BMEG3105 **Fall 2025**

Data analytics for personalized genomics and precision medicine
Lecture 20: CNN

November 14, 2025

Lecturer: Prof. LI Yu *Scribe: LI Haoyuan*

20.1 Recap

20.1.1 Artificial intelligence VS Machine learning VS Deep learning

- Artificial Intelligence (AI)
  Any techniques which enable computers to mimic human behavior
- Machine Learning (ML)
  A subset of AI, which effectively perform a specific task without using explicit instructions, relying on patterns and inference from the data
- Deep Learning (DL)
  A subset algorithms of ML which takes advantage of multi-layer neural networks

20.1.2 From LR to deep neural networks

To resolve complicated problems

- Increase the number of nodes
- Increase the number of layers
- Add non-linear function

Fully-connected layers

- A general function approximator
- We can approximate any function (relation) if we have enough nodes and layers
- Universal approximation theorem

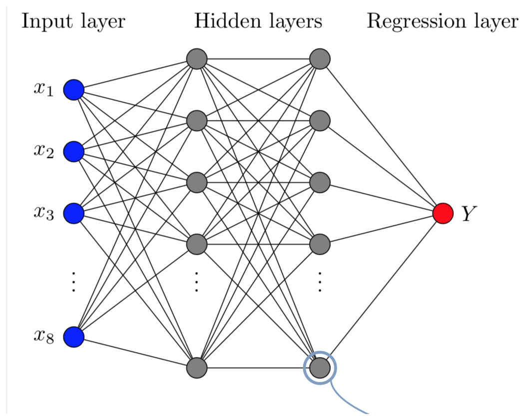

Figure 20.1: Fully-connected Layer

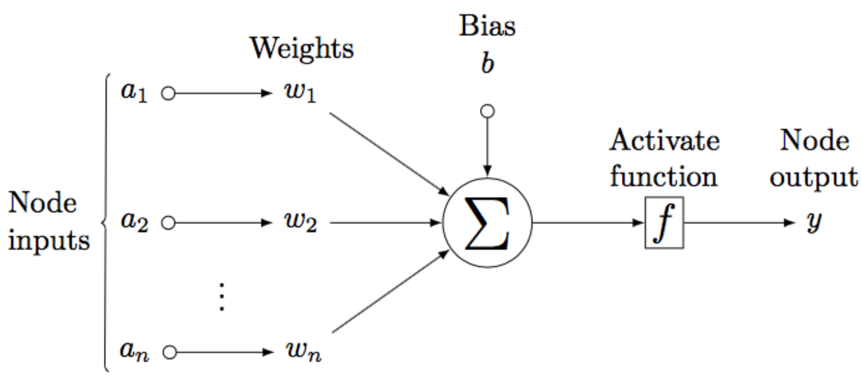

Figure 20.2: The flow chart of the simulation process in each node

20.1.3 Deep neural networks

The function is much more complicated

The number of parameters is very large

We may use it to resolve complex problems with a huge amount of data

20.1.4 The problems of fully-connected networks?

- How to determine the number of nodes and layers?
  - As many as possible? Of course not!
- Storage
- Running time

  - Embedded systems
- Hard to train
- Prior knowledge is ignored
- Overfitting
- ......

20.1.5 What are the properties of objects in the image?

- Translation invariance
  - Capture the patch information, no matter where it is
- Locality
  - Focus on local regions first
  - Should be aggregated later

20.1.6 Convolutional layers

"Modern" Deep Learning: Hierarchical Representation Learning (Feature extraction)

"Classical" Fully-connected Neural Networks (Classification)

20.2 Deep learning and biomedical imaging

20.2.1 What can we get from convolutional layers?

An example:

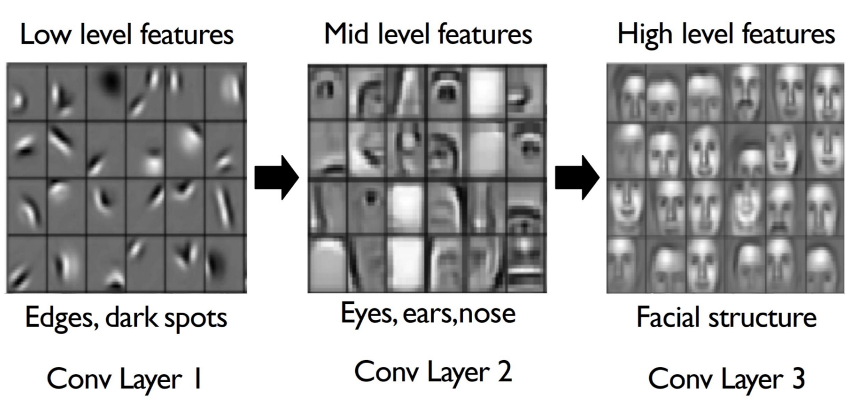

Figure 20.3: Features at different levels

Low level features (Conv Layer 1) $\rightarrow$ Edges, dark spots, etc.

Mid level features (Conv Layer 2) $\rightarrow$ Eyes, ears, nose, etc.

High level features (Conv Layer 3+) $\rightarrow$ Facial structure, whole object

Spatial patterns

20.2.2 One convolution operation

Example:

$$\begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix} * \begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix} = \begin{bmatrix} 19 & 25 \\ 37 & 43 \end{bmatrix}$$

Iuput * Kernel = Output

Cross-correlation

Parameters = Filter = Weights = Kernel

20.2.3 How to do convolution?

Share parameters

- Alleviate the overfitting problem
- Detect translation invariant features
- Locality

20.2.4 Number of trainable parameters

First-layer filter size: 3 * 3

Bias: 1

Number of filters: 6

Number of trainable parameters in the first layer: (3*3+1)*6 = 60

Filter size: Usually 3 by 3 or 5 by 5.

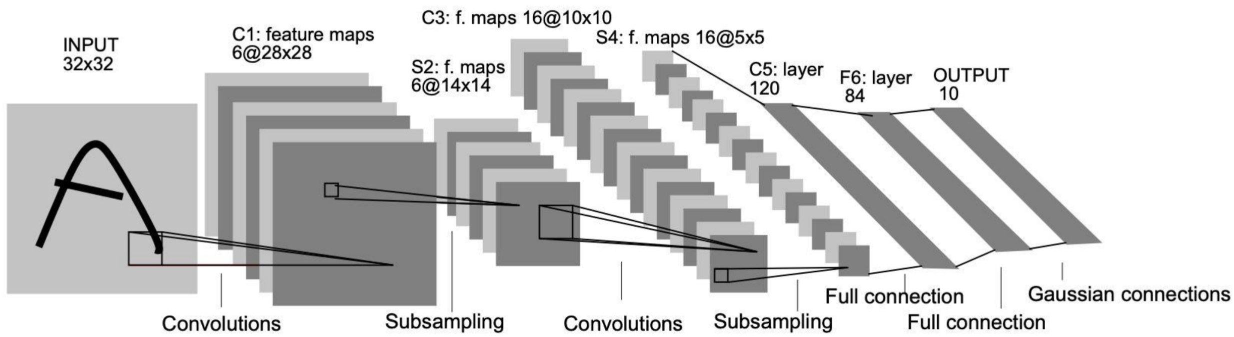

Figure 20.4: Number of trainable parameters in CNN

20.3 More discussion of convolutional layer

20.3.1 How to deal with the boundary?

The dimension of the output matrix with values is slightly smaller than the original input matrix because of the dimension of the filter.

Padding is the addition of (typically) 0-valued pixels on the borders of an image. This is done so that the border pixels are not undervalued (lost) from the output because they would ordinarily participate in only a single receptive field instance.

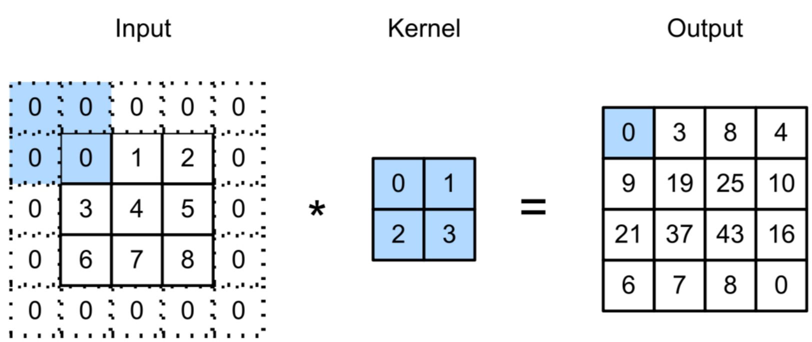

Figure 20.5: Padding

20.3.2 Stride

The stride is the number of pixels that the analysis window moves on each iteration.

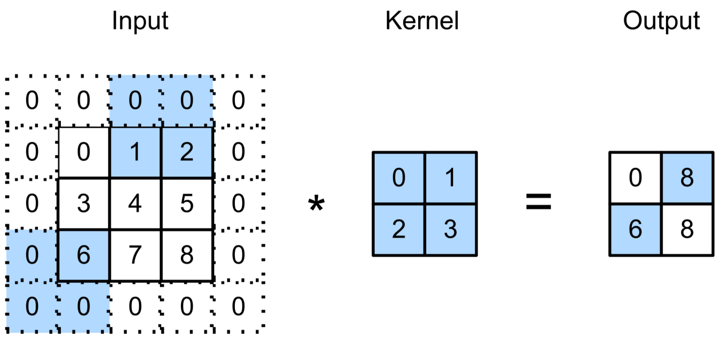

Figure 20.6: Stride

Column stride = 2

Row stride = 3

20.3.3 Pooling

Max pooling

Average pooling

(Also with padding and stride)

Max pooling is typically used, often with a 2x2 dimension. This implies that the input is drastically down-sampled, reducing processing cost.

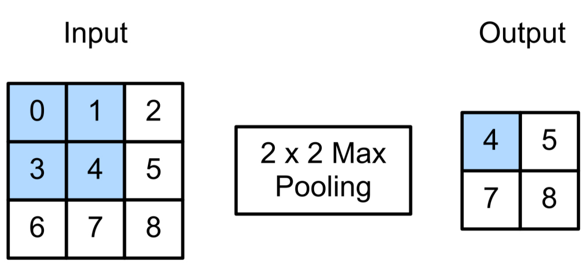

Figure 20.7: Pooling

20.3.4 Build models for real-life healthcare problems

Example: (Esteva et al., 2017, Nature)

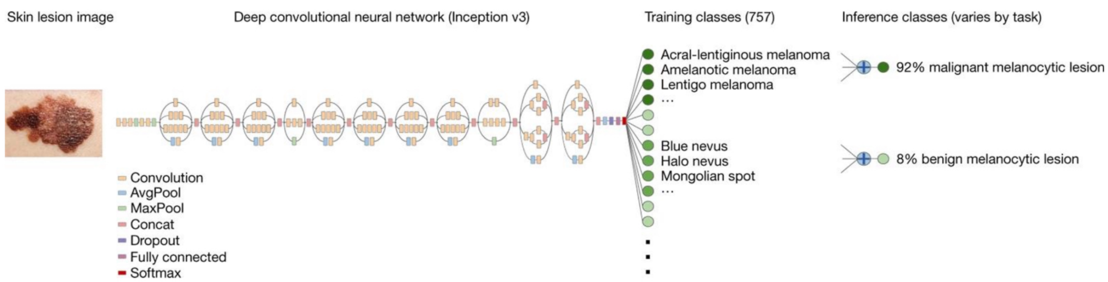

Figure 20.8: Dermatologist-level classification of skin cancer with deep neural networks

20.3.5 Deep learning in Python

- Used to be in chaos
- Now, Pytorch and Tenserflow
  - https://github.com/pytorch/examples
  - Of course, there are other packages

Reference

https://en.wikipedia.org/wiki/Convolutional_neural_network

Esteva, Andre, et al. "Dermatologist-Level Classification of Skin Cancer with Deep Neural Networks." Nature (London), vol. 542, no. 7639, 2017, pp. 115–18, https://doi.org/10.1038/nature21056.