MASL Project

27 January 2020

Convolutional Neural Network: theory and application

Noemi Benci

1. Introduction
2. Convolutional Neural Networks
   - Convolutional Layer
   - Pooling Layer
   - CNNs Architecture
3. Practical Example
4. Conclusion

Artificial Neural Networks

- Many neurons create layers;
- Different type of layers (Input, Hidden, Output);
- A connection is a weight;
- Many applications in real problems (Supervised and Unsupervised learning, Classification and Regression, many types of data..).

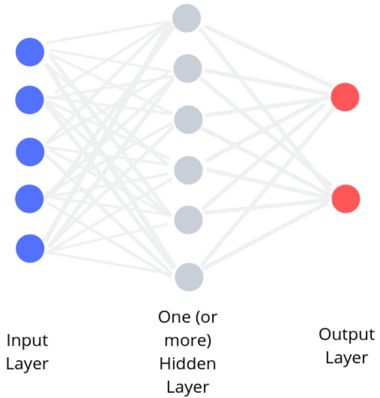
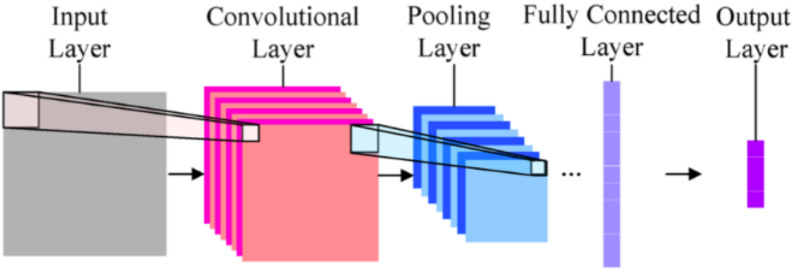


Figure 1: General structure of ANNs

Convolutional Neural Networks

General Characteristics:

- Multidimensional layers suitable for images;
- Characteristic layers to reduce images, detect patterns and predict;
- Each neuron transform images in a specific way [1];



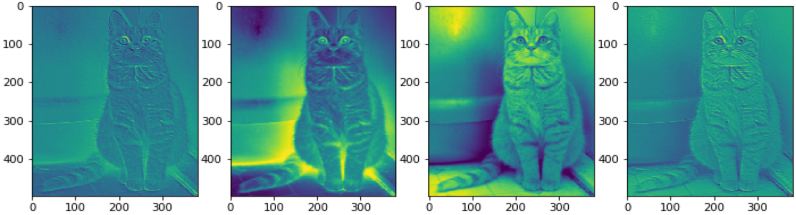
1. Johnson J., n.d.

Convolutional Layer

- Tool to transform images and detect patterns;
- Convolution operator:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

where I and K are respectively Input and Kernel;



Convolutional Layer

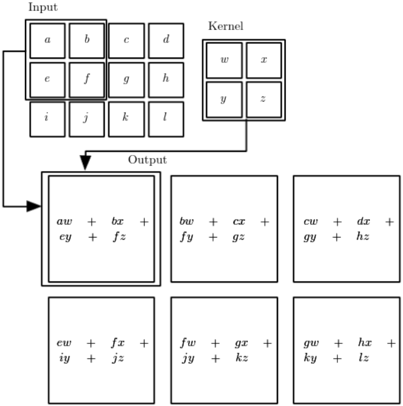
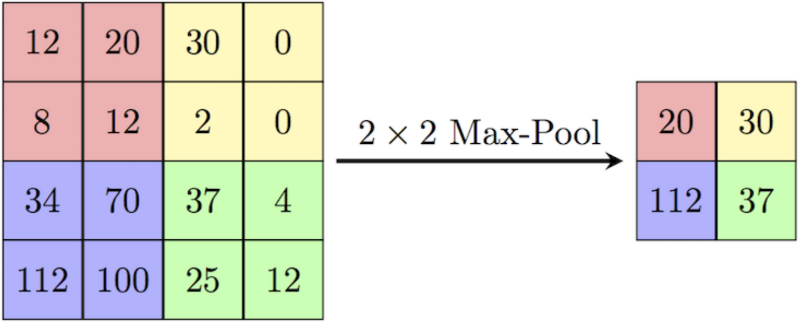


Figure 2: Representation of convolution operation taken from the book of Goodfellow, Bengio and Courville 2016. Applying convolution to a 3x4 image and 2x2 kernel, we get as output a 2x3 transformed image.

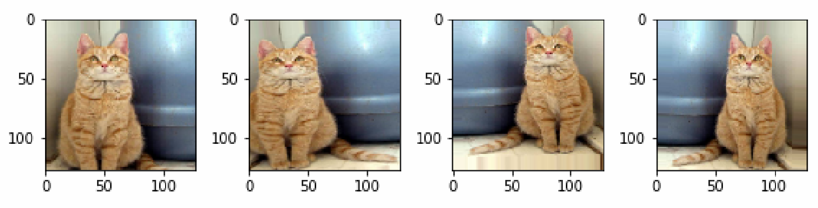
Pooling Layer

Tool to reduce images based on statistics, i.e. average, maximum, minimum etc.
Most common: Max-Pooling.



Further Components

- Dense Layers;
- Non-linear Activation Function[2] (ReLU);
- Data Augmentation.



2. Nair and Hinton 2010 for more details.

CNNs Architecture

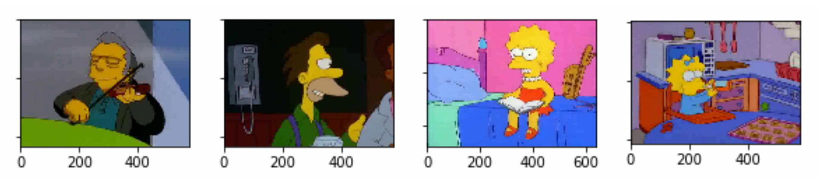
- Many different layers overlapped to make images smaller and more abstract ;
- Example:
  `INPUT`$\rightarrow$`CONV (5x5)`$\rightarrow$`MAX POOL (2X2)`$\rightarrow$`CONV (5x5)` $\rightarrow$ `MAX POOL (2X2)` $\rightarrow$`FLATTEN` $\rightarrow$ `FULLY CONN`$\rightarrow$ `OUTPUT`
- To reduce over-fitting, use at the beginning small and few convolutional kernels, then increase size and quantity.

Practical Example

Dataset Description:

- Simpsons Characters Data Set, provided on Kaggle by Alexattia 2017;
- 42 Simpsons Characters;
- 20933 images in the training set (split in 16746 training and 4187 validation);
- 990 in the test set.

Python libraries: `Tensorflow` and `Keras`



Practical Example - Model Description

- Data augmentation (15 transformations per image);
- 20 epochs;
- Adam Optimization algorithm;
- Cross-Entropy Loss function.

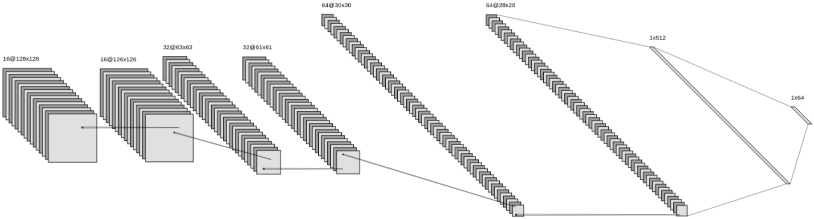
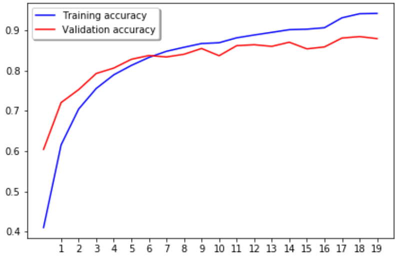
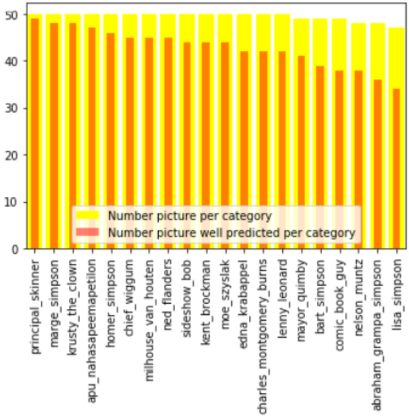


Figure 3: Convolution (3x3)+ ReLU → Max-Pooling → Convolution (3x3)+ReLU → Max-Pooling → Convolution (3x3) +ReLU→ Max-Pooling → Dense (512)+ReLU → Dense (42)+Softmax

Practical Example - Results





Train accuracy	0.9417
Validation accuracy	0.8792
Test accuracy	0.8657

Conclusion

CNNs:

- Powerful tools for images;
- Good results with simple architecture;
- Multi-class problem solvers.

Future works:

- avoid over-fitting with specific layers (i.e. Dropout, Batch Normalization);
- automatic choice of hyperparameters.

Thank you for the attention!

References

Alexattia. 2017. "The simpson Characters Dataset". Accessed 15th June. https://www.kaggle.com/alexattia/the-simpsons-characters-dataset.

Goodfellow, Ian, Yoshua Bengio and Aaron Courville. 2016. *Deep learning*. Chap. 9, 326–366. MIT press.

Johnson J., Karpathy A. n.d. "Convolutional Neural Networks (CNNs / ConvNets)". http://cs231n.github.io/convolutional-networks/.

Nair, Vinod, and Geoffrey E Hinton. 2010. "Rectified linear units improve restricted boltzmann machines". In *Proceedings of the 27th international conference on machine learning (ICML-10)*, 807–814.