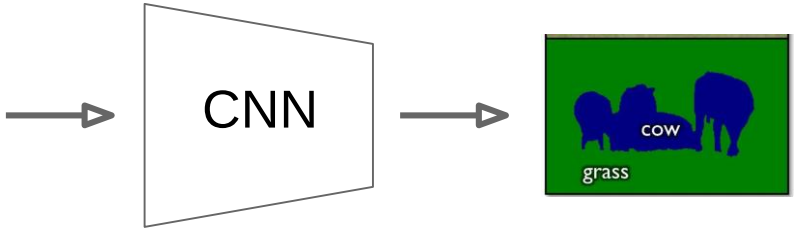
- Q1-9. Please discuss the results of Q1-9 (5pts if your meet the requirement of Q1-8)

Tips in Q1-8:

- Deeper pre-trained models
- Different optimizers
- More heavy data augmentation
- Different preprocessing tricks
- Training longer (not recommended!)
- More appropriate learning rate

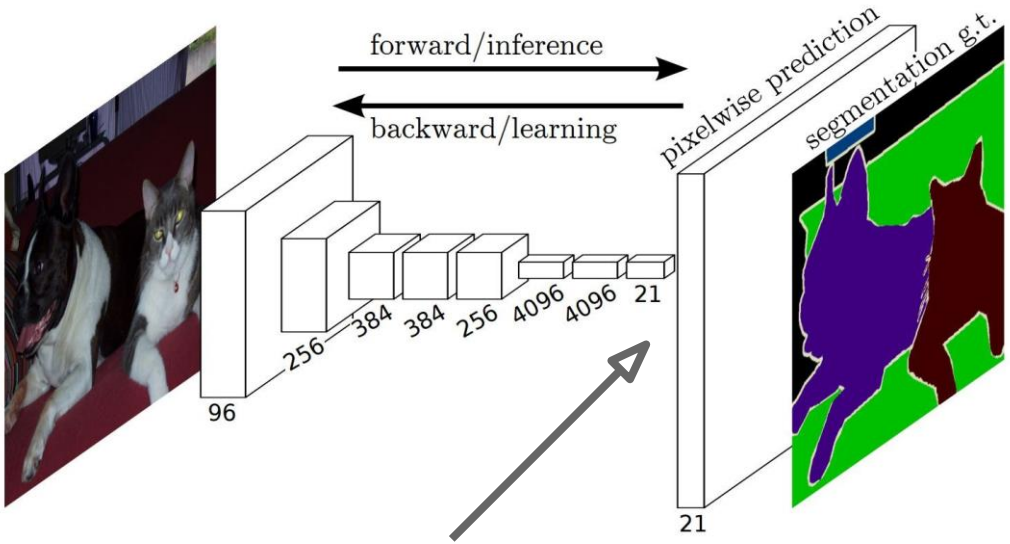
Semantic Segmentation

Run “fully convolutional” network
to get all pixels at once



Smaller output
due to pooling

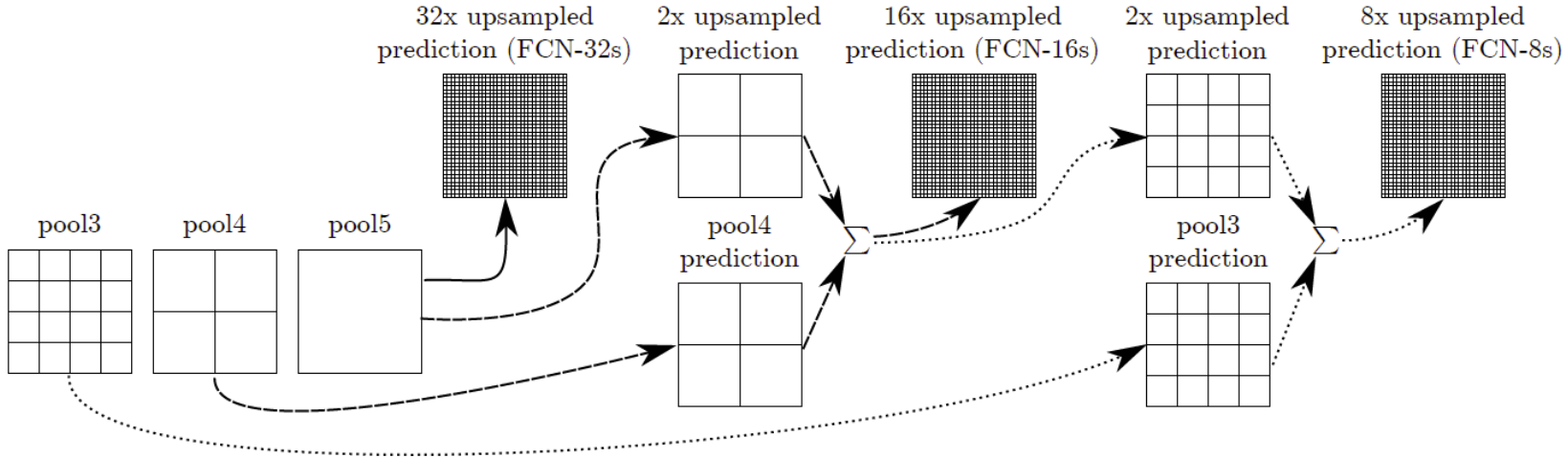
Semantic Segmentation: Upsampling

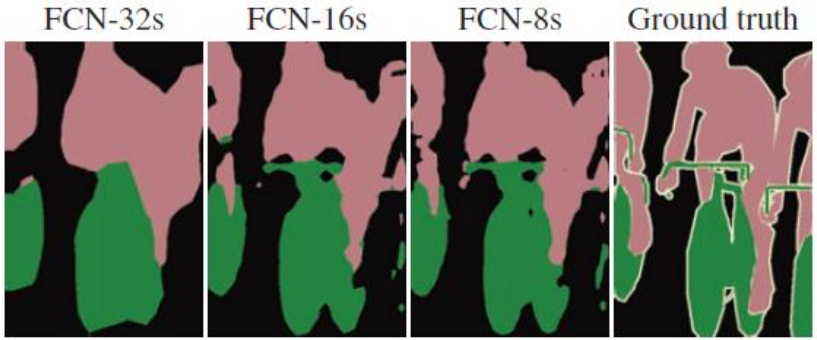


Learnable upsampling!

Long, Shelhamer, and Darrell, “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015

Semantic Segmentation





Ref:https://towardsdatascience.com/review-fcn-semantic-segmentation-eb8c9b50d2d1

Metrics of segmentation

Metric:

Let n_{ij} be the number of pixels of class i predicted as class j, let $t_i = \sum_j n_{ij}$ be the total number of pixels of class i, and let N be the number of classes, and let M=$\sum_i t_i$ be the number of total pixels

$$\text{pixel acc.} = \frac{\sum_i n_{ii}}{\sum_i t_i}. \qquad \text{mIoU} = \frac{1}{N}\sum_i \frac{n_{ii}}{t_i + \sum_j n_{ji} - n_{ii}}$$

Pixel accuracy = $\frac{\sum_i n_{ii}}{\sum_i t_i} = \frac{(17+1+1)}{(17+1+1)+(1+1+1)+(1+1+1)} = 0.76$

mIOU=(0.81+0.2+0.2)=0.4

0	0	0	0	0
0	0	1	2	0
0	0	1	2	0
0	0	1	2	0
0	0	0	0	0

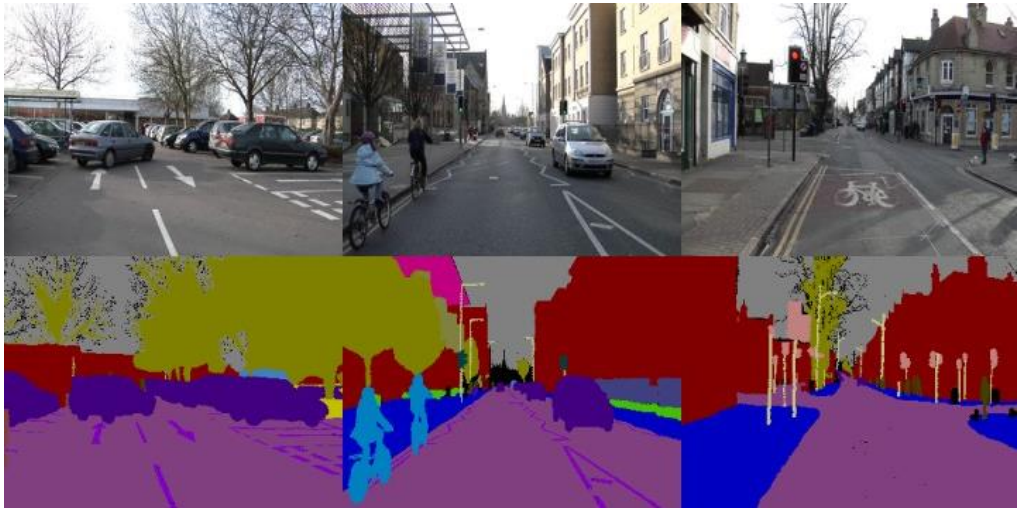
Ground-Truth

0	0	0	0	0
0	0	0	0	0
0	1	1	1	0
0	2	2	2	0
0	0	0	0	0

Segmentation Result

Dataset of our homework Q2: CamVid

- Original classes: 31+1(void)
- Simplified classes: 11+1(void)
- Images: 367 for training, 101 for validation, 233 for testing.



The Ground-Truth of Segmentation

- The segmentation ground-truth of every image is very dark because the range is between 0 & 11.

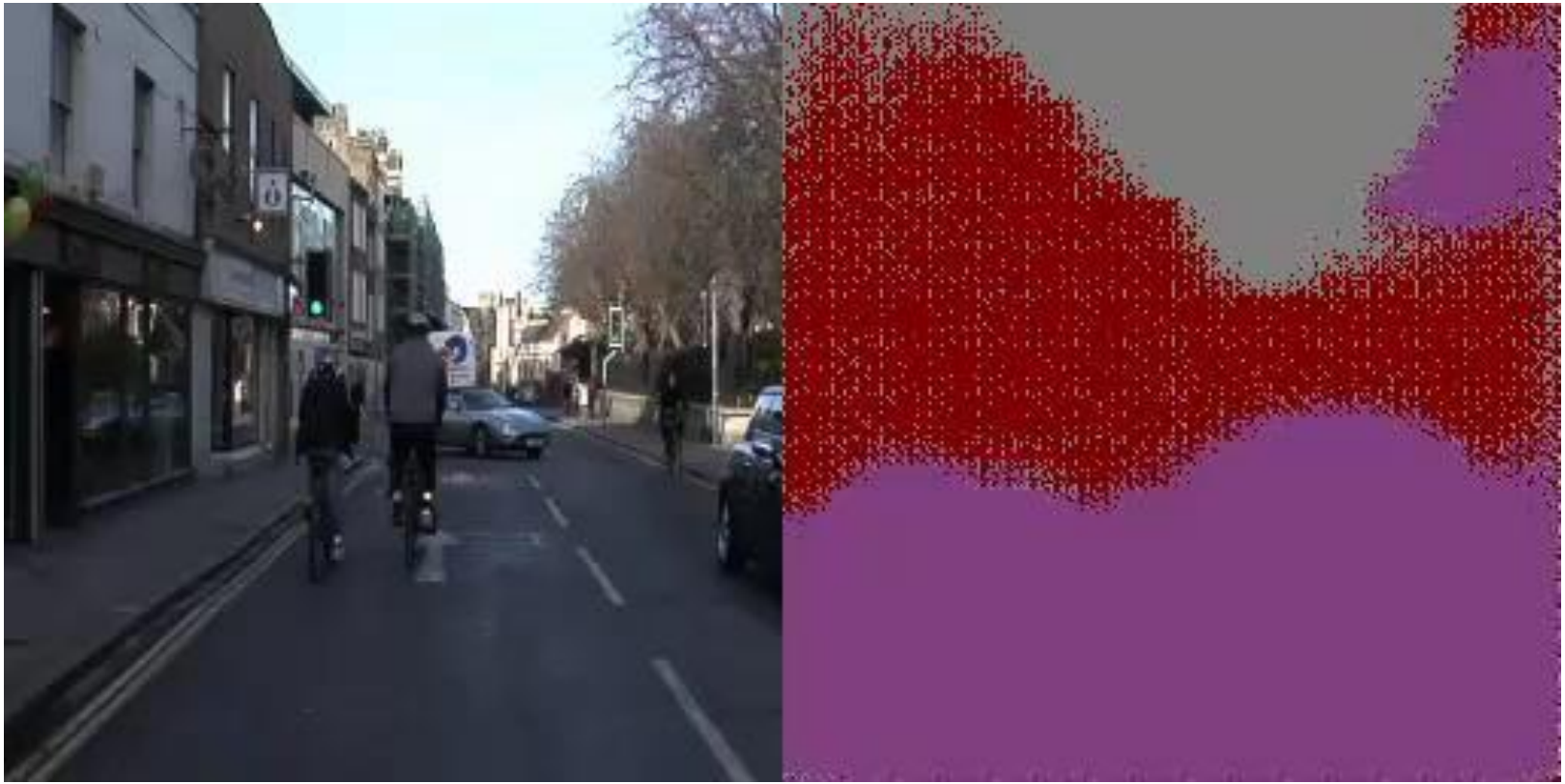
Color code of CamVid

- "Sky" 128 128 128
- "Building" 128 0 0
- "Pole" 192 192 128
- "Road" 128 64 128
- "Pavement" 0 0 192 (the color of sidewalk)
- "Tree" 128 128 0
- "SignSymbol" 192 128 128
- "Fence" 64 64 128
- "Car" 64 0 128
- "Pedestrian" 64 64 0
- "Bicyclist" 0 128 192
- **“void" 0 0 0**

Further class number reduction

- Class-0: Sky
- Class-1: Building, Pole, Road, Pavement, Tree, SignSymbol, Fence & **void**
- Class-2: Car, Pedestrian, Bicyclist

Expected results in 11-class version



- Q2-1. Please try to “eliminate” the skip-connection so the output of convolution layers of FCN8s will be directly upsampled for 32x. Please report pixel accuracy and mIOU before and after. (10 pts)

- Q2-2. Please discuss the results of Q2-1. (10 pts)

  ps. Is skip connection quantitatively beneficial?

- Q2-3. Please try to further reduce the number of classes from 11 to 3 and report the pixel accuracy & mIOU of FCN8s. (10 pts)

  ps. Please don’t create another copy of dataset

- Q2-4. Please discuss the results of Q2-3. Was mIOU increased when the number of classes reduce? Please explain why! (10 pts)

The structure of hw4.zip

- hw4.zip contains:
- HW4_1_Transfer_Learning_in_CNN_PyTorch.ipynb
- HW4_2_Semantic_Segmentation_PyTorch.ipynb
- HW_4_tutorial.pdf
- heavy_makeup_CelebA/train/heavy_makeup
- heavy_makeup_CelebA/train/no_heavy_makeup
- heavy_makeup_CelebA/val/heavy_makeup
- heavy_makeup_CelebA/val/no_heavy_makeup
- CamVid/trainannot
- CamVid/train
- CamVid/train.csv
- CamVid/valannot
- CamVid/val
- CamVid/val.csv
- CamVid/results_comparision

The structure of your turned-in file

- hw4_107062566.zip should be:
- hw4/
  - Q1-8.ipynb
  - Q2-1.ipynb
  - Q2-3.ipynb
  - report.pdf

PS. Don’t upload the training data of Q1 & Q2 again!

requirements

- Don’t try to prolong the training epochs in every case because TA can’t justify your submitted scripts when you claim that your best results could be achieved at 100k epochs!
- 3 ipynb files should be able to be executed directly by pressing “Runtime/RunAll”
- If your code is not executable, you will get no point.

Thank you!