UFPR
UNIVERSIDADE FEDERAL DO PARANÁ

Vision, Robotics and Imaging Laboratory

UNICO

BOVIFOCR Project

Biometria Ocular, Vivacidade de Imagens Faciais e Reconhecimento de Texto (OCR) em Documentos Oficiais

Aula - 06/04/2024

Bernardo Biesseck

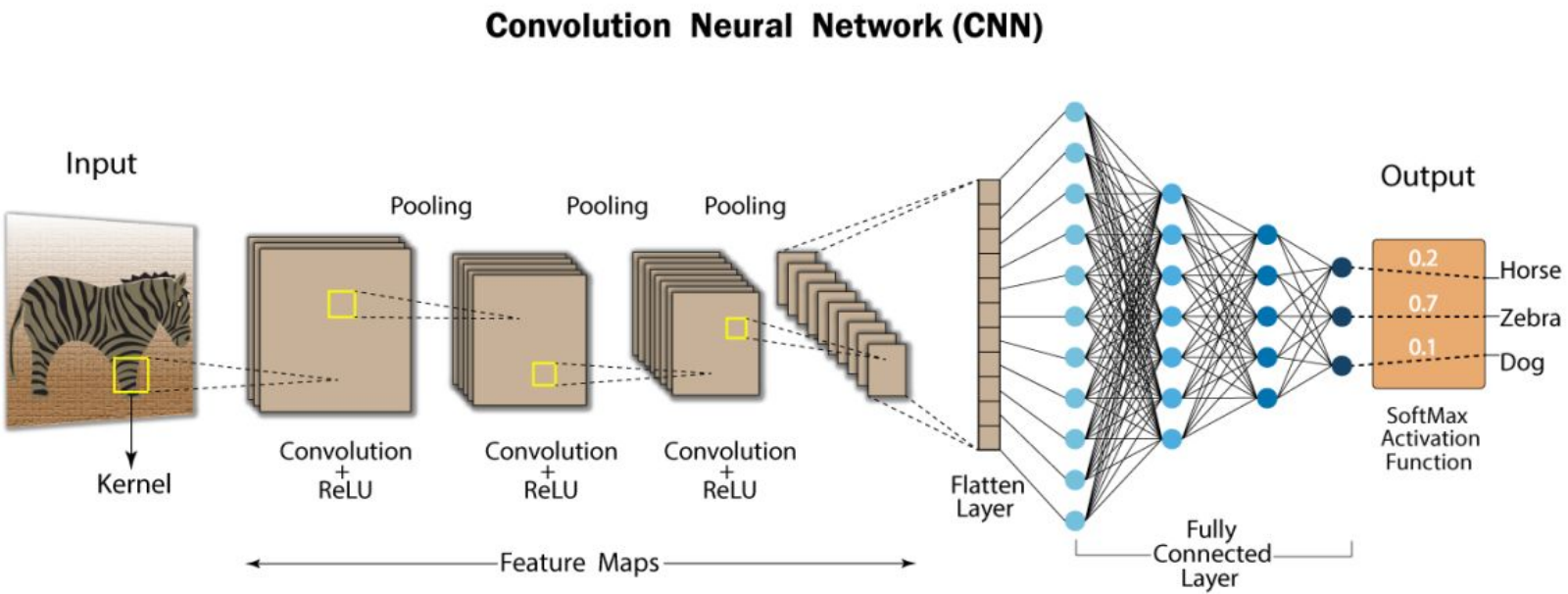
Ph.D. student at the PPGInf (UFPR)

Advisor: David Menotti Gomes

TÓPICOS

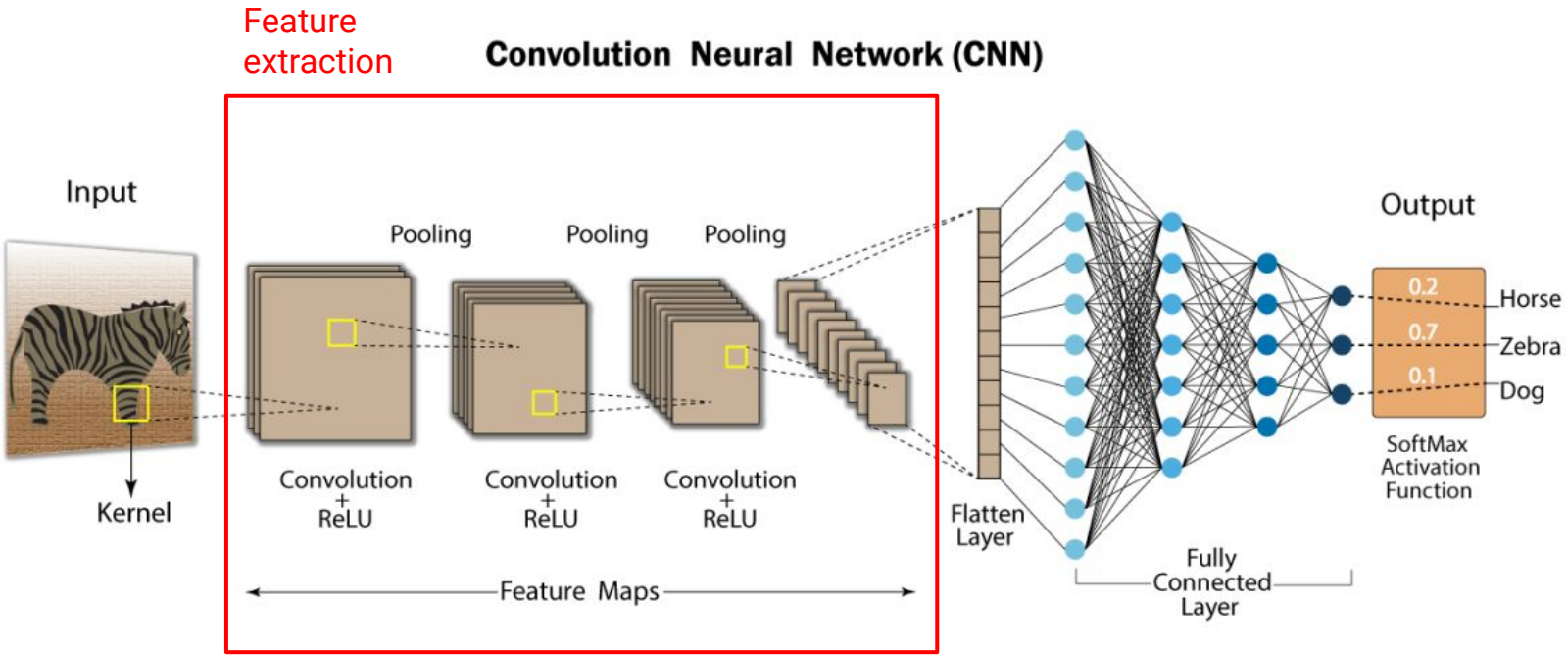
1. Redes Neurais Convolucionais (CNN)
2. Convolução
3. Multilayer perceptron (MLP)
4. Tutorial em Python

Rede Neural Convolucional (CNN)



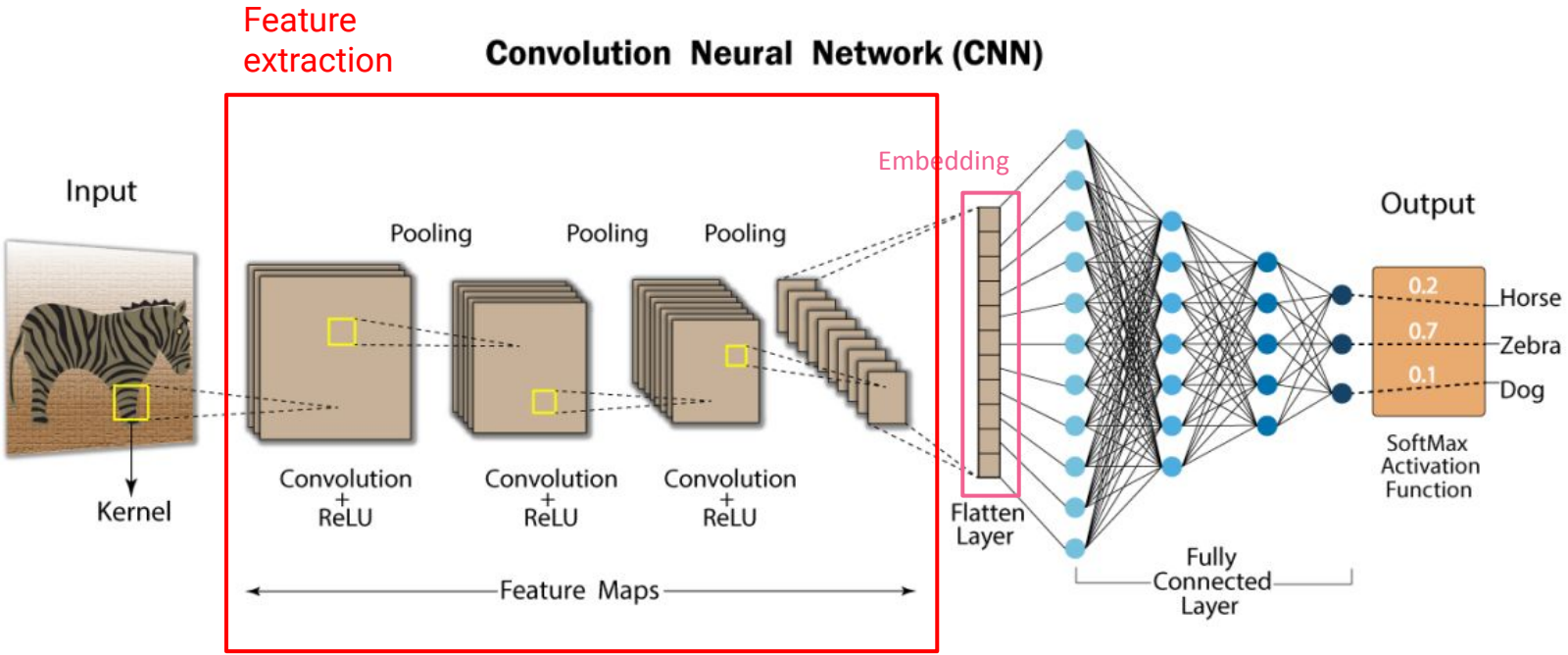
Source: Swapna K E, 2020 - https://developersbreach.com/convolution-neural-network-deep-learning/

Rede Neural Convolucional (CNN)



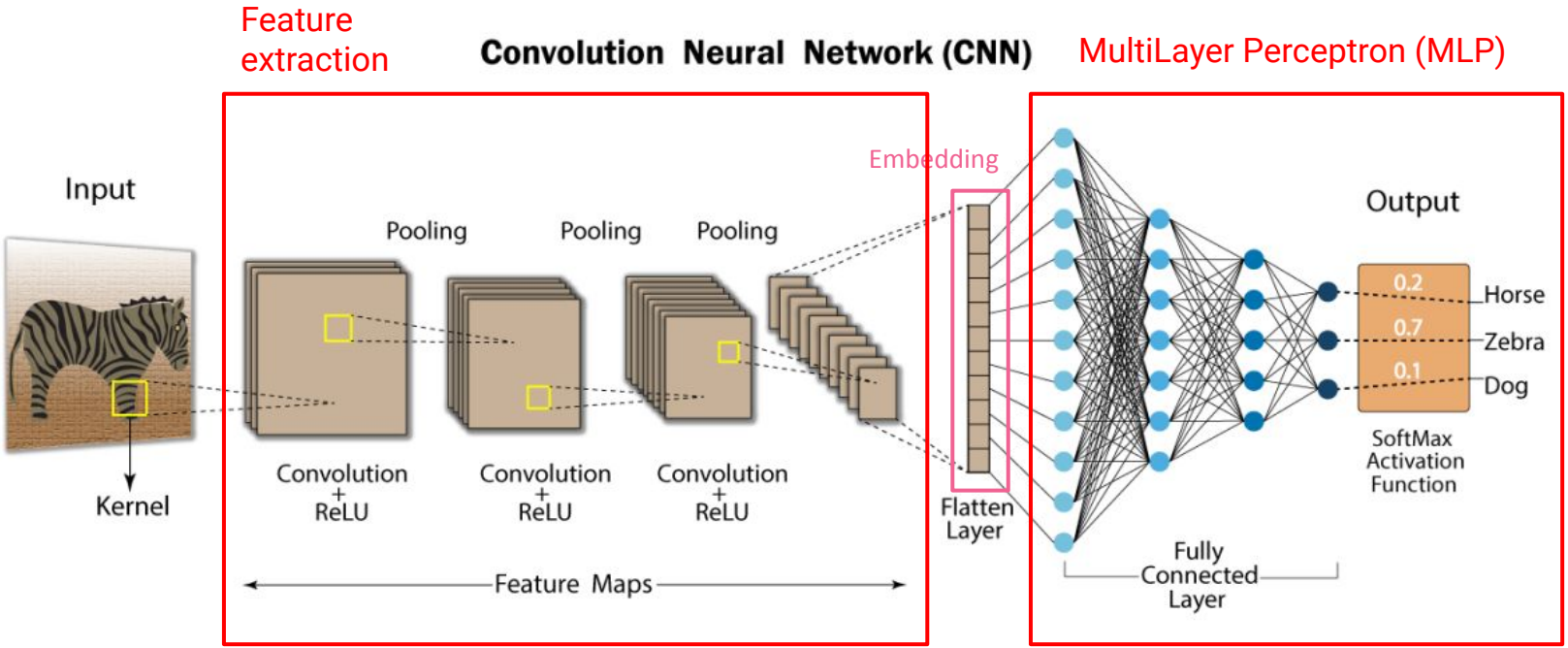
Source: Swapna K E, 2020 - https://developersbreach.com/convolution-neural-network-deep-learning/

Rede Neural Convolucional (CNN)



Source: Swapna K E, 2020 - https://developersbreach.com/convolution-neural-network-deep-learning/

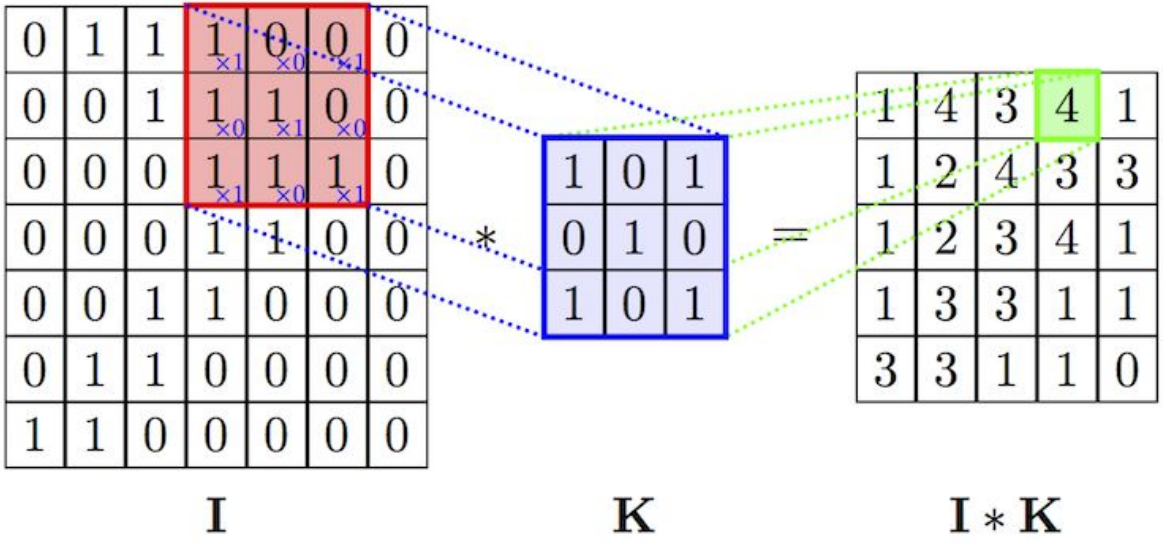
Rede Neural Convolucional (CNN)

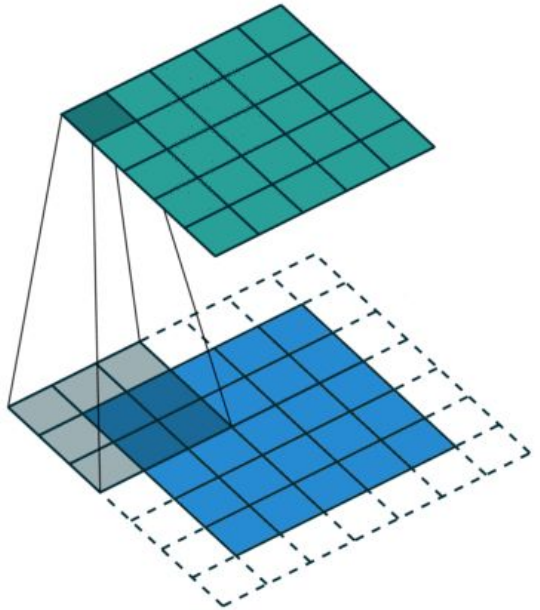


Source: Swapna K E, 2020 - https://developersbreach.com/convolution-neural-network-deep-learning/

Convolução

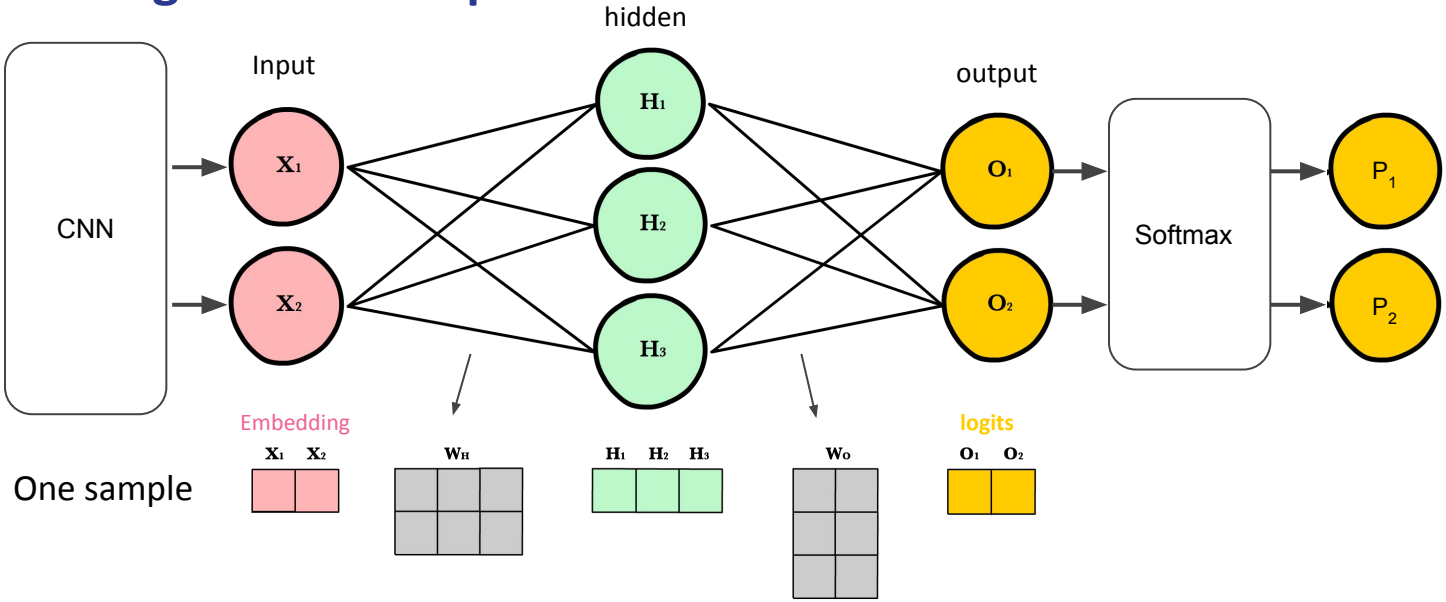
- Convolution





- https://youtu.be/CXOGvCMLrkA?si=a97_j8xxxknIdASU
- https://youtu.be/f0t-OCG79-U?si=WYMbuxs57PZ6R7FB
- https://youtu.be/JboZfxUjLSk?si=HMQ3v22WS8TJLQCL

Background concepts - MLP

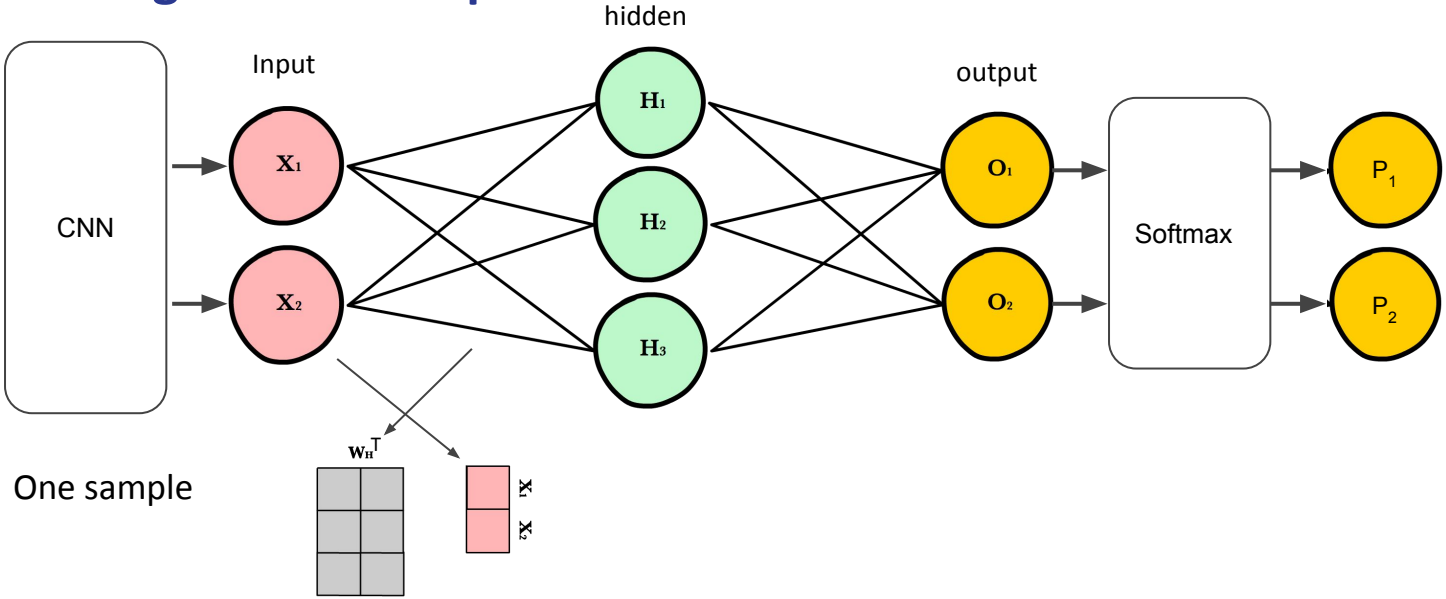


Cross-Entropy/Softmax Loss Function:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{w_{y_i}^T x_i}}{\sum_{j=1}^{C} e^{w_j^T x_i}}$$

Source: Machine Learning Glossary, 2017 - https://ml-cheatsheet.readthedocs.io/en/latest/forwardpropagation.html

Background concepts - MLP



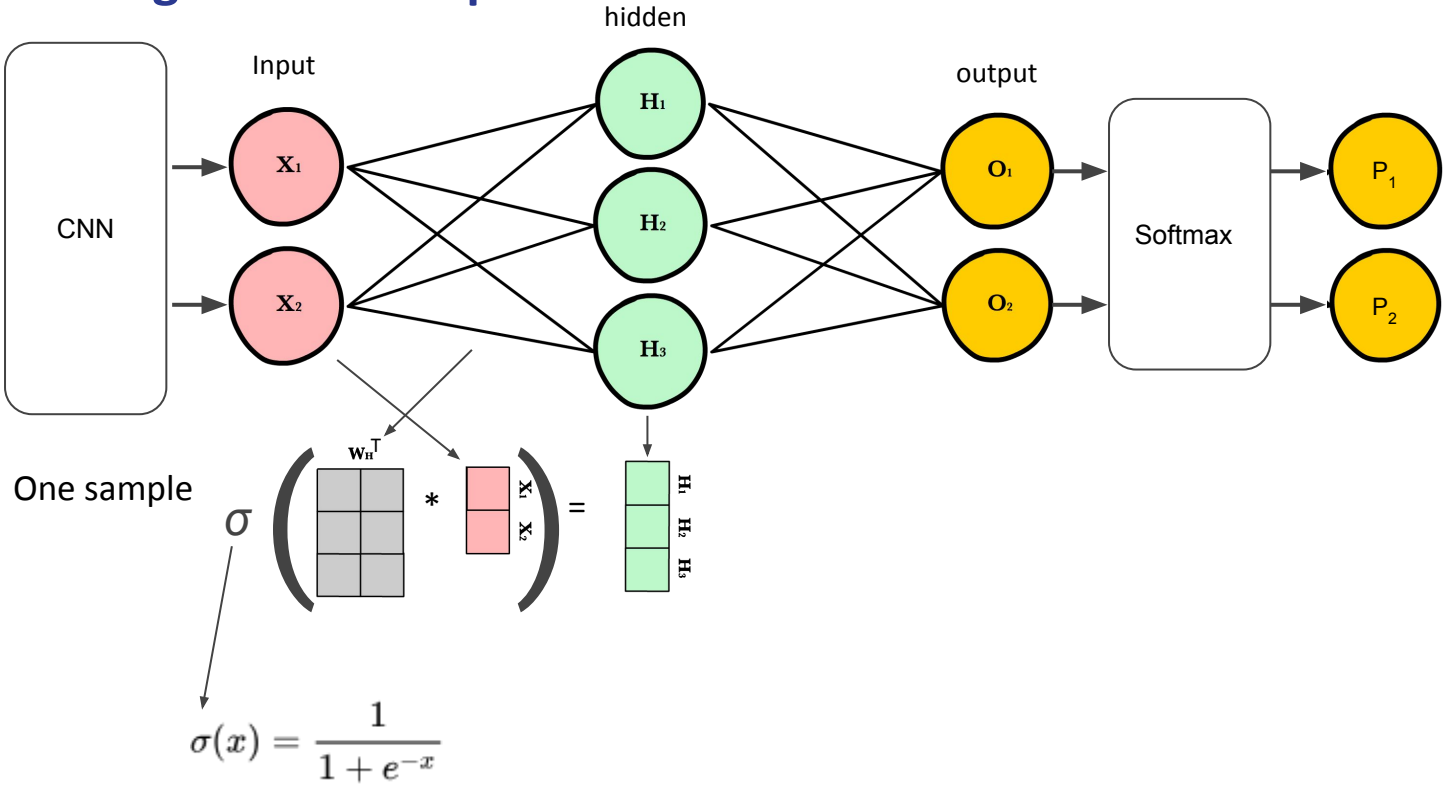
One sample

Cross-Entropy/Softmax Loss Function:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{w_{y_i}^T x_i}}{\sum_{j=1}^{C} e^{w_j^T x_i}}$$

Source: Machine Learning Glossary, 2017 - https://ml-cheatsheet.readthedocs.io/en/latest/forwardpropagation.html

Background concepts - MLP



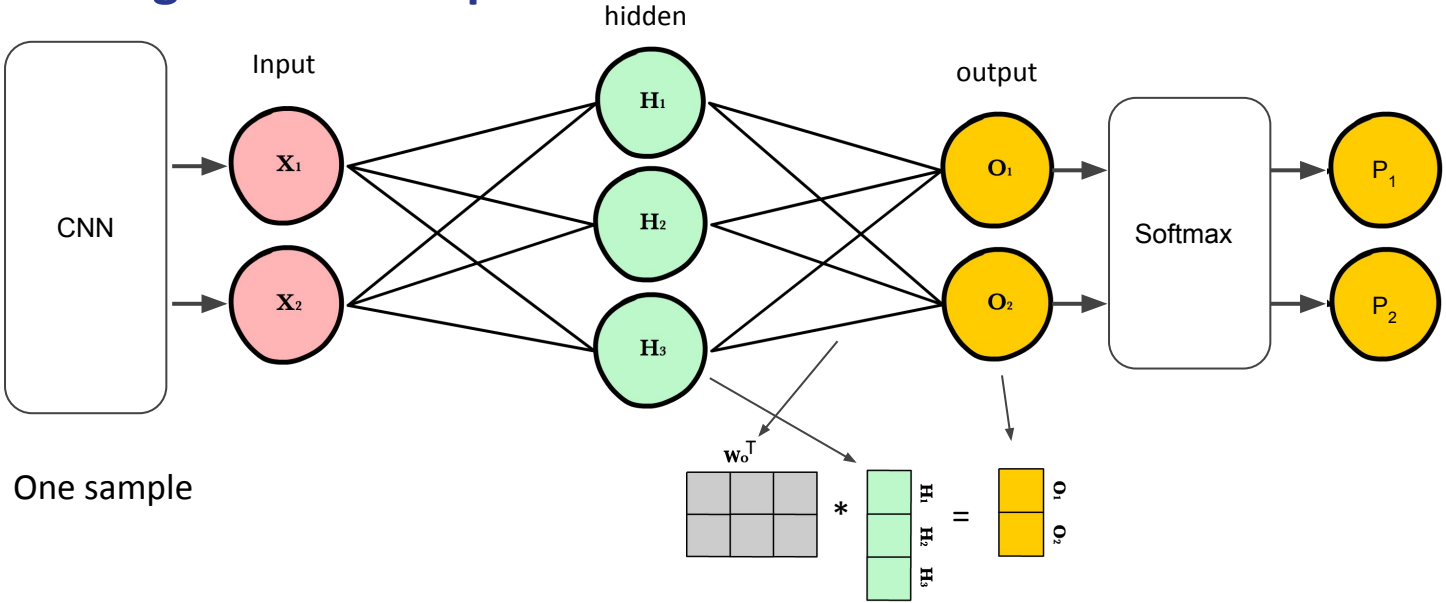
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Cross-Entropy/Softmax Loss Function:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{w_{y_i}^T x_i}}{\sum_{j=1}^{C} e^{w_j^T x_i}}$$

Source: Machine Learning Glossary, 2017 - https://ml-cheatsheet.readthedocs.io/en/latest/forwardpropagation.html

Background concepts - MLP



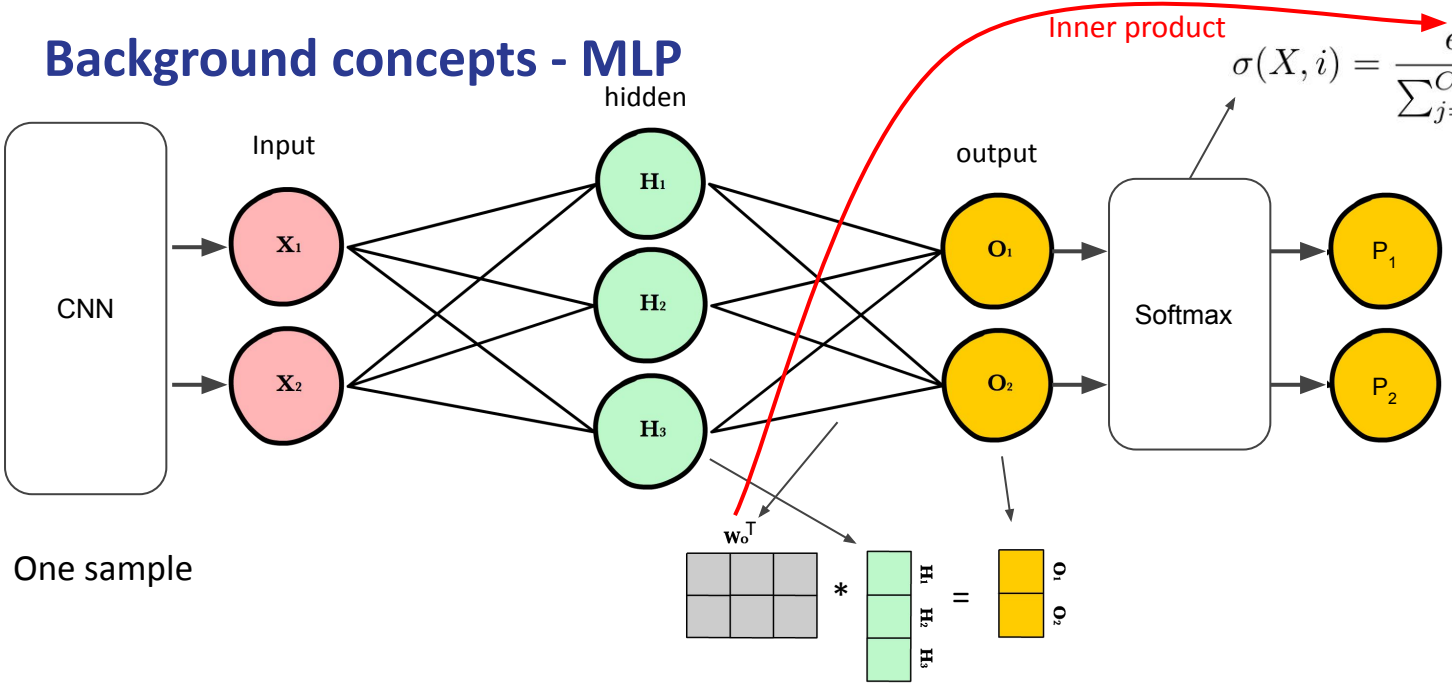
One sample

Cross-Entropy/Softmax Loss Function:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{w_{y_i}^T x_i}}{\sum_{j=1}^{C} e^{w_j^T x_i}}$$

Source: Machine Learning Glossary, 2017 - https://ml-cheatsheet.readthedocs.io/en/latest/forwardpropagation.html

Background concepts - MLP



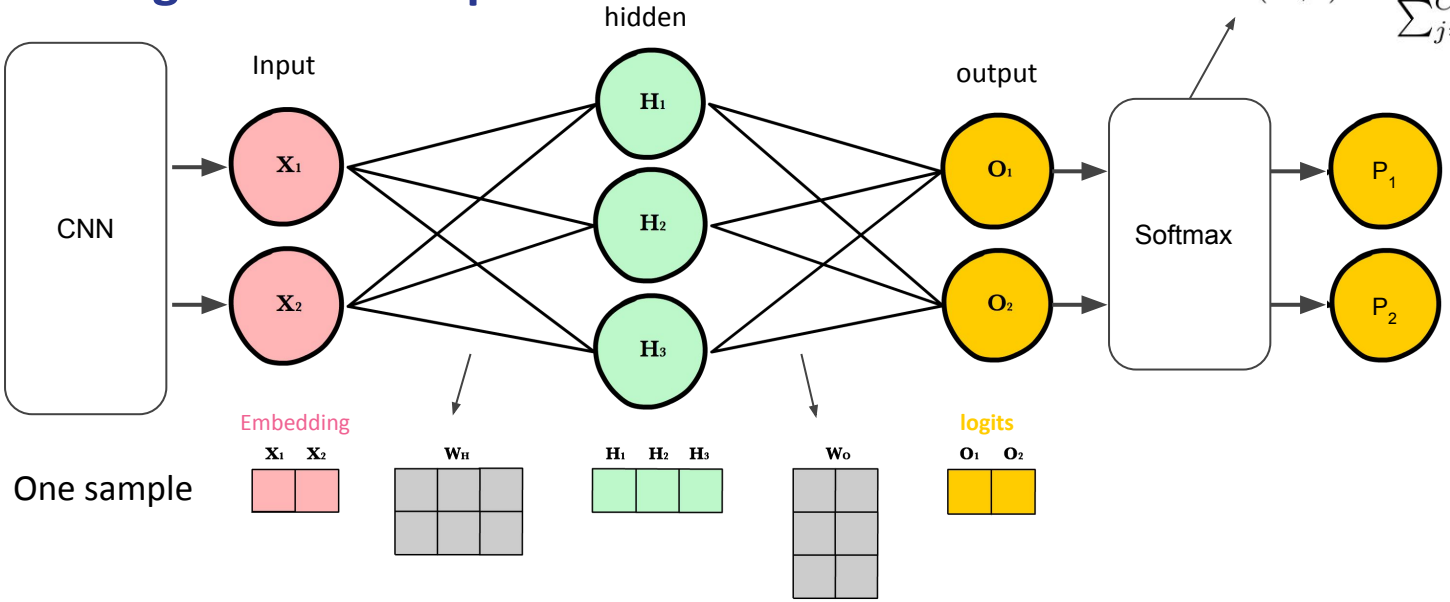
One sample

Cross-Entropy/Softmax Loss Function:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{w_{y_i}^T x_i}}{\sum_{j=1}^{C} e^{w_j^T x_i}}$$

Source: Machine Learning Glossary, 2017 - https://ml-cheatsheet.readthedocs.io/en/latest/forwardpropagation.html

Background concepts - MLP



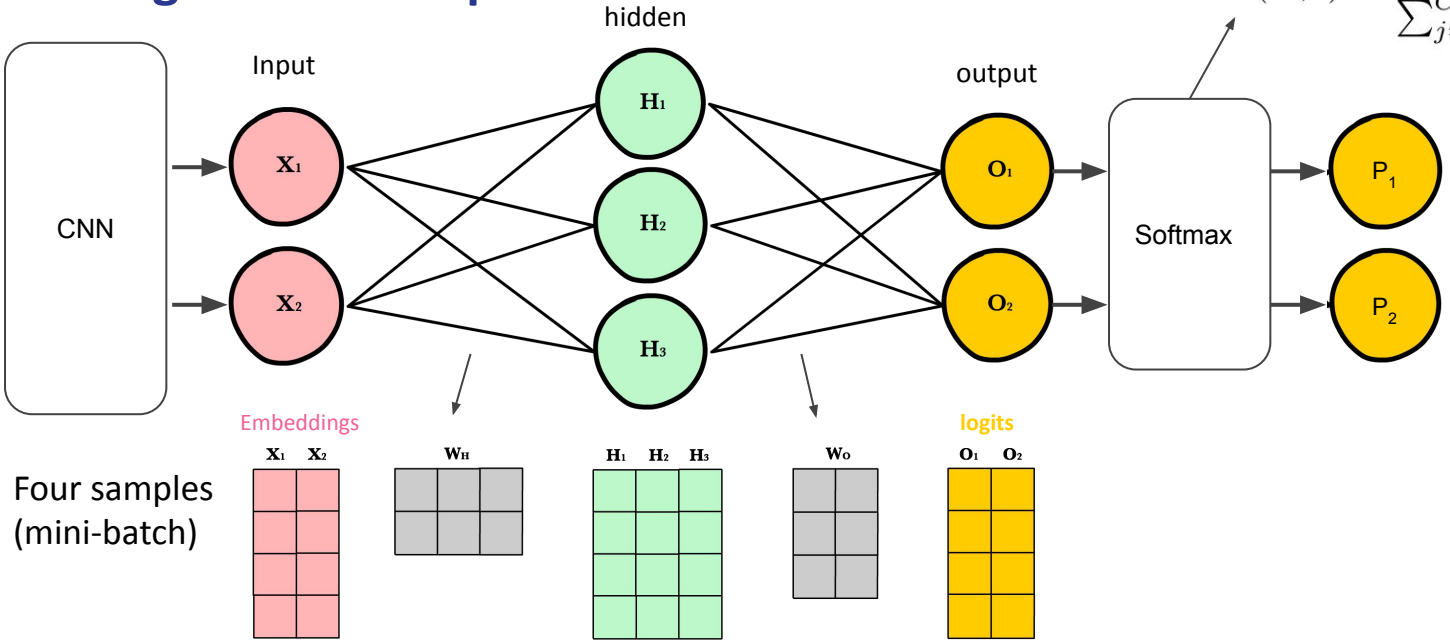
$$\sigma(X,i) = \frac{e^{w_i^T X}}{\sum_{j=1}^{C} e^{w_j^T X}}$$

Cross-Entropy/Softmax Loss Function:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{w_{y_i}^T x_i}}{\sum_{j=1}^{C} e^{w_j^T x_i}}$$

Source: Machine Learning Glossary, 2017 - https://ml-cheatsheet.readthedocs.io/en/latest/forwardpropagation.html

Background concepts - MLP



$$\sigma(X,i) = \frac{e^{w_i^T X}}{\sum_{j=1}^{C} e^{w_j^T X}}$$

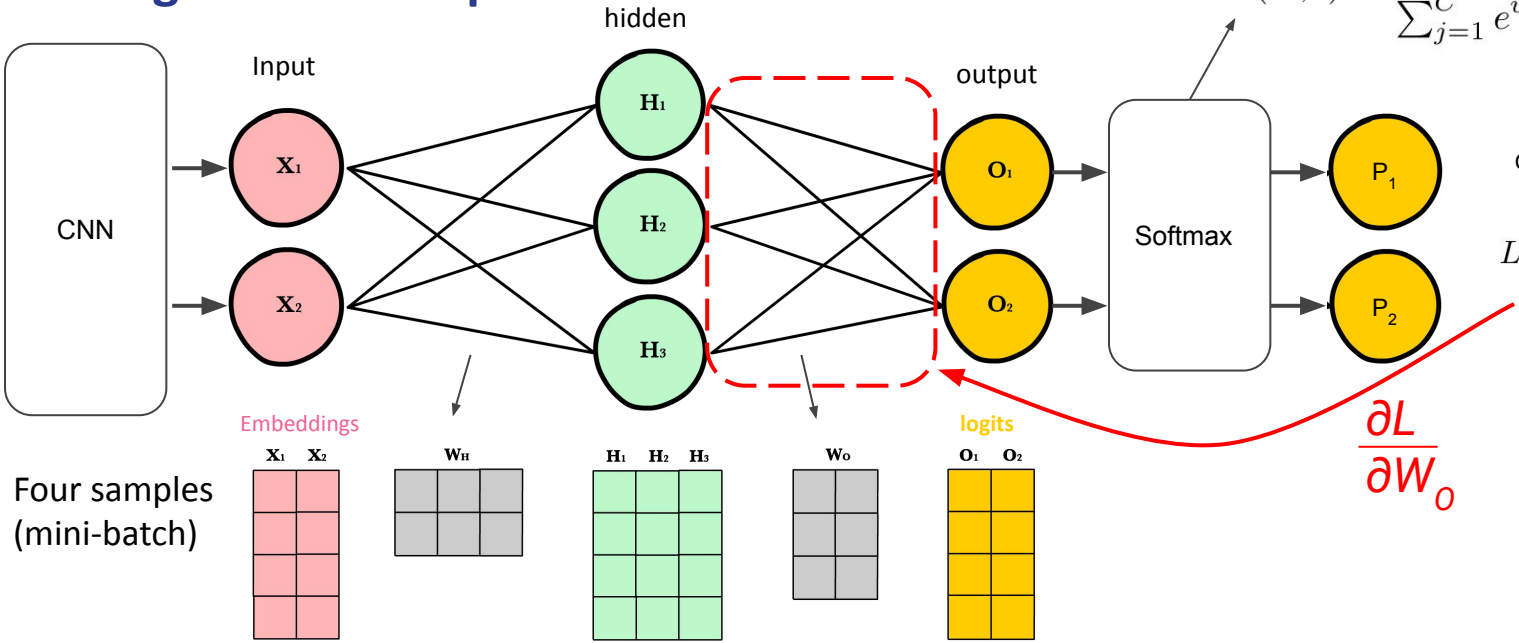
Cross-Entropy/Softmax Loss Function:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{w_{y_i}^T x_i}}{\sum_{j=1}^{C} e^{w_j^T x_i}}$$

Four samples (mini-batch)

Source: Machine Learning Glossary, 2017 - https://ml-cheatsheet.readthedocs.io/en/latest/forwardpropagation.html

Background concepts - MLP



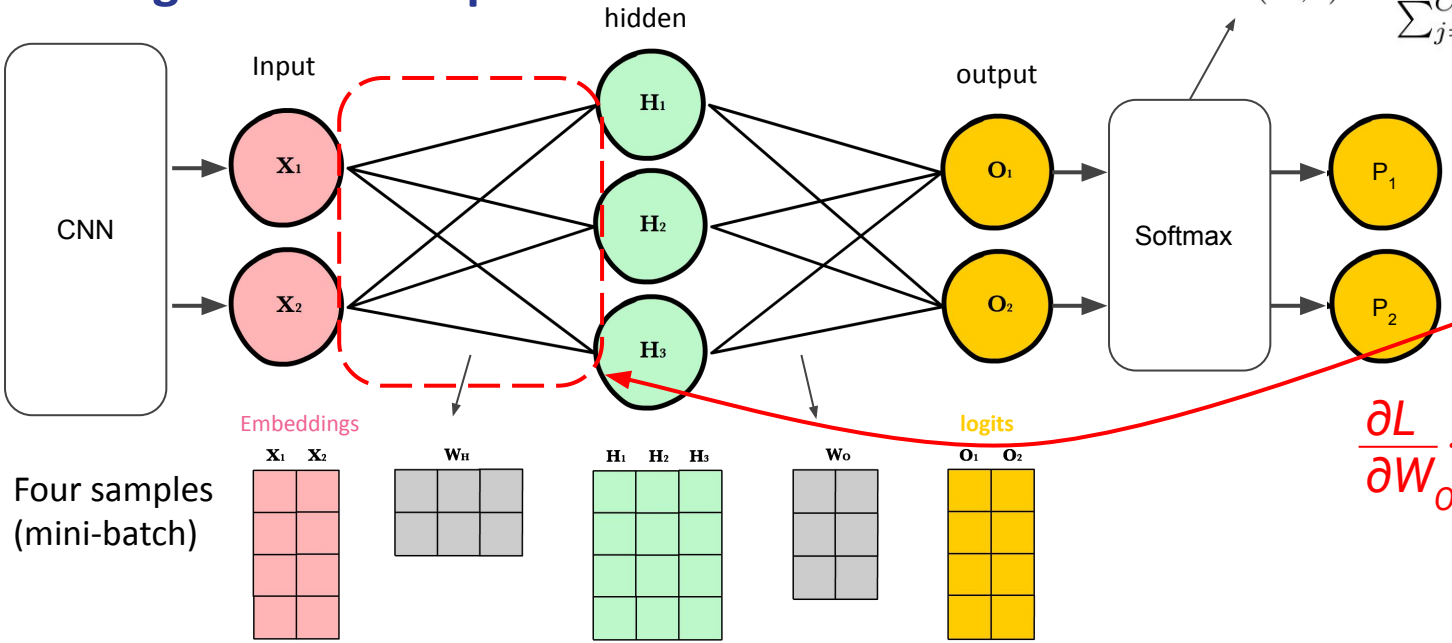
$$\sigma(X,i) = \frac{e^{w_i^T X}}{\sum_{j=1}^{C} e^{w_j^T X}}$$

Cross-Entropy/Softmax Loss Function:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{w_{y_i}^T x_i}}{\sum_{j=1}^{C} e^{w_j^T x_i}}$$

Source: Machine Learning Glossary, 2017 - https://ml-cheatsheet.readthedocs.io/en/latest/forwardpropagation.html

Background concepts - MLP



$$\sigma(X,i) = \frac{e^{w_i^T X}}{\sum_{j=1}^{C} e^{w_j^T X}}$$

Cross-Entropy/Softmax Loss Function:

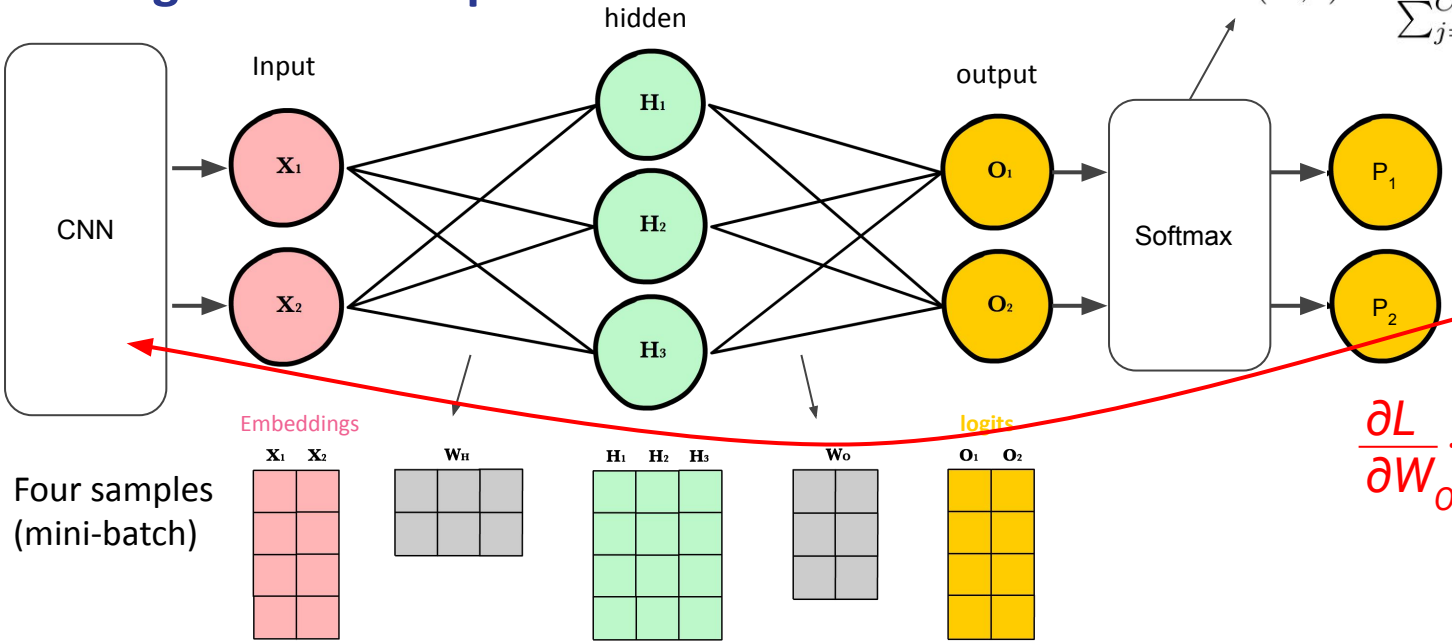
$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{w_{y_i}^T x_i}}{\sum_{j=1}^{C} e^{w_j^T x_i}}$$

$$\frac{\partial L}{\partial W_O} \cdot \frac{\partial L}{\partial W_H}$$

Four samples (mini-batch)

Source: Machine Learning Glossary, 2017 - https://ml-cheatsheet.readthedocs.io/en/latest/forwardpropagation.html

Background concepts - MLP



$$\sigma(X,i) = \frac{e^{w_i^T X}}{\sum_{j=1}^{C} e^{w_j^T X}}$$

Cross-Entropy/Softmax Loss Function:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{w_{y_i}^T x_i}}{\sum_{j=1}^{C} e^{w_j^T x_i}}$$

$$\frac{\partial L}{\partial W_O} \cdot \frac{\partial L}{\partial W_H} \cdot \frac{\partial L}{\partial W_{CNN}}$$

- https://youtu.be/sLsCN9ZL9RI?si=WAI5TuvvptfRjSdZ

Source: Machine Learning Glossary, 2017 - https://ml-cheatsheet.readthedocs.io/en/latest/forwardpropagation.html

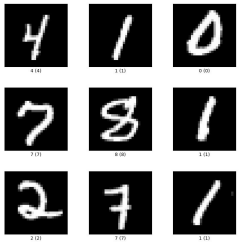
Na prática

Bibliotecas Python:

- Pytorch - https://pytorch.org/
- TensorFlow - https://www.tensorflow.org/
- MXNet - https://mxnet.apache.org/
- Keras - https://keras.io/

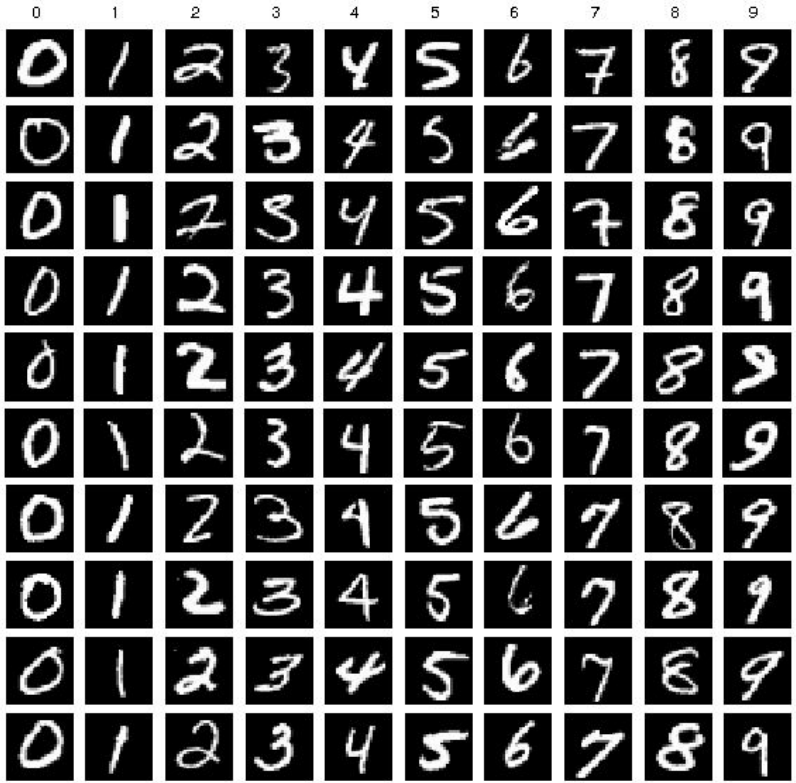
Toy example:

- Dataset MNIST (1994)
  - 28x28 pixels (1x784)
  - 60,000 training images
  - 10,000 testing images

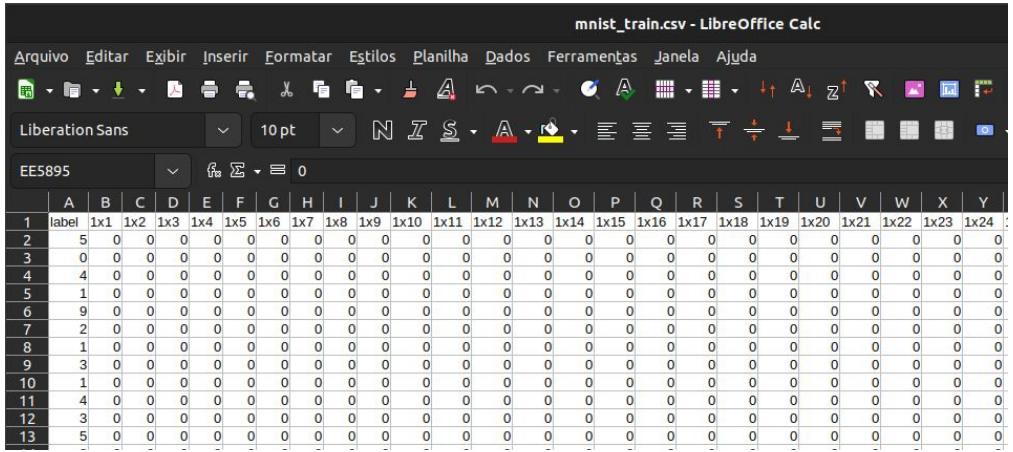


Dataset MNIST

https://www.kaggle.com/datasets/oddrationale/mnist-in-csv



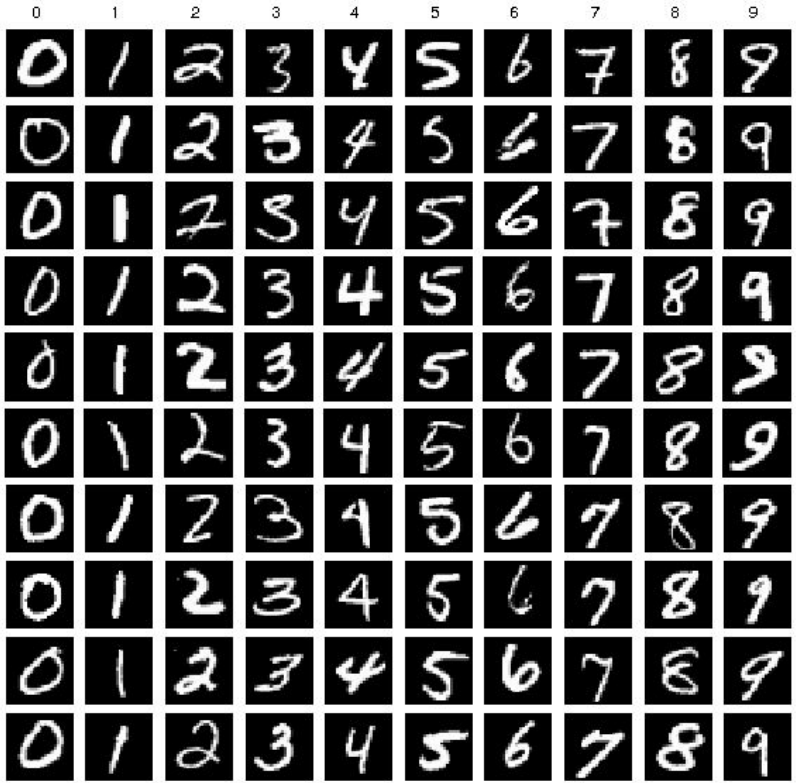
- archive.zip
  - mnist_train.csv
  - mnist_test.csv



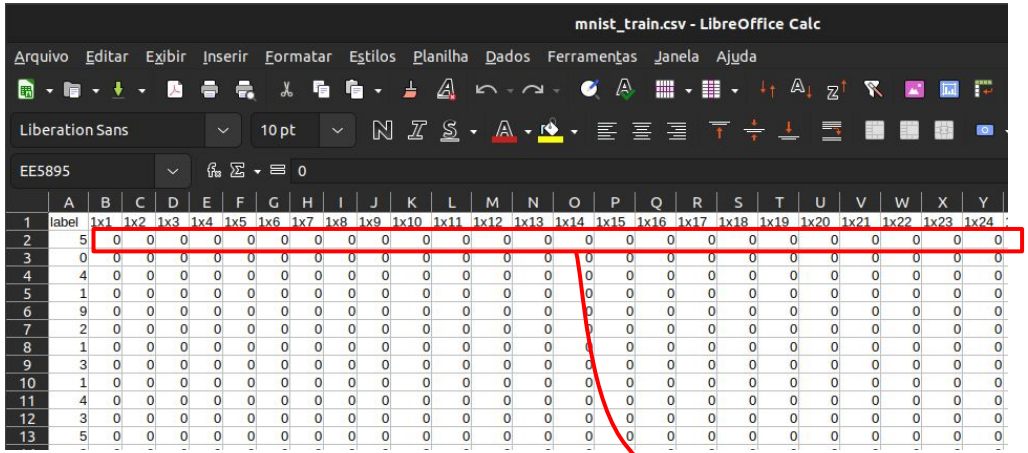

```
import pandas as pd
df = pd.read_csv('../input/mnist_train.csv')
torch_X_train = df.view(-1, 1,28,28).float()
```

Dataset MNIST

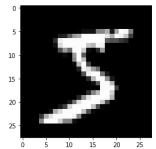
https://www.kaggle.com/datasets/oddrationale/mnist-in-csv



- archive.zip
  - mnist_train.csv
  - mnist_test.csv




```
import pandas as pd
df = pd.read_csv('../input/mnist_train.csv')
torch_X_train = df.view(-1, 1,28,28).float()
```

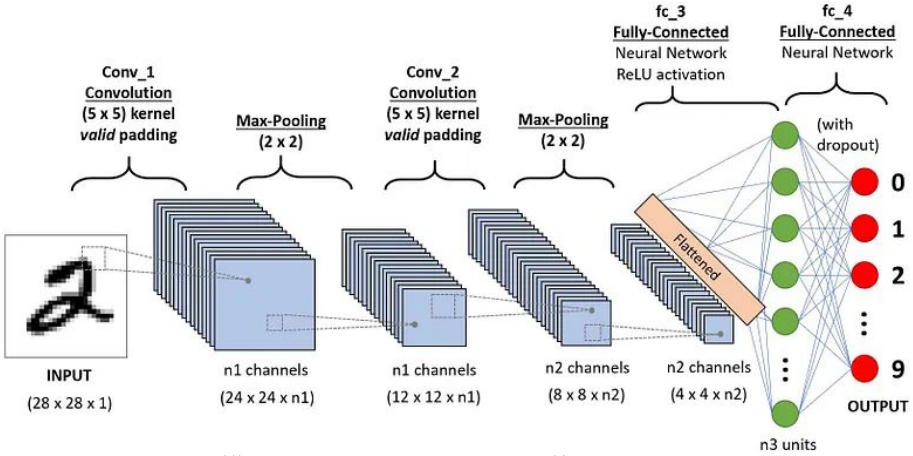


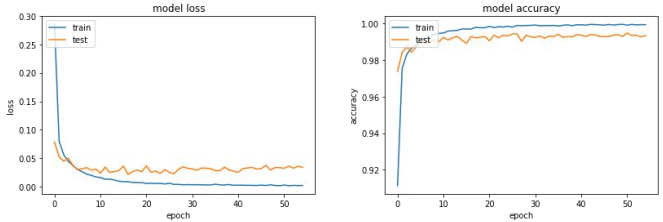
Dataset MNIST

Pytorch tutorial

- https://www.kaggle.com/code/sdelecourt/cnn-with-pytorch-for-mnist

```
# Install dependencies
pip3 install numpy pandas scikit-learn torch
```






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

class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Conv2d(1, 32, kernel_size=5)
        self.conv2 = nn.Conv2d(32, 32, kernel_size=5)
        self.conv3 = nn.Conv2d(32,64, kernel_size=5)
        self.fc1 = nn.Linear(3*3*64, 256)
        self.fc2 = nn.Linear(256, 10)

    def forward(self, x):
        x = F.relu(self.conv1(x))
        x = F.relu(F.max_pool2d(self.conv2(x), 2))
        x = F.dropout(x, p=0.5, training=self.training)
        x = F.relu(F.max_pool2d(self.conv3(x),2))
        x = F.dropout(x, p=0.5, training=self.training)
        x = x.view(-1,3*3*64 )
        x = F.relu(self.fc1(x))
        x = F.dropout(x, training=self.training)
        x = self.fc2(x)
        return F.log_softmax(x, dim=1)
```

Dataset MNIST

Keras tutorial

- https://www.kaggle.com/code/tobikaggle/keras-mnist-cnn-learning-curve

  # Install dependencies
  pip3 install numpy pandas scikit-learn keras matplotlib