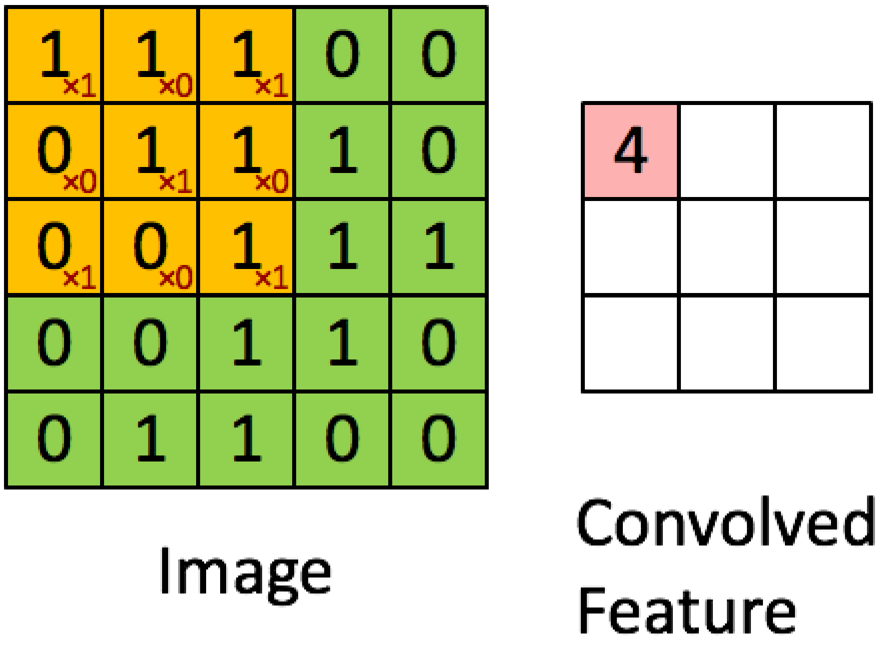
Herbert Wertheim College of Engineering
Florida Institute for Cybersecurity Research
Preparer: Sean Taheri

PO BOX
116200216
Larsen Hall
Gainesville, FL
32611-2026
Phone: (352) 294-3945
info@eng.ufl.edu

Convolutional Neural Network (CNN) Basics and Implementation

1. CNNs → Major breakthroughs in Image Classification and are the core of most Computer Vision systems today.

2. Convolution → Thinking of it as a sliding window function applied to a matrix!



(a) Imagine that the matrix on the left represents a black and white image.

(b) Each entry corresponds to one pixel, 0 for black and 1 for white (typically it's between 0 and 255 for grayscale images).

(c) The sliding window is called a *kernel*, *filter*, or *feature detector*.

* Here we use a 3×3 filter; multiply its values element-wise with the original matrix, then sum them up.

* To get the full convolution we do this for each element by sliding the filter over the whole matrix.

* Applications:

    - Averaging each pixel with its neighboring values blurs an image.

    - Taking the difference between a pixel and its neighbors detects edges.

3. CNNs → Just several layers of convolutions with *nonlinear activation functions* like **ReLU** or **tanh** applied to the results.

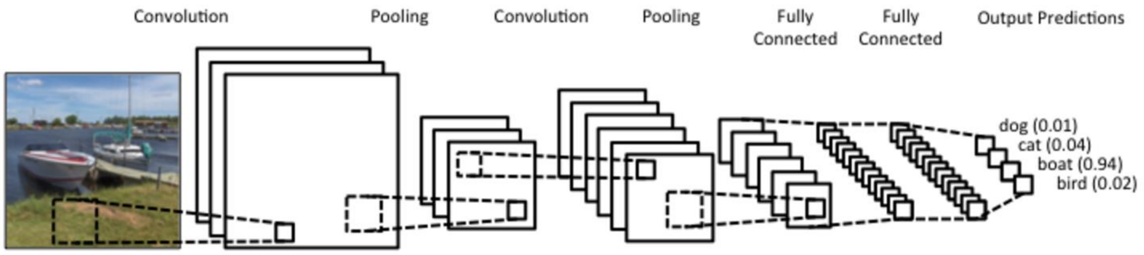


Figure 1: A CNN Architecture.

4. A traditional feedforward neural network → Connecting each input neuron to each output neuron in the next layer → A fully connected layer, or affine layer.

5. In CNNs → Using convolutions over the input layer to compute the output instead of traditional feedforward approach. What are the other specifications:

* Local connections → Each region of the input is connected to a neuron in the output.

* Each layer applies different filters, typically hundreds or thousands like the ones showed above, and combines their results.

* Pooling (subsampling) layers → Provide you invariance to translation, rotation and scaling.

* Training Phase → The CNN automatically learns the values of its filters based on the task you want to perform.

Example: In Image Classification a CNN may learn to detect edges from raw pixels in the first layer, then use the edges to detect simple shapes in the second layer, and then use these shapes to deter higher-level features, such as facial shapes in higher layers. The last layer is then a classifier that uses these high-level features.

- You build edges from pixels, shapes from edges, and more complex objects from shapes.

6. <u>CNN Computation Aspects</u>:

1) **Location Invariance** → You don't really care *where* the object occurs!
2) (Local) **Compositionality →** Each filter *composes* a local patch of lower-level features into higher-level representation.

7. CNN Hyperparameters:

* Narrow Vs. Wide convolution → Using zero-padding for edges of image data in applying the convolution process. It is also called the Wide Convolution. When there is no zero-padding, the process is called narrow convolution.

* Stride Size → The amount you want to shift your filter at each step.

* Pooling Layers → They subsample their input. The most common way to do pooling it to apply a MAX operation to the result of each filter. Pooling can be done over a window and not the entire matrix. It provides a fixed size output matrix (no matter what the filter and the image size is), which typically is required for classification. This allows you to use variable size sentences, and variable size filters, but always get the same output dimensions to feed into a classifier. It also reduces the output dimensionality but (hopefully) keeps the most salient information. This means detecting the specific features. In other words, you are losing global information about locality, but you are keeping local information captured by your filters

* Channels → They are *different "views" of your input data. You can apply convolutions across channels, either with different or equal weights.*

8. Possible Architectures:

* LeNet-5: Its convolutional layers use a subset of the previous layer's channels for each filter to reduce computation and force a break of symmetry in the network. The subsampling layers use a form of average pooling.

Paper: Gradient-based learning applied to document recognition

* AlexNet: The general architecture is quite similar to LeNet-5, although this model is considerably larger.

Paper: ImageNet Classification with Deep Convolutional Neural Networks

* DenseNet: Its main idea is: **it may be useful to reference feature maps from earlier in the network**. Thus, each layer's feature map is concatenated to the input of *every successive layer* within a dense block. This allows later layers within the network to *directly* leverage the features from earlier layers, encouraging feature reuse within the network. Concatenating feature-maps learned by *different layers* increases variation in the input of subsequent layers and improves efficiency.

The idea behind dense convolutional networks is simple: **it may be useful to reference feature maps from earlier in the network**. Thus, each layer's feature map is concatenated to the input of *every successive layer* within a dense block. This allows later layers within the network to *directly* leverage the features from earlier layers, encouraging feature reuse within the network. The authors state, "Concatenating feature-maps learned by *different layers* increases variation in the input of subsequent layers and improves efficiency."

Paper: Densely Connected Convolutional Networks

9. Educational Videos:

Video 01: CNN Introduction:

- https://www.youtube.com/watch?v=aircAruvnKk
- https://www.youtube.com/watch?v=vT1JzLTH4G4
- https://www.youtube.com/watch?v=FmpDIaiMIeA
- https://www.youtube.com/watch?v=FTr3n7uBIuE

Video 02- CNN Architectures (AlexNet and LeNet):

- https://www.youtube.com/watch?v=T7t1uTzh3oI
- https://www.youtube.com/watch?v=pgXVV1Ypdyo

Video 03 – DenseNet Architecture:

- https://www.youtube.com/watch?v=-W6y8xnd--U
- https://www.youtube.com/watch?v=fe2Vn0mwALI
- https://www.youtube.com/watch?v=oV4YBitzXKw

Video 04 – Tensorflow and Keras:

- https://www.youtube.com/watch?v=wQ8BIBpya2k
- https://www.youtube.com/watch?v=WvoLTXIjBYU
- https://www.youtube.com/watch?v=BqgTU7_cBnk
- https://www.youtube.com/watch?v=oZikw5k_2FM
- https://www.youtube.com/watch?v=QfNvhPx5Px8
- https://www.youtube.com/watch?v=4eIBisqx9_g&t=115s
- https://www.youtube.com/watch?v=j_pJmXJwMLA

10. Sample Codes:

- Sample CNN: https://github.com/vzhou842/cnn-from-scratch
- All-CNN Implementation: https://github.com/PAN001/All-CNN
- Simple CNN: https://github.com/lucko515/cnn-tensorflow-keras
- Tensorflow 01: https://github.com/zotroneneis/tensorflow_deep_learning_models
- Tesnorflow 02: https://github.com/aymericdamien/TensorFlow-Examples
- Keras 01: https://github.com/keras-team/keras
- Keras 02: https://github.com/sagar448/Keras-Convolutional-Neural-Network-Python
- Text Recognition 01: https://github.com/sushant097/Handwritten-Line-Text-Recognition-using-Deep-Learning-with-Tensorflow
- Text Recognition 02: https://github.com/harshul1610/OCR
- Text Recognition 03: https://github.com/qjadud1994/CRNN-Keras
- Text Recognition 04: https://github.com/aayushsharma9/ocr-cnn
- Text Recognition 05: https://github.com/githubharald/SimpleHTR
- Text Recognition 06: https://github.com/gasparian/CRNN-OCR-lite
- Text Recognition 07: https://github.com/ikergarcia1996/Handwritten-Names-Recognition

11. References for Text Recognition:

- https://towardsdatascience.com/build-a-handwritten-text-recognition-system-using-tensorflow-2326a3487cd5
- https://medium.com/@arthurflor23/handwritten-text-recognition-using-tensorflow-2-0-f4352b7afe16
- https://nanonets.com/blog/attention-ocr-for-text-recogntion/
- https://www.digitalocean.com/community/tutorials/how-to-build-a-neural-network-to-recognize-handwritten-digits-with-tensorflow
- https://towardsdatascience.com/handwriting-recognition-using-tensorflow-and-keras-819b36148fe5
- https://keras.io/examples/image_ocr/
- https://missinglink.ai/guides/tensorflow/building-tensorflow-ocr-systems-key-approaches-and-tutorials/

12. LeNet Architecture and Implementation:

- https://medium.com/@mgazar/lenet-5-in-9-lines-of-code-using-keras-ac99294c8086
- https://www.pyimagesearch.com/2016/08/01/lenet-convolutional-neural-network-in-python/
- https://engmrk.com/lenet-5-a-classic-cnn-architecture/
- https://medium.com/@RaghavPrabhu/kaggles-digit-recogniser-using-tensorflow-lenet-architecture-92511e68cee1
- https://github.com/sujaybabruwad/LeNet-in-Tensorflow
- http://datahacker.rs/lenet-5-implementation-tensorflow-2-0/
- https://sumitbinnani.github.io/CarND-LeNet-Lab/

13. AlexNet Architecture and Implementation:

- https://engmrk.com/alexnet-implementation-using-keras/
- https://medium.com/datadriveninvestor/cnn-architecture-series-alexnet-with-implementation-part-ii-7f7afa2ac66a
- https://www.mydatahack.com/building-alexnet-with-keras/
- https://github.com/duggalrahul/AlexNet-Experiments-Keras
- https://ai-pool.com/m/alexnet-1564424451
- https://awesomeopensource.com/project/duggalrahul/AlexNet-Experiments-Keras
- http://www.michaelfxu.com/neural%20networks%20series/neural-networks-pt4-cnn-codes/
- https://rahulduggal2608.wordpress.com/2017/04/02/alexnet-in-keras/
- https://neurohive.io/en/popular-networks/alexnet-imagenet-classification-with-deep-convolutional-neural-networks/
- http://dandxy89.github.io/ImageModels/alexnet/
- https://medium.com/@joeyism/creating-alexnet-on-tensorflow-from-scratch-part-2-creating-alexnet-e0cd948d7b04
- http://datahacker.rs/tf-alexnet/
- https://kratzert.github.io/2017/02/24/finetuning-alexnet-with-tensorflow.html
- https://github.com/gholomia/AlexNet-Tensorflow
- https://github.com/amir-saniyan/AlexNet

14. DenseNet Architecture and Implementation:

- https://github.com/flyyufelix/DenseNet-Keras
- https://github.com/titu1994/DenseNet
- https://keras.io/api/applications/densenet/
- https://towardsdatascience.com/exploring-densenets-from-paper-to-keras-dcc01725488b
- https://pythonawesome.com/densenet-implementation-in-keras/
- https://medium.com/intuitionmachine/notes-on-the-implementation-densenet-in-tensorflow-beeda9dd1504
- https://github.com/YixuanLi/densenet-tensorflow
- https://github.com/taki0112/Densenet-Tensorflow
- https://github.com/leonndong/DenseNet-Tensorflow
- https://github.com/pudae/tensorflow-densenet
- https://github.com/sthalles/dense-net
- https://sthalles.github.io/densely-connected-conv-nets/
- https://keras.io/api/applications/
- http://www.gaohuang.net/papers/DenseNet-CVPR-Slides.pdf