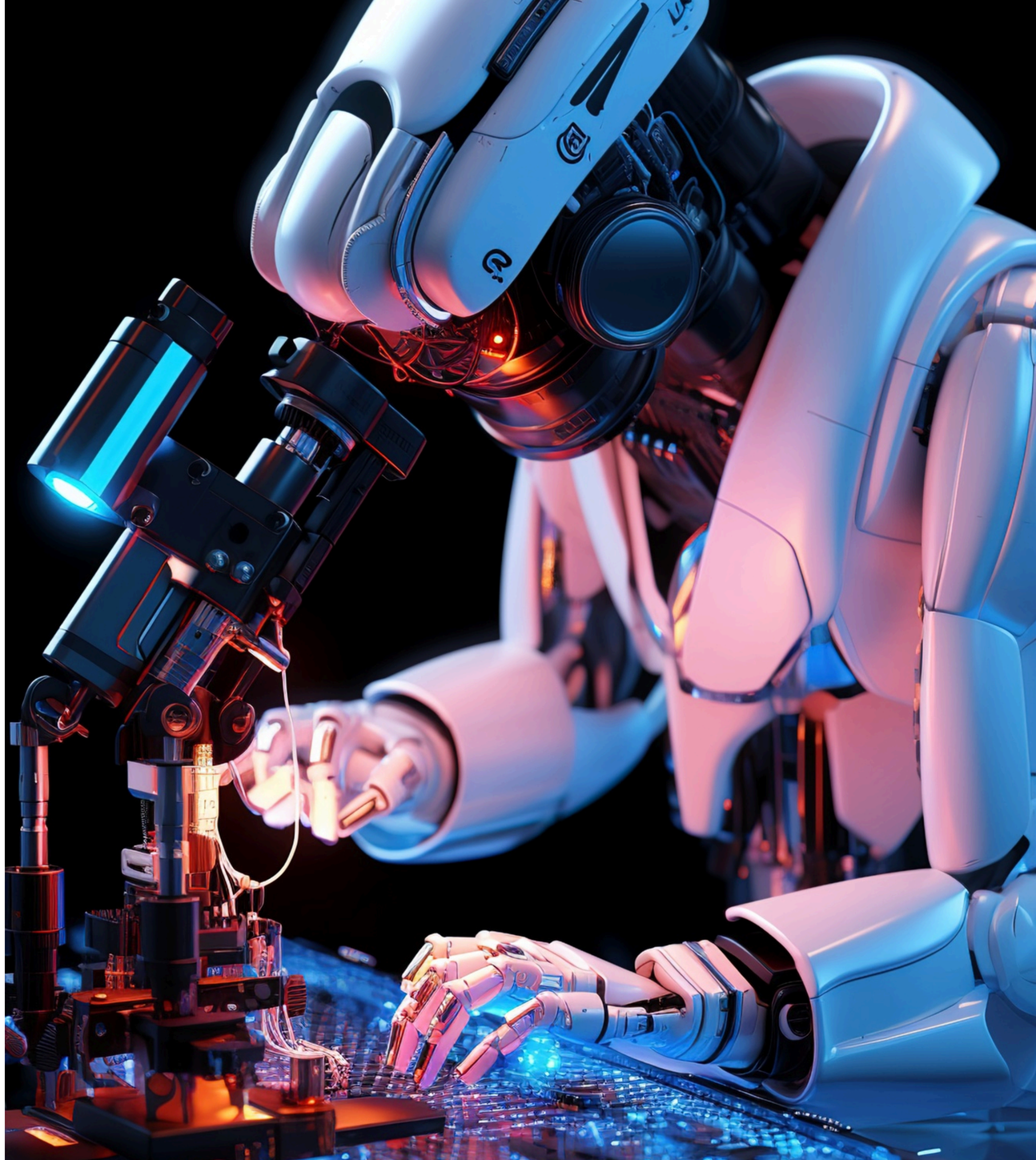
Facial Landmark Detection

Kaiyang Zhou

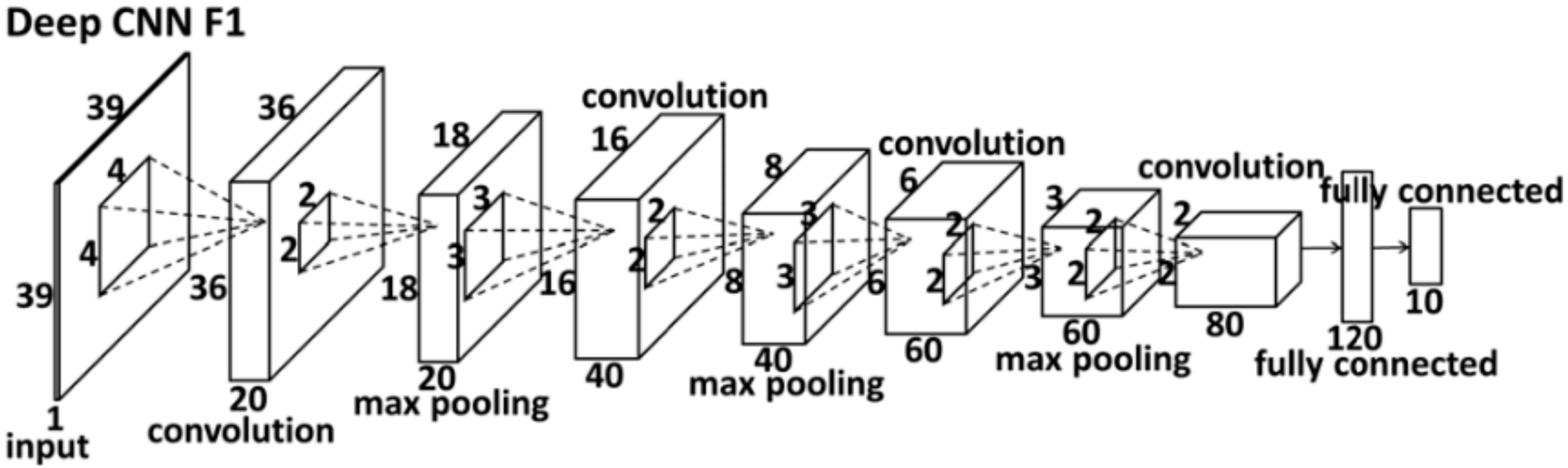
https://kaiyangzhou.github.io/

香港浸會大學
HONG KONG BAPTIST UNIVERSITY

DEPARTMENT OF
COMPUTER SCIENCE
計算機科學系



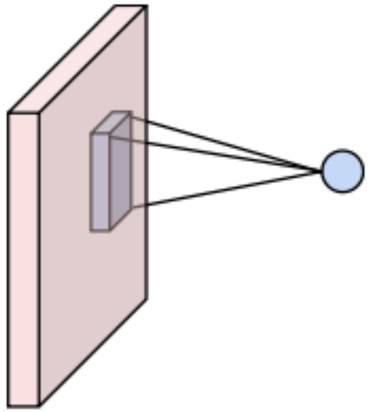
Convolutional Neural Network (CNN)



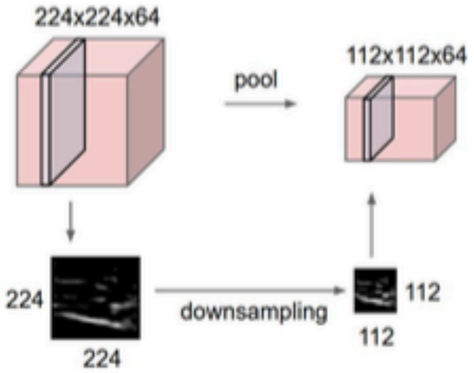
Sun et al. Deep Convolutional Network Cascade for Facial Point Detection. CVPR 2013.

Convolutional Neural Network (CNN)

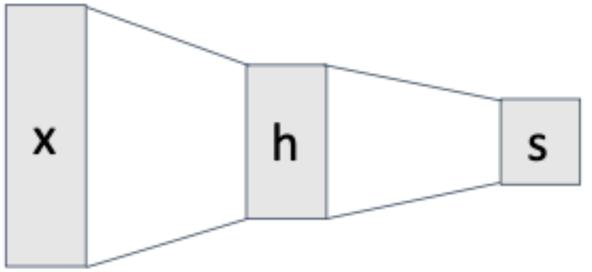
Convolution Layers



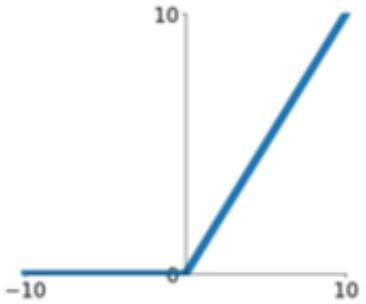
Pooling Layers



Fully-Connected Layers



Activation Function



Normalization

$$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}$$

https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

CLASS torch.nn.Conv2d(*in_channels*, *out_channels*, *kernel_size*, *stride=1*, *padding=0*, *dilation=1*, *groups=1*, *bias=True*, *padding_mode='zeros'*, *device=None*, *dtype=None*) [SOURCE]

CLASS torch.nn.MaxPool2d(*kernel_size*, *stride=None*, *padding=0*, *dilation=1*, *return_indices=False*, *ceil_mode=False*) [SOURCE]

CLASS torch.nn.Linear(*in_features*, *out_features*, *bias=True*, *device=None*, *dtype=None*) [SOURCE]

CLASS torch.nn.BatchNorm2d(*num_features*, *eps=1e-05*, *momentum=0.1*, *affine=True*, *track_running_stats=True*, *device=None*, *dtype=None*) [SOURCE]

CLASS torch.nn.ReLU(*inplace=False*) [SOURCE]

https://pytorch.org/

```python
import torch.nn as nn
import torch.nn.functional as F


class Net(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv1 = nn.Conv2d(3, 6, 5)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(6, 16, 5)
        self.fc1 = nn.Linear(16 * 5 * 5, 120)
        self.fc2 = nn.Linear(120, 84)
        self.fc3 = nn.Linear(84, 10)

    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = torch.flatten(x, 1) # flatten all dimensions except batch
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return x


net = Net()
```

https://pytorch.org/

ResNet

Deep residual learning for image recognition

K He, X Zhang, S Ren, J Sun - Proceedings of the IEEE …, 2016 - openaccess.thecvf.com

… **Deeper** neural networks are more difficult to train. We present a **residual learning** framework to ease the training of networks that are substantially **deeper** than those used previously. …

☆ Save Cite Cited by 200137 Related articles All 76 versions »

Optimization problem:
deeper NNs are harder to train

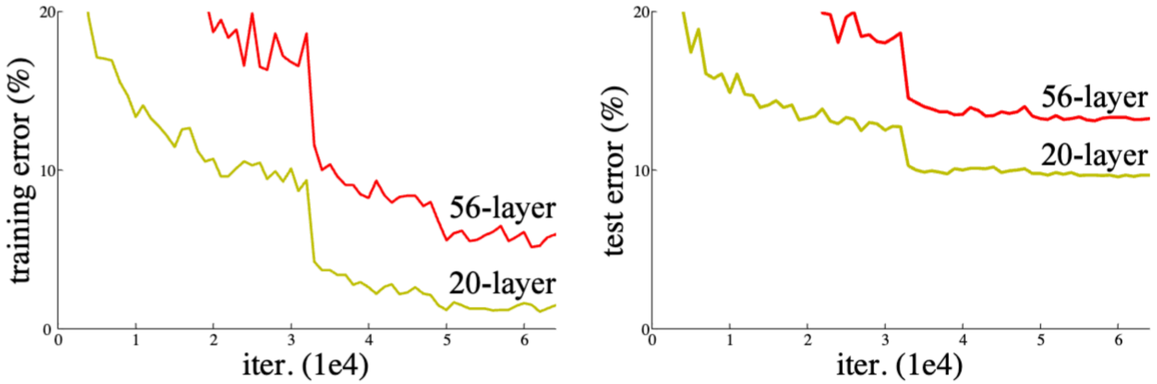
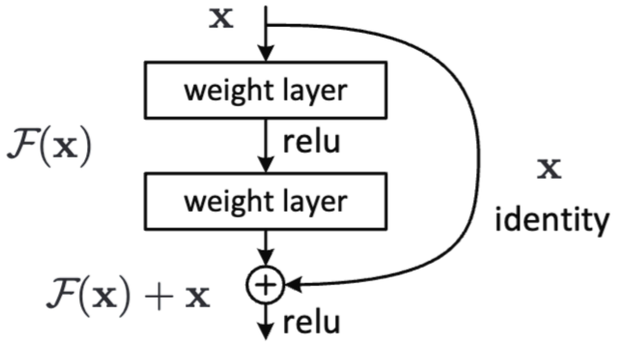
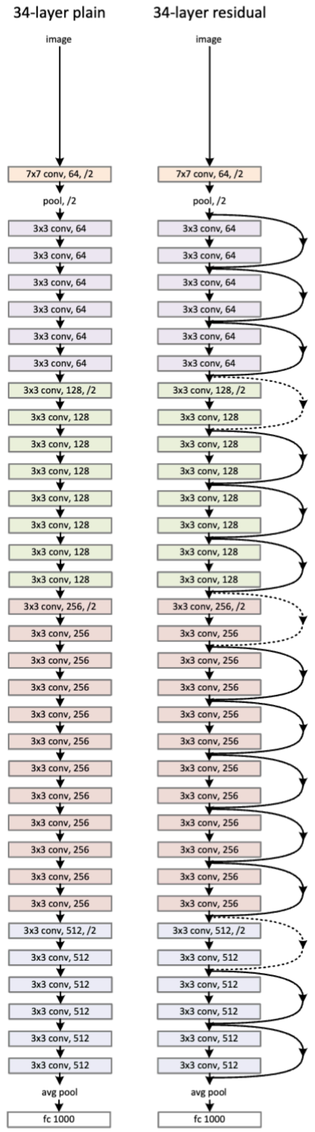


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.





MobileNet

Mobilenets: Efficient convolutional neural networks for mobile vision applications
AG Howard, M Zhu, B Chen, D Kalenichenko… - arXiv preprint arXiv …, 2017 - arxiv.org
… models called **MobileNets** for mobile and embedded vision applications. **MobileNets** are …
We then demonstrate the effectiveness of **MobileNets** across a wide range of applications …
☆ Save 99 Cite Cited by 23475 Related articles All 10 versions »

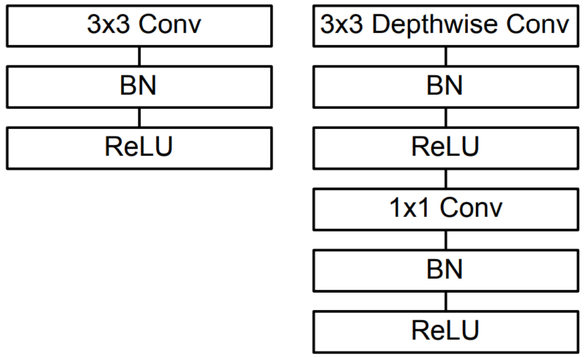
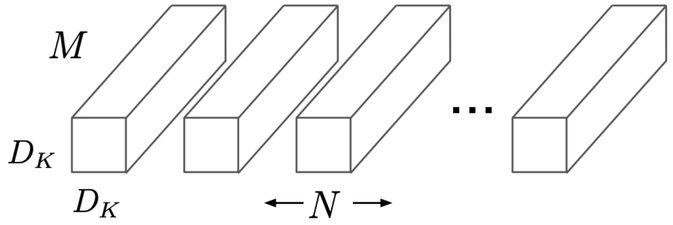
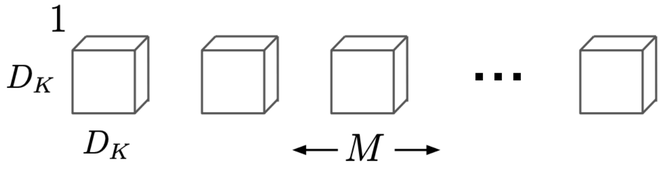


Figure 3. Left: Standard convolutional layer with batchnorm and ReLU. Right: Depthwise Separable convolutions with Depthwise and Pointwise layers followed by batchnorm and ReLU.

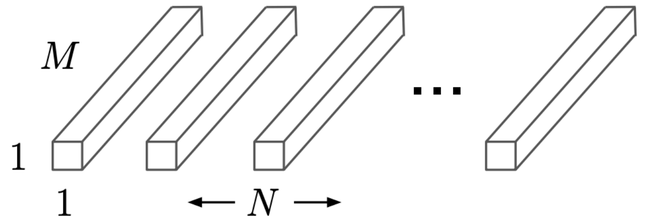
3x3 depthwise separable convolutions consume 8 to 9 times less computation, with only a small performance drop



(a) Standard Convolution Filters



(b) Depthwise Convolutional Filters



(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

Figure 2. The standard convolutional filters in (a) are replaced by two layers: depthwise convolution in (b) and pointwise convolution in (c) to build a depthwise separable filter.

The key idea is to decompose a DxDxM filter into a DxDx1 filter and a 1x1xM filter

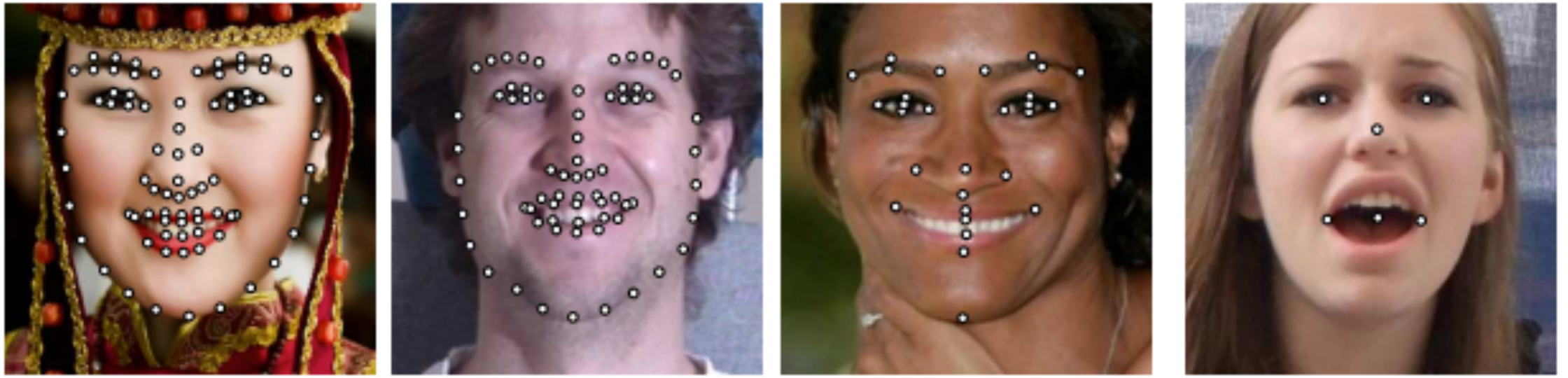
Facial landmark detection

source: https://github.com/1adrianb/face-alignment

Prominent facial features:
- eyebrows
- eyes
- nose
- mouth
- chin

Annotations

- Facial key points
- Interpolated landmarks (face contour)



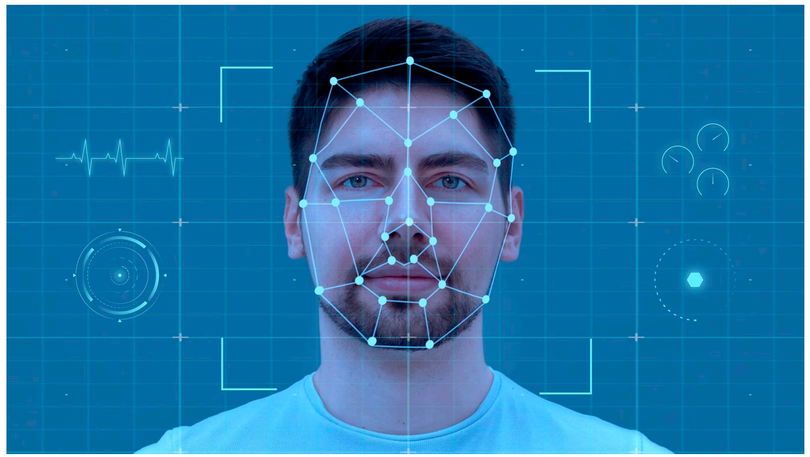
(a) Helen (b) Multi-PIE (c) LFPW (d) IBUG, AFW

Smith and Zhang. Collaborative Facial Landmark Localization for Transferring Annotations Across Datasets. ECCV 2014.

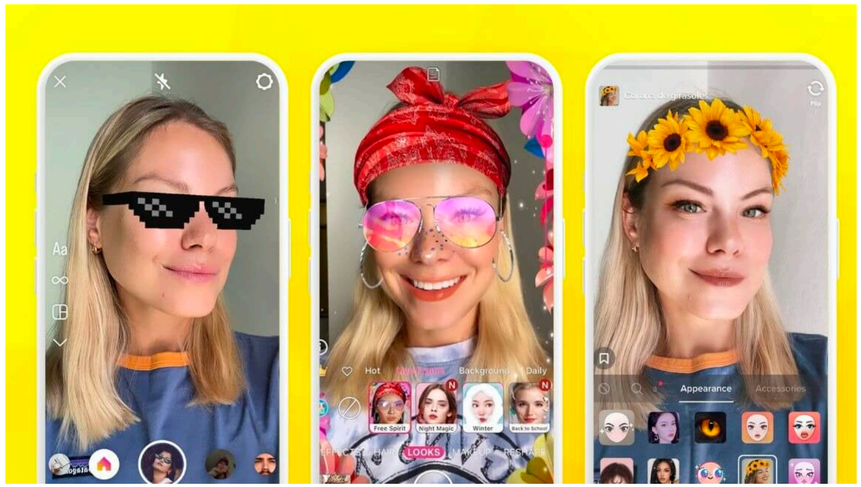
Applications

Face alignment

Facial expression analysis



Driving assistant



Entertainment

Face reconstruction

Challenges

Wu et al. "Look at boundary: A boundary-aware face alignment algorithm." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

Evaluation

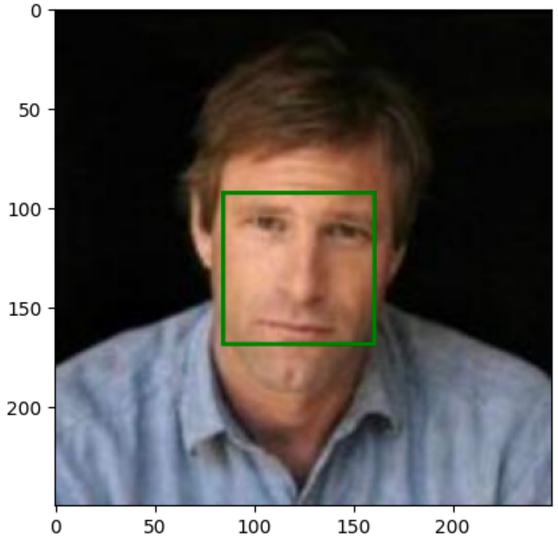
- Detected locations: $d_i = \{d_{x,i}, d_{y,i}\}$ & ground truth: $g_i = \{g_{x,i}, g_{y,i}\}$
- Detection error: $e = \frac{1}{N}\sum_{i=1}^{N} ||d_i - g_i||_2$ (there is scale issue with different face sizes)
- Inter-ocular distance. g_{le}: left pupil centre. g_{re}: right pupil centre.
- Normalised detection error: $e = \frac{1}{N}\sum_{i=1}^{N} \frac{||d_i - g_i||_2}{||g_{le} - g_{re}||_2}$

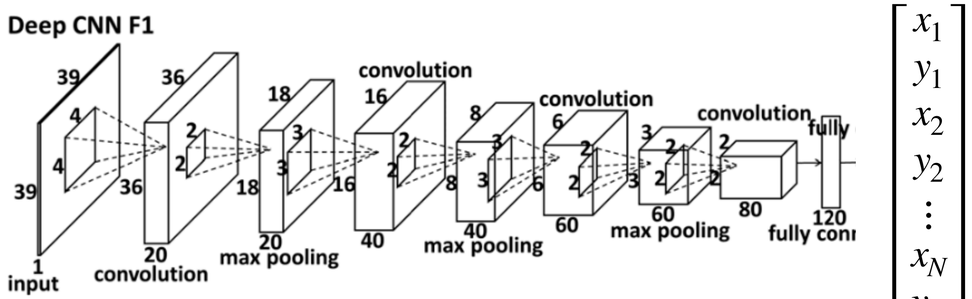
Jin et al. Pixel-in-Pixel Net: Towards Efficient Facial Landmark Detection in the Wild. IJCV 2021.

A deep learning-based approach

Learn a function to map pixels to 2D coordinates—a regression problem

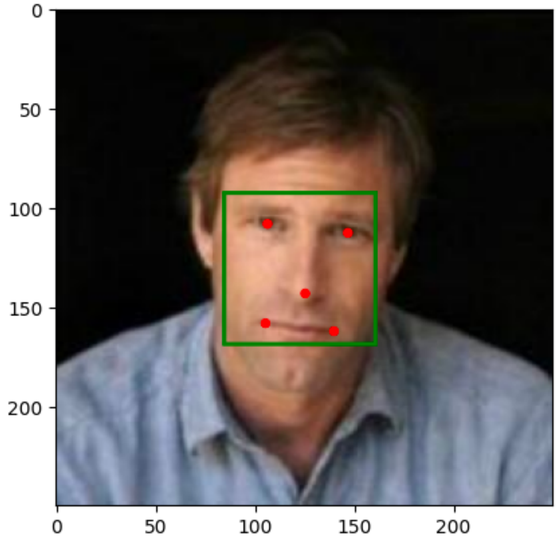
Input: face region





$$\begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ x_N \\ y_N \end{bmatrix}$$

Output: landmarks

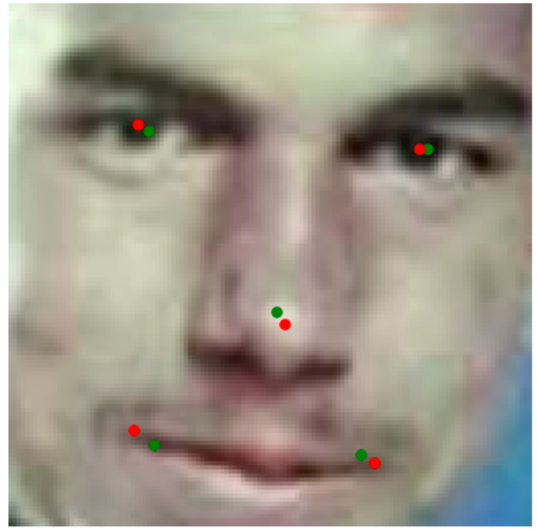


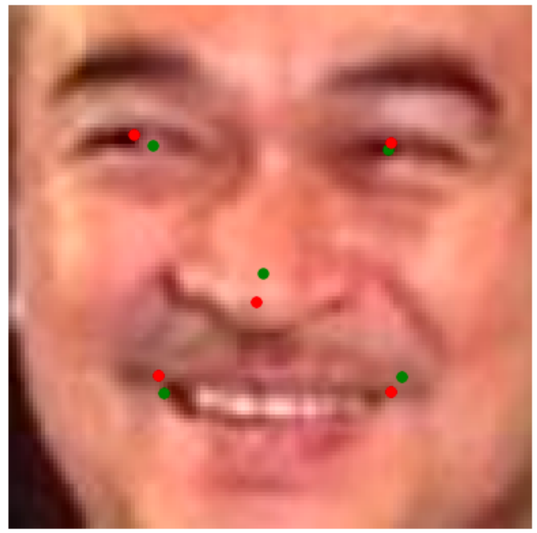
Mean squared error loss: $\arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} ||\mathbf{d}_i - \mathbf{g}_i||_2$

Case study

1. Download the dataset
   - http://mmlab.ie.cuhk.edu.hk/archive/CNN/data/train.zip
   - 5,590 LFW images and 7,876 web images
   - train: 10,000; test: 3,466; each line: image name, bbox (4), landmarks (5x2)
2. Analyse the dataset (todo: print statistics, visualise landmarks)
3. Build train & test data loaders
4. Build model (CNN), optimiser, and loss
5. Build train loop
6. Evaluate the model (todo: compute normalised error, visualise preds & labels)

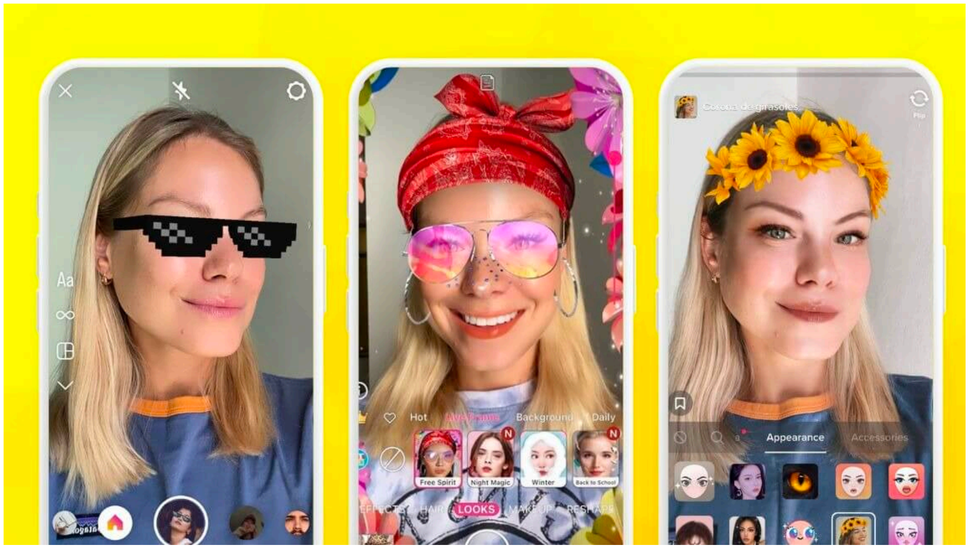
green: prediction
red: ground truth





Exercise

1. Download the dataset
   - https://wywu.github.io/projects/LAB/WFLW.html
2. Follow the same steps mentioned previously to develop a facial landmark detection model using the current dataset
3. Build a face filters demo



Further reading

- Wu and Ji. "Facial landmark detection: A literature survey." International Journal of Computer Vision 127.2 (2019): 115-142.
- Ioffe and Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. ICML’15.
- He et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
- Howard et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.

Happy Coding