UNIVERSITÄT LEIPZIG

ICDS - Classification Problem

Detecting Pneumonia in X-Ray Images

Leipzig, 02.06.2023

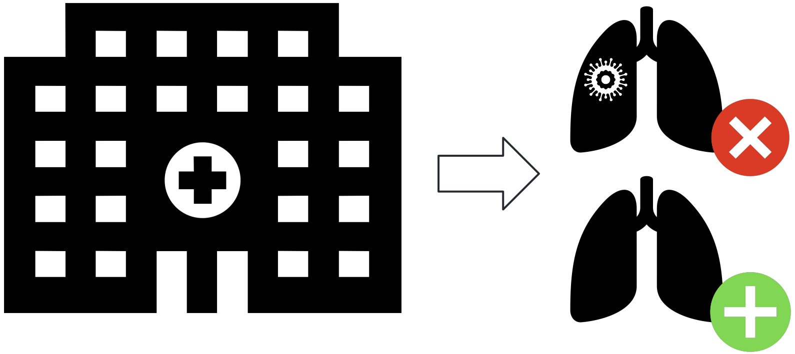
Group 1: Baibekova Asel, Bertels Finn, Hacker Leonard, Perkovic Nikola, Schaefer Erik

The problem and our goal.

Problem | Basics | Technology | Organisation

The problem and our goal.

- Medical classification of X-Ray images of lungs.
  ⟹ “Healthy” or pneumonia



Manual classification:

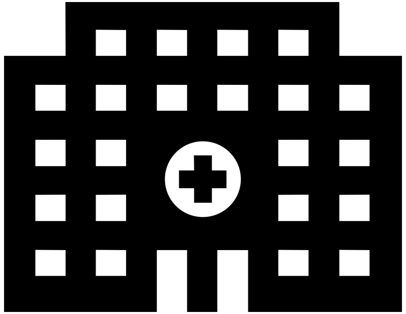
- Time consuming
- Not cost-efficient
- Error-prone

Model classification:

- Most important metric is recall
  - TP / (TP + FN)
- False negative is worse than false positive

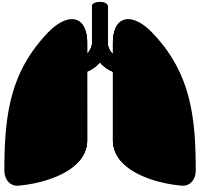
Dataset

- Production of the image classification set:

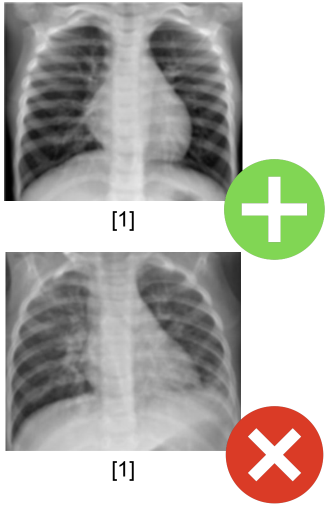


Guangzhou Women and Children's Medical Center

X-Ray images of children between 1-5 year



Images of low quality removed



Classification by two expert physicians

5.200 images

Control by a third expert physician

[1] Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning, D. S. Kermany et al.

What is classification?

Problem | Basics | Technology | Organisation

What is classification?

- **Binary Classification**
  - only two classes
- Multi-Class Classification
  - more than two classes
- Multi-Label Classification
  - one data point can have multiple labels
- Imbalanced Classification
  - imbalanced class distribution in training data

[1]

[2]

What is classification?

- Binary Classification
  - only two classes

- **Multi-Class Classification**
  - more than two classes

- Multi-Label Classification
  - one data point can have multiple labels

- Imbalanced Classification
  - imbalanced class distribution in training data

[1]

[2]

[3]

What is classification?

- Binary Classification
  - only two classes

- Multi-Class Classification
  - more than two classes

- **Multi-Label Classification**
  - one data point can have multiple labels

- Imbalanced Classification
  - imbalanced class distribution in training data

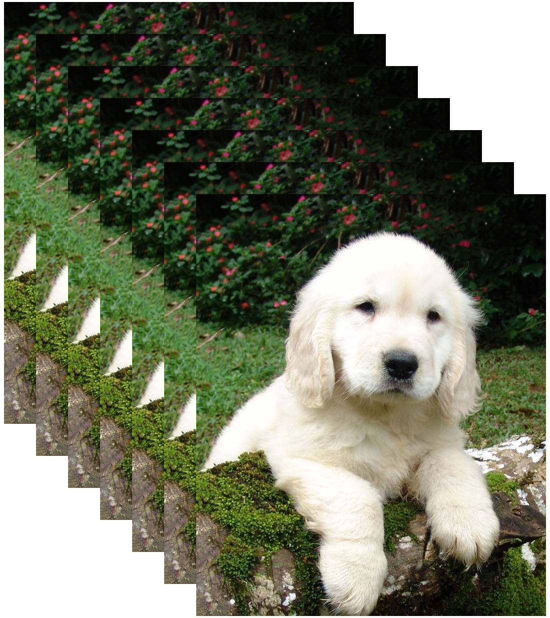
[1]

[1] Cat & Dog: By Prskavka at Slovak Wikipedia - Transferred from sk.wikipedia to Commons., Public Domain

What is classification?

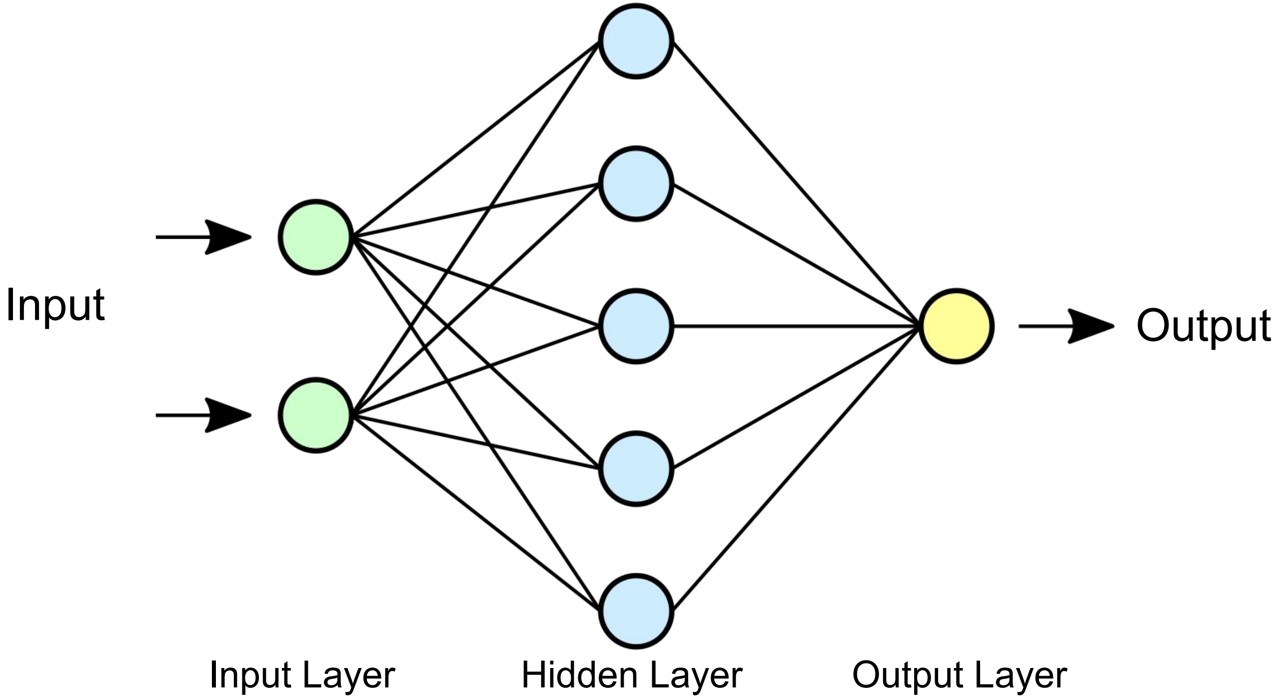
- Binary Classification
  - only two classes
- Multi-Class Classification
  - more than two classes
- Multi-Label Classification
  - one data point can have multiple labels
- **Imbalanced Classification**
  - imbalanced class distribution in training data

[1]



[2]

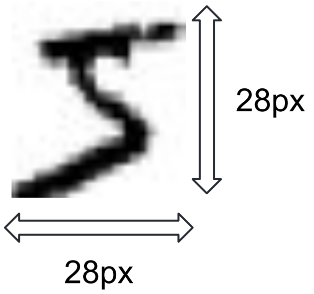
Classic Neural Network

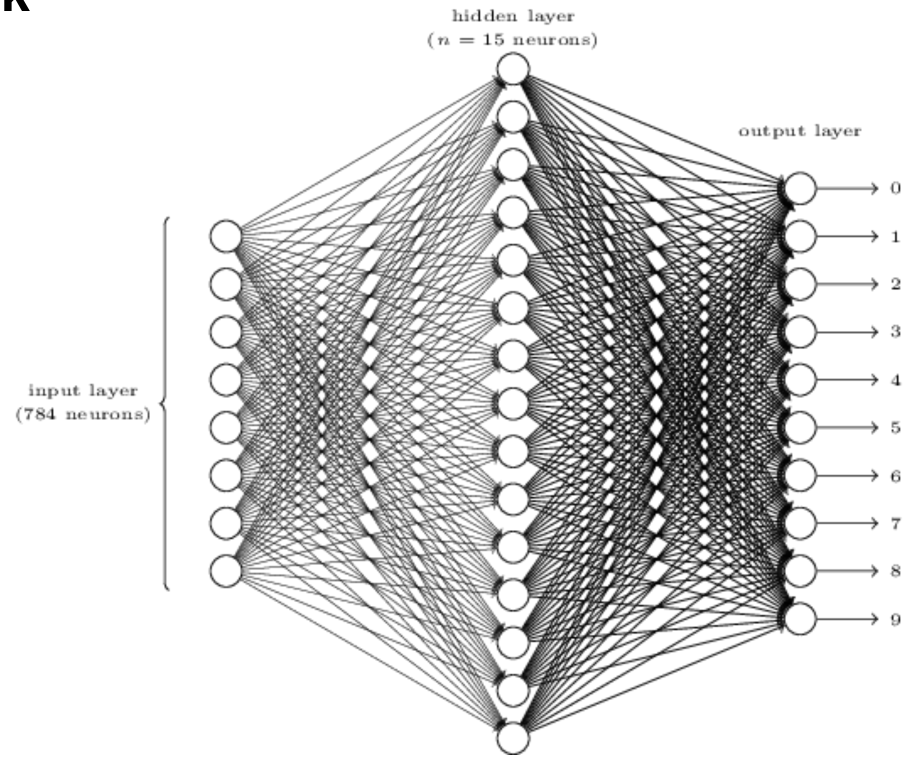


[1]

[1] Von Dake, Mysid - Vectorized by Mysid in CorelDraw on an image by Dake., CC BY 1.0

Classic Neural Network





[1]

[1] Michael Nielsen - Using neural nets to recognize handwritten digits

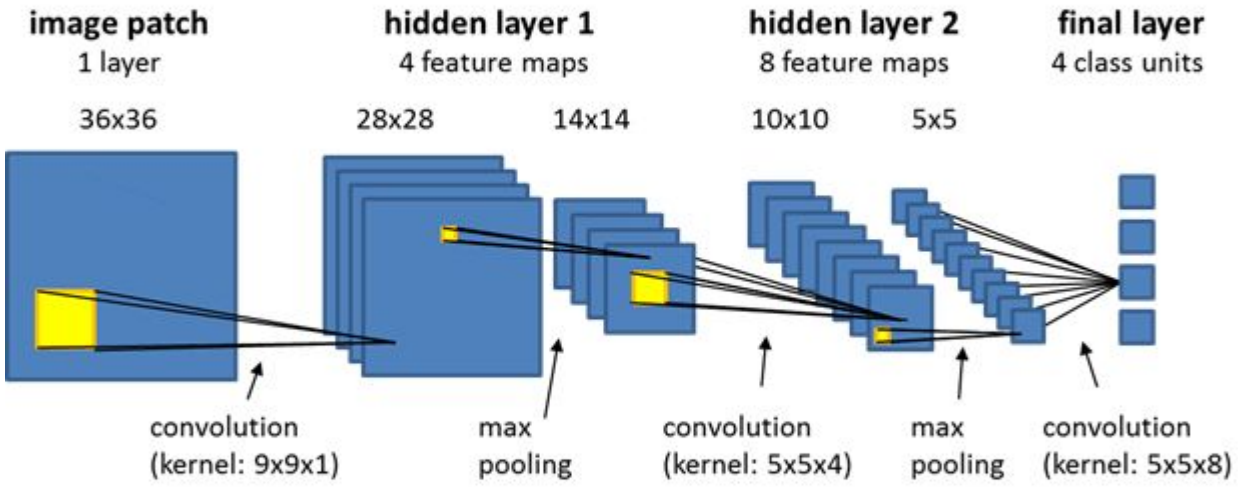
Filter / Kernel

[1]

1	1	1
0	0	0
-1	-1	-1

1	0	-1
1	0	-1
1	0	-1

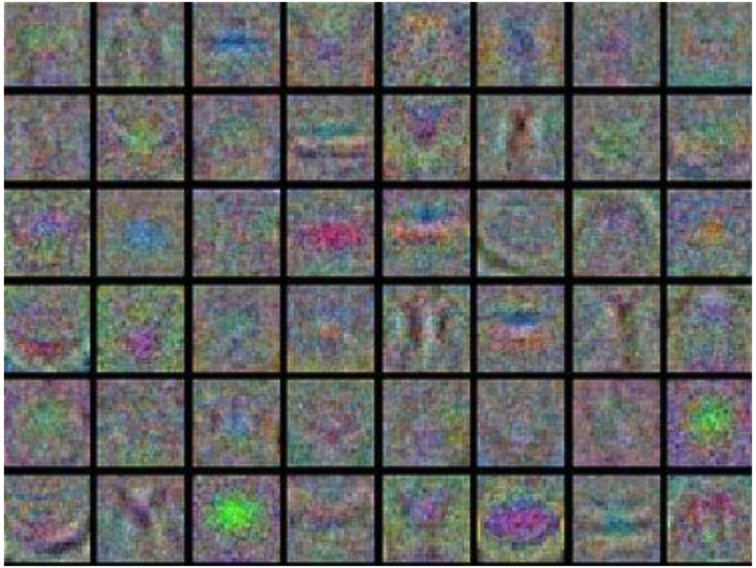
Convolutional Neural Network (CNN)



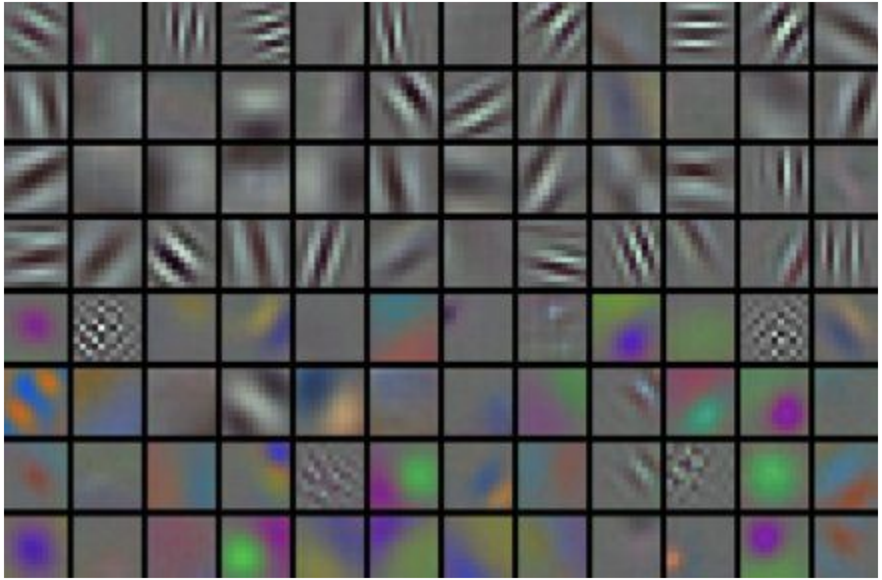
[1]

[1] 2019, Trimble Inc. - Convolutional Neural Network Algorithms

Kernel



[1]



[1]

[1] cs231n - First-layer Visualizations

What technologies do we use?

Problem › Basics › **Technology** › Organisation

Libraries

- NumPy
    - powerful n-dimensional arrays
    - interoperable
    - simply fast
- OpenCV
    - widely used
    - highly optimized
- PyTorch
    - a replacement for NumPy to make use of the power of GPUs
    - flexible
    - extremely fast
- Pre trained model
    - ResNet

How we work as an organisation.

Problem → Basics → Technology → Organisation

How we work together

Importance of collaboration and teamwork to achieve success

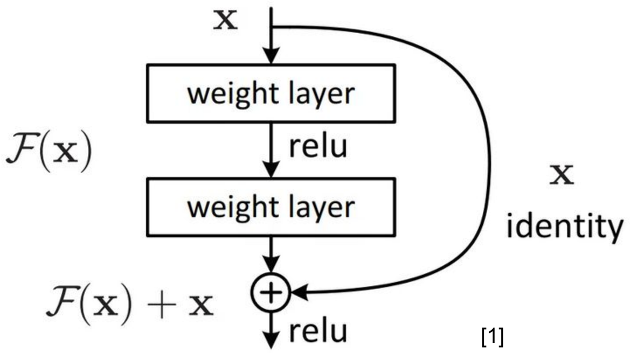
- Meeting cadence:
  - two times a week
  - hybrid work
- In Progress:
  - Introduction of project management
    - Milestone plan
    - Project board on GitHub
    - Project documentation on GitHub

Presentation → Data Preparation → First Implementation → Integration of PyTorch → Integration of ResNet → Final Presentation

Thank you

pre-trained models

- ResNet:
    - skip connections → allow the network to effectively learn from the difference between the current layer's representation and the desired output
    - advantages:
        - better accuracy
        - faster convergence
        - better generalization
    - disadvantages:
        - Overfitting with smaller data sets
        - Higher complexity (bigger requirements in computing and storage)



[1]

[1] Residual Block | Source: Deep Residual Learning for Image Recognition

Agenda

- The Problem (Nikola, Finn) - 5 min
  - Dataset
  - Goals
    - High sensitivity
- Basics (Leonard, Erik) - 5 min
  - What is classification?
  - Convolutional Neural Networks
- Technologies - 2,5 min
  - Libraries
    - OpenCV (Leonard)
    - PyTorch (Asel)
    - numpy (Asel)
  - pre-trained models (Finn, Erik)
    - resnet
    - VGG net
- // How do we work? (Nikola) 2,5 min

Convolutional Neural Network (CNN)

Convolution:

- detect local patterns and characteristics in the input data with filter

Pooling:

- reduce data dimensionality and improve computational efficiency

Flattening:

- converting all the resultant 2-dimensional arrays into a single long continuous linear vector

Full Connecntion:

- each node in the output layer connects directly to a node in the previous layer

Convolutional Neural Network (CNN)

- Convolutional Layer
  - kernel (filter) weight certain parts of images into next layer
  - kernels are changed in training
- Pooling layer
  - Pictures are downsampled
  - taking the most significant pixel out of an area

pre-trained models

	MobileNet	VGGNet*
Intended Use	Detection, Classification, Recognition	Detection and framing of object types
Application	Lightweight app on mobile device	
Disadvantages	Lower accuracy than ResNet*	Heavy in resources; Overfitting with small data set

*Visual Geometry Group Network
*in benchmarking datasets

NumPy ndarray: a multi-dimensional array object

- The NumPy ndarray object is a fast and flexible container for large data sets in Python.
- NumPy arrays are a bit like Python lists, but are still a very different beast at the same time.
- Arrays enable you to store multiple items of the same data type. It is the facilities around the array object that makes NumPy so convenient for performing math and data manipulations.

Difference between lists and ndarrays

- The key difference between an array and a list is that arrays are designed to handle vectorised operations while a python lists are not.

  That means, if you apply a function, it is performed on every item in the array, rather than on the whole array object

Measuring the Accuracy of a Classification Model with NumPy

Let say, cat_dog_goose_other function tries to classify whether a picture is of a cat (class 0), a dog (class 1), a goose (class 2), or something else (class 3). We want to measure the *accuracy* of our classifier. That is, we want to feed it a series of images whose contents are known and tally the number of times the model's prediction matches the true content of an image. The accuracy is the fraction of images that the model classifies correctly.

For each image we feed the cat_dog_goose_other model, it will produce four **scores** - one score for each class. The model was designed such that the class with the highest score corresponds to its prediction. There are no constraints on the values the scores can take. For example, if the model processes one image it will return a shape- (1,4) score-array:

```
>>> scores = cat_dog_goose_other(image)
# processing one image produces a 1x4 array of classification scores
>>> scores
array([[-10, 33, 580, 100]])
```

Measuring the Accuracy of a Classification Model with NumPy

```
>>> scores = cat_dog_goose_other(image)
# processing one image produces a 1x4 array of classification scores
>>> scores
array([[-10, 33, 580, 100]])
```

This model has predicted that this is a picture of a goose, since the score associate with class 2 is the largest value. In general, if we pass cat_dog_goose_other an array of N images, it will return a shape-(N,4) array of classification scores - each of the N images has 4 scores associated with it.

Measuring the Accuracy of a Classification Model with NumPy

Because we are measuring our model's accuracy, we have curated a set of images whose contents are known. That is, we have a true **label** for each image, which is encoded as a class-ID. For example, a picture of a cat would have the label 0 associated with it, a picture of a dog would have the label 1 and so on. Thus, a stack of N images would have associated with it a shape-N array of integer labels, each label is within (0,4).

Suppose we have passed our model five images, and it produced the following scores:

Measuring the Accuracy of a Classification Model with NumPy

Suppose we have passed our model five images, and it produced the following scores:

```
# Classification scores produced by `cat_dog_goose_other`
# on five images. A shape-(5, 4) array.
>>> import numpy as np
>>> scores = np.array([[ 30,   1,  10,  80],  # prediction: other
...                    [-10,  20,   0,  -5],  # prediction: dog
...                    [ 27,  50,   9,  30],  # prediction: dog
...                    [ -1,   0,  84,   3],  # prediction: goose
...                    [  5,   2,  10,   0]]) # prediction: goose
```

And suppose that the true labels for these five images are:

```
# truth: cat, dog, dog, goose, other
>>> labels = np.array([0, 1, 1, 2, 3])
```

Measuring the Accuracy of a Classification Model with NumPy

Our model classified three out of five images correctly; thus, our accuracy function should return 0.6:

```
>>> classification_accuracy(scores, labels)
0.6
```

Unvectorized Solution

```
pred_labels = []  # Will store the N predicted class-IDs
for row in classification_scores:
    # store the index associated with the highest score for each datum
    pred_labels.append(np.argmax(row))
```

```
num_correct = 0
for i in range(len(pred_labels)):
    if pred_labels[i] == true_labels[i]:
        num_correct += 1
```

```
num_correct = sum(p == t for p, t in zip(pred_labels, true_labels))
```

```
def unvectorized_accuracy(classification_scores, true_labels):
```

```
pred_labels = []  # Will store the N predicted class-IDs
for row in classification_scores:
    pred_labels.append(np.argmax(row))

num_correct = 0
for i in range(len(pred_labels)):
    if pred_labels[i] == true_labels[i]:
        num_correct += 1
return num_correct / len(true_labels)
```

PyTorch

- PyTorch is a Python-based library designed to provide flexibility as a deep learning development platform
- The PyTorch workflow is as close as possible to the Python scientific computing library: NumPy
- PyTorch and TensorFlow are similar in that the core components of both are tensors and graphs

Tensors are a core PyTorch data type, similar to a multidimensional array, used to store and manipulate the inputs and outputs of a model, as well as the model’s parameters. Tensors are similar to NumPy’s ndarrays, except that tensors can run on GPUs to accelerate computing.