Department of Computing

Imperial College of Science, Technology and Medicine

Popular Network Architectures (& BatchNorm)

LeNet-5 [4], MNIST, AlexNet [3], ImageNet, VGG [5], Inception (GoogleLeNet), BatchNorm [2], ResNet [1], DenseNet, Squeeze-Excite Net U-Net, Data Augmentation

Author: Anton Zhitomirsky

Contents

1 LeNet-5

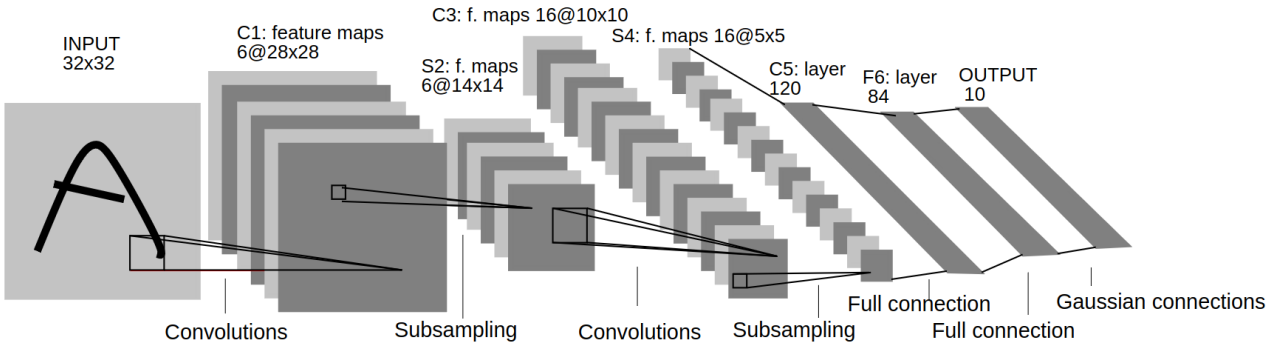

Figure 1: Architecture of LeNet-5 a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical [4]

```python
# from https://pytorch.org/tutorials/beginner/blitz/neural_networks_tutorial.html
import torch
import torch.nn as nn
import torch.nn.functional as F


class Net(nn.Module):

    def __init__(self):
        super(Net, self).__init__()
        # 1 input image channel, 6 output channels, 5x5 square convolution
        # kernel
        self.conv1 = nn.Conv2d(1, 6, 5)
        self.pool1 = nn.MaxPool2d(2)
        self.conv2 = nn.Conv2d(6, 16, 5)
        self.pool2 = nn.MaxPool2d(2)
        # an affine operation: y = Wx + b
        self.fc1 = nn.Linear(16 * 5 * 5, 120)  # 5*5 from image dimension
        self.fc2 = nn.Linear(120, 84)
        self.fc3 = nn.Linear(84, 10)

    def forward(self, x):
        x = self.conv1(x)
        x = F.relu(x)
        x = self.pool1(x)
        x = self.conv2(x)
        x = F.relu(x)
        x = self.pool2(x)

        x.torch.flatten(x,1) # flatten all dimensions except the batch dimension

        x = self.fc1(x)
        x = F.relu(x)
        x = self.fc2(x)
        x = F.relu(x)
        x = self.fc3(x)

        return x
```

code/LetNet.py

LeNet [4] was initially designed for low-resolution, black and white image recognition, specifically for digits. It demonstrated that CNNs could reliably perform both tasks of object localization and recognition (on low-resolution black and white images).

The last steps of the LeNet-5 architecture employ fully connected layers to convert features into final predictions. The scenario of MNIST data didn't matter, because of its small number of output classes. This *complicates the architecture when scaling up, influencing both computational resources and architecture*.

References

[1] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385.

[2] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. arXiv: 1502.03167 [cs.LG]. URL: https://arxiv.org/abs/1502.03167.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet classification with deep convolutional neural networks". In: *Commun. ACM* 60.6 (May 2017), pp. 84–90. ISSN: 0001-0782. DOI: 10.1145/3065386. URL: https://papers.nips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a6
Paper.pdf.

[4] Y. Lecun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: 10.1109/5.726791. URL: http://vision.stanford.edu/cs598_spring07/papers/Lecun98.pdf.

[5] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV]. URL: https://arxiv.org/pdf/1409.1556.pdf.