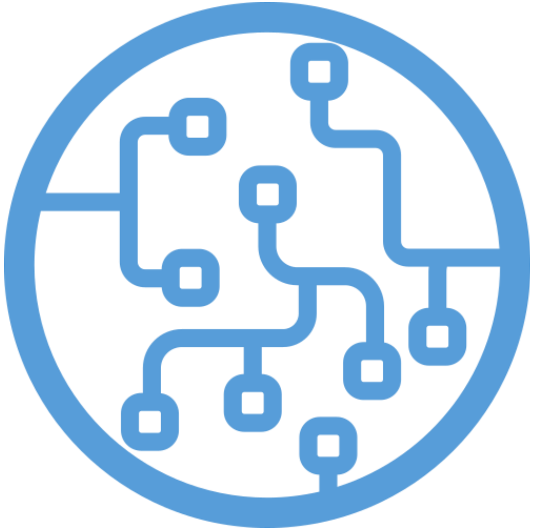
Facial Emotion Recognition

Jonas Eckhoff – Timo Greve – Max Lewerenz – Giulia Satiko Maesaka – John-Robert Wrage

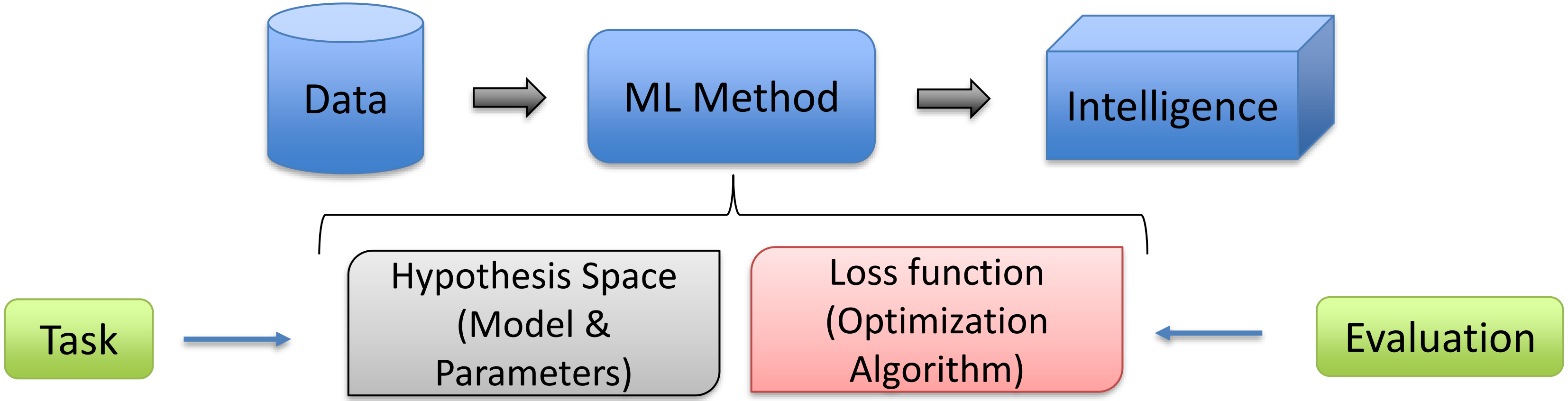
Overview

1. Intro
2. Project
3. Product
4. Algorithms
5. Presentation
6. Conclusion



About Machine Learning

- Find algorithms/methods to perform certain tasks independently;
- Used for tasks where any rule-based approach is unfeasable.



Let’s get concrete

Pandemic

How to see if your students are falling asleep?

Use machine learning for facial emotion recognition, aka our Project!

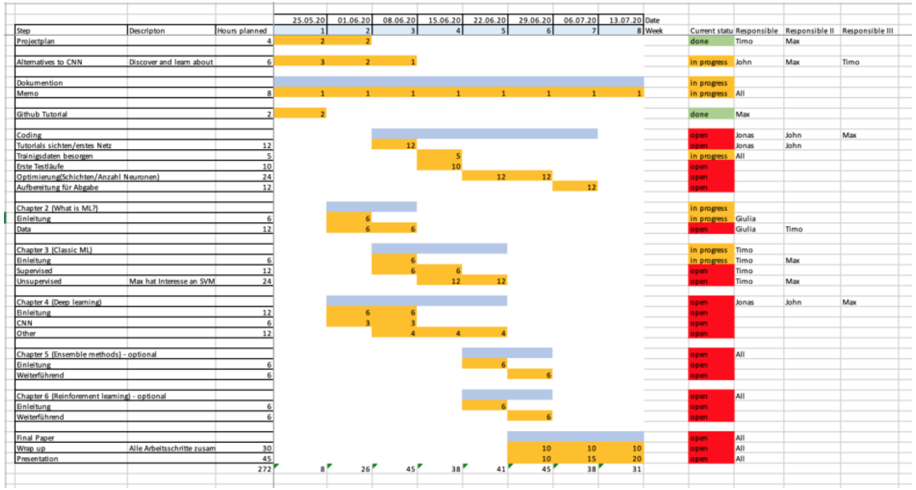
Goals

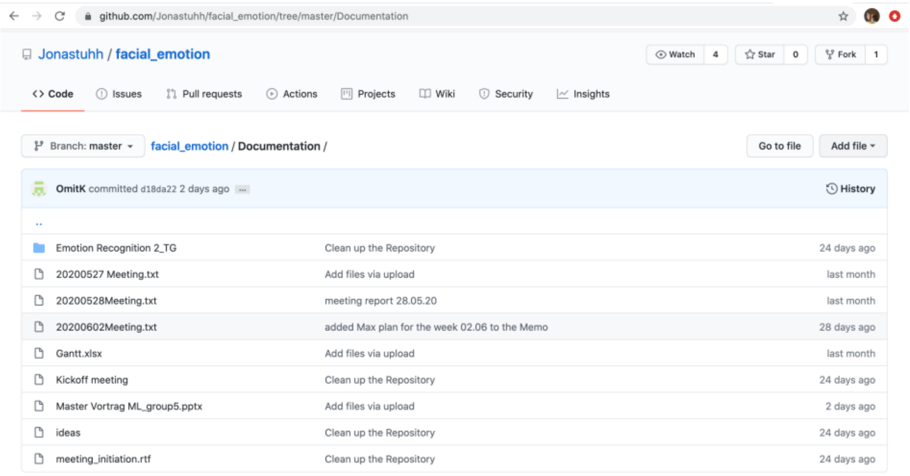
- Learn to work together: management and organization + GIT.
- Research different ML methods: which suits a given task best?
  - Write a document in Latex about the acquired knowledge.
- Implement at least one ML algorithm in Python and optimize its accuracy:
  - Costumer requirement: classify emotions of faces in an input image.

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Results

- Learn to work together: management and organization + GIT.
  - Gantt chart, weekly meetings, communication via Whatsapp.





Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Results

- Research different ML methods: which suits a given task best?
  - Write a document in Latex about the acquired knowledge.

PROJECT ML

University of Hamburg
Department of Mathematics

Jonas Eckhoff - Timo Greve
Max Lewerenz - Giulia Satiko Maesaka - John-Robert Wrage

Machine Learning Methods
Group 5

Contents

1 Introduction 1

2 What is ML? 1

3 Classical learning 3
3.1 KNN 3
3.2 Decision Trees 3
3.3 Support Vector Machine 5
3.3.1 Introduction 5
3.3.2 Mathematics behind SVMs 6
3.3.3 In our case 6

4 Neural Networks and Deep Learning 8
4.1 Convolutional Neural Networks CNN 8
4.2 Recurrent Neural Networks RNN 8
4.3 Generative adversarial networks GAN 9
4.4 Autoencoders 9

5 Ensemble Methods 10
5.1 AdaBoost 11

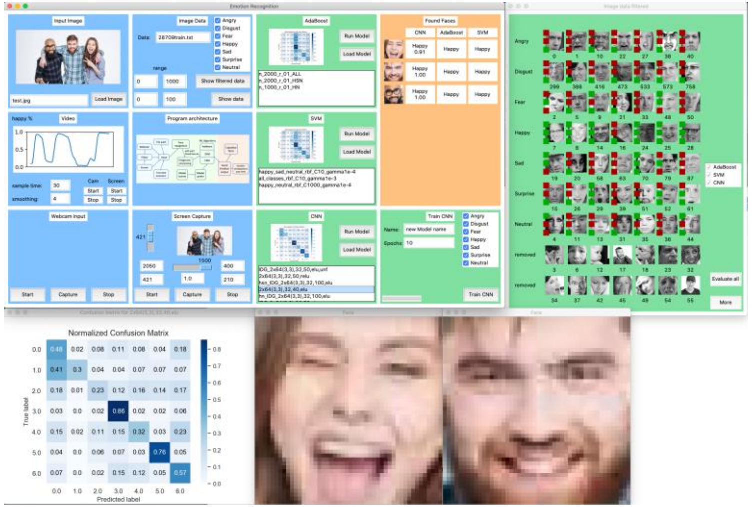
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Results

- Implement at least one ML algorithm in Python and optimize its accuracy:
  - Costumer requirement: classify emotions of faces in an input image.

Responsible	Hyp Space
John	KNN
Jonas	CNN
Max	SVM
Timo	AdaBoost

- GUI developed mainly by Jonas.



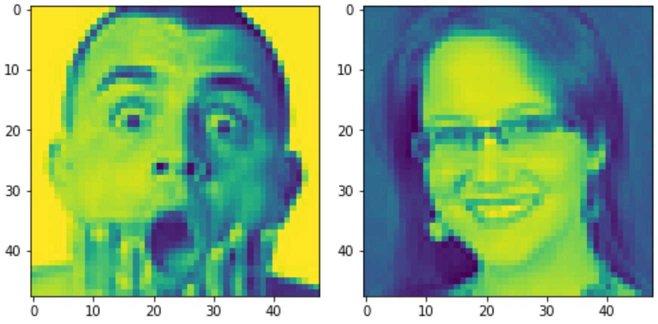
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

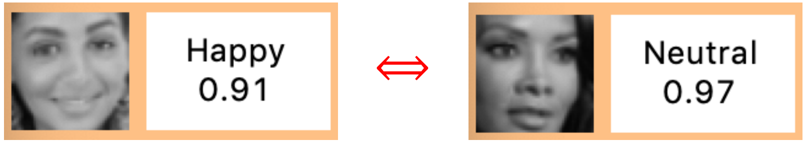
Algorithms

KNN	AdaBoost
SVM	CNN

Dataset

- Dataset from kaggle competition
- 48 x 48 Pixel grayscale images
- 28.709 Training-samples
  7.178 Test-samples
  = 35.887 samples
- 7 Emotions as labels
- Problems with the Dataset





Labeled as neutral

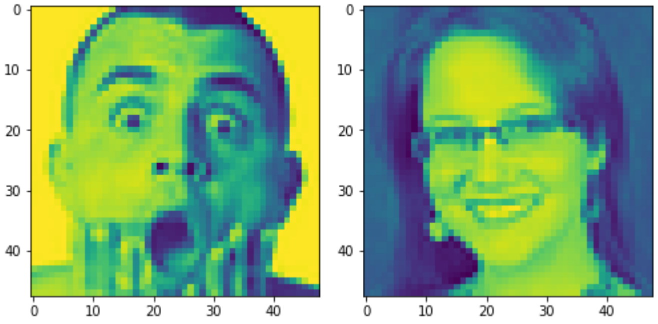
Labeled as happy

Dataset

- Preparing dataset (training / test)
- (48 x 48) = 2.304 dimensional input-data

$$\Longrightarrow x_n \in \{0, \dots, 255\}^{2.304}$$

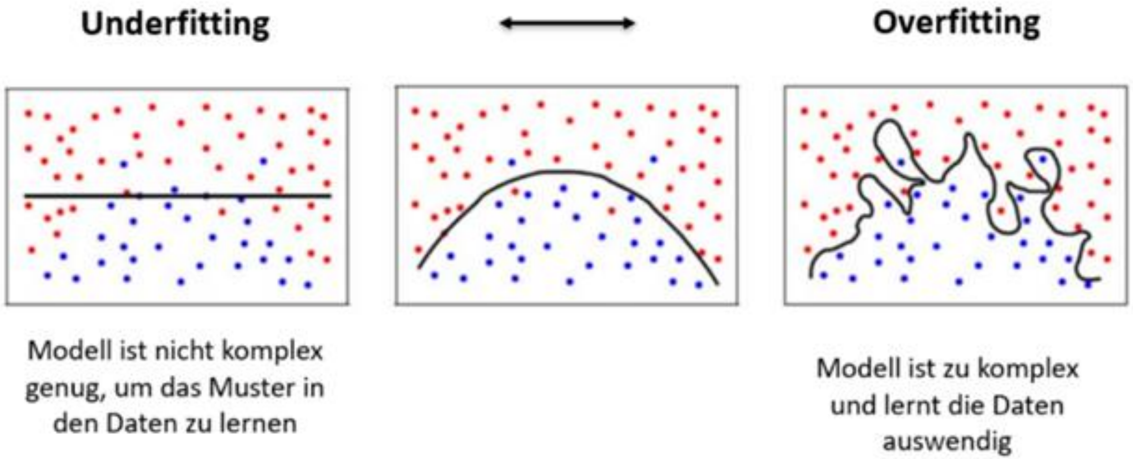
- Output: $y_n \in \{0, \dots, 6\}$



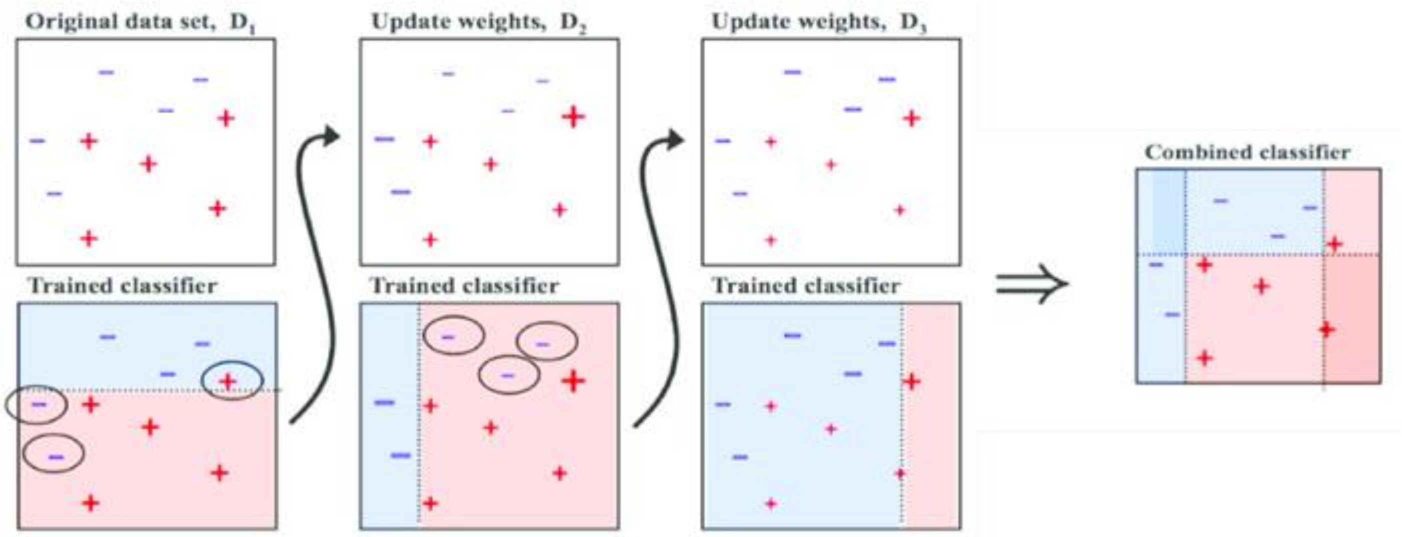

```
In [5]: file_name = './../Resources/dataset/fer2013.csv'
        df = pandas.read_csv(file_name)
        print(df)

       emotion                                             pixels        Usage
0            0  70 80 82 72 58 58 60 63 54 58 60 48 89 115 121...     Training
1            0  151 150 147 155 148 133 111 140 170 174 182 15...     Training
2            2  231 212 156 164 174 138 161 173 182 200 106 38...     Training
3            4  24 32 36 30 32 23 19 20 30 41 21 22 32 34 21 1...     Training
4            6  4 0 0 0 0 0 0 0 0 0 0 0 3 15 23 28 48 50 58 84...     Training
...        ...                                                ...          ...
35882        6  50 36 17 22 23 29 33 39 34 37 37 37 39 43 48 5...  PrivateTest
35883        3  178 174 172 173 181 188 191 194 196 199 200 20...  PrivateTest
35884        0  17 17 16 23 28 22 19 17 25 26 20 24 31 19 27 9...  PrivateTest
35885        3  30 28 28 29 31 30 42 68 79 81 77 67 67 71 63 6...  PrivateTest
35886        2  19 13 14 12 13 16 21 33 50 57 71 84 97 108 122...  PrivateTest

[35887 rows x 3 columns]
```

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

AdaBoost



AdaBoost


Original data set, D_1
Update weights, D_2
Update weights, D_3
Trained classifier
Trained classifier
Trained classifier
Combined classifier

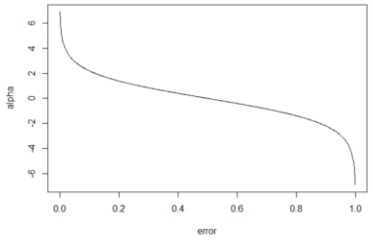
AdaBoost

Calculate error for first Stump G(x):

$$\mathrm{err}_1 = \frac{\sum_{i=1}^{N} w_i\, \mathbb{I}(y_i \neq G_1(x_i))}{\sum_{i=1}^{N} w_i}$$

„Importance of say“ for every Stump:

$$\alpha_1 = \log\left(\frac{1-\mathrm{err}_1}{\mathrm{err}_1}\right)$$



All samples get a new weight:

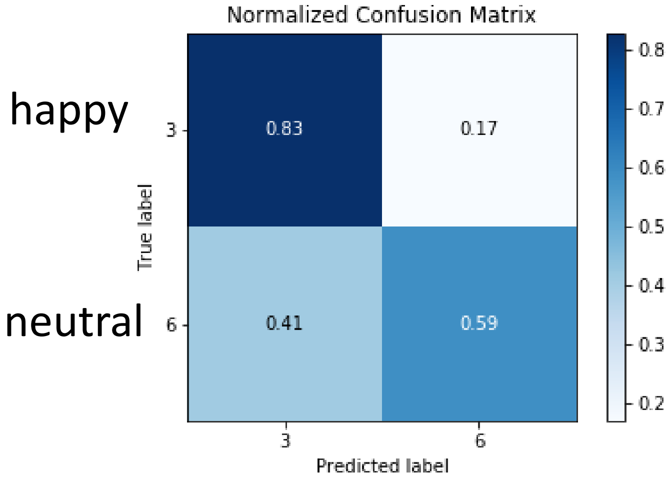
$$w_i = w_i * e^{(\alpha_1 * \mathbb{I}(y_i \neq G_1(x_i)))}$$

We predict a sample with M classifiers:

$$y^* = \mathrm{sign}\left[\sum_{j=1}^{M} \alpha_j G_j(x^*)\right]$$

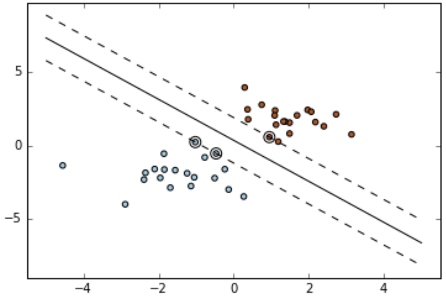
AdaBoost

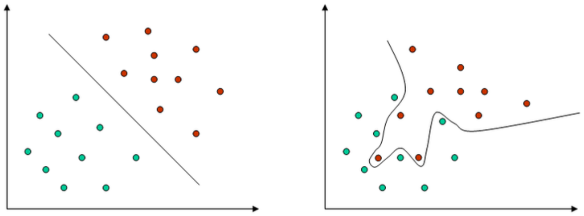
- Accuracy of 36.4% for all 7 classes
- Accuracy of 56.7% for Happy / Sad / Neutral
- Accuracy of 73.3% for Happy / Neutral



SVM – General Idea

- Input data: $X \in \mathbb{R}^{2.034 \times 28.709}$
  Labels vector: $Y \in \mathbb{R}^{28.709}$
- Find hyperplane to seperate data
  i.e.: $\min\limits_{w,b,s} \|w\|^2 + C \sum_{i=1}^{28.709} s_i,$
  s.t.: $y_i(w^T x_i + b) \geq 1 - s_i$ and $s_i \geq 0$
- „One-vs-one“ multiclass classification
- Dual problem allows for kernel trick





linear trennbar

nicht linear trennbar

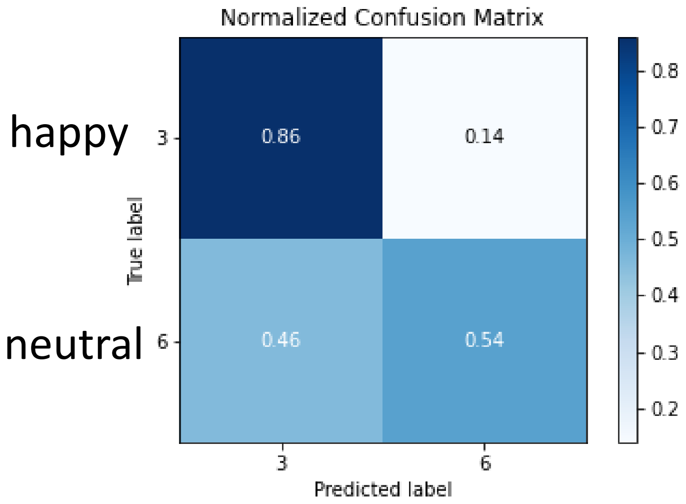
SVM – Parameter Choice

- import Scikit-learn
- Grid search
- Cross-validation
- Best parameter choice:
  Kernel = $\exp(-\gamma\|x - y\|^2)$
  C = 10,
  γ = 0.0001

```
Best parameters set found on development set:

{'C': 10, 'gamma': 0.0001, 'kernel': 'rbf'}

Grid scores on development set:

0.695 (+/-0.055) for {'C': 1, 'gamma': 0.001, 'kernel': 'rbf'}
0.299 (+/-0.001) for {'C': 1, 'gamma': 0.0001, 'kernel': 'rbf'}
0.708 (+/-0.037) for {'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}
0.711 (+/-0.028) for {'C': 10, 'gamma': 0.0001, 'kernel': 'rbf'}
0.684 (+/-0.036) for {'C': 100, 'gamma': 0.001, 'kernel': 'rbf'}
0.695 (+/-0.028) for {'C': 100, 'gamma': 0.0001, 'kernel': 'rbf'}
0.669 (+/-0.044) for {'C': 1000, 'gamma': 0.001, 'kernel': 'rbf'}
0.669 (+/-0.042) for {'C': 1000, 'gamma': 0.0001, 'kernel': 'rbf'}
0.628 (+/-0.036) for {'C': 1, 'kernel': 'linear'}
0.620 (+/-0.021) for {'C': 10, 'kernel': 'linear'}
0.604 (+/-0.036) for {'C': 100, 'kernel': 'linear'}
0.609 (+/-0.034) for {'C': 1000, 'kernel': 'linear'}

Detailed classification report:

The model is trained on first 2000 entries of development set.
The scores are computed on the first 200 entries of evaluation set.

              precision    recall  f1-score   support

           3       0.61      0.93      0.74       115
           6       0.67      0.19      0.30        84

    accuracy                           0.62       199
   macro avg       0.64      0.56      0.52       199
weighted avg       0.63      0.62      0.55       199
```

SVM

- Accuracy of 42.7% for all 7 classes
- Accuracy of 56.7% for happy / sad /neutral
- Accuracy of 73% for happy / neutral



CNN

What type of layers to use?

- convolutional
- fully connected
- layer size
- batch normalization
- max pooling

Which activation function?

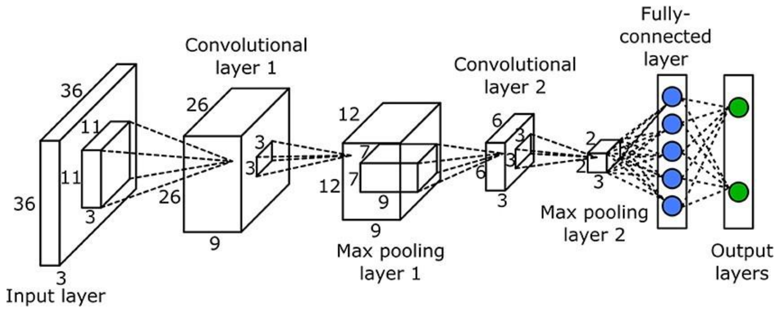
- SoftPlus
- Leaky ReLU
- Rectified linear unit (ReLU)
- Exponential linear unit (ELU)

How to prevent overfitting?

- dropout
- image generator
- early stopping

CNN architecture

Agenda
Intro
Project
Product
Algorithms
Presentation
Conclusion



Fully connected layer:
Classifies the output of the convolution and pooling process

Convolutional layer:
Uses convolution to find certain features in an image

Max pooling layer:
Reduces the complexity of a network by combining several nodes to the maximum value

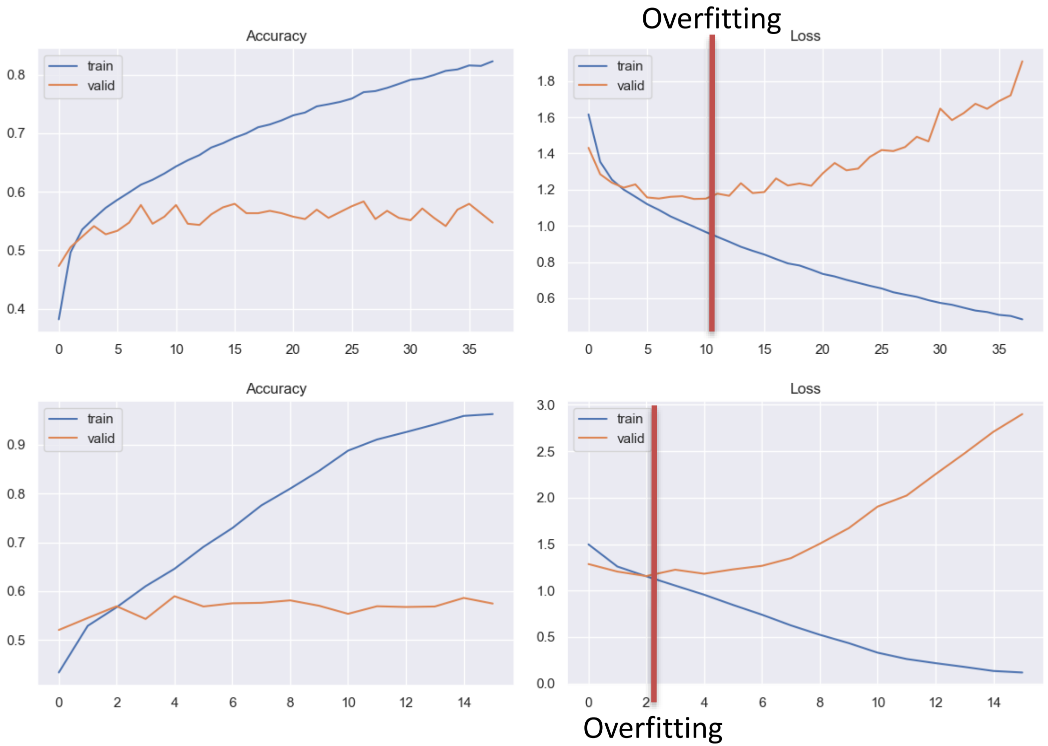
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

CNN

ReLU

Activation function

ELU



- Training process with ELU is a lot faster
  - 90% trainings accuracy after 10 epochs vs 80% after 30 epochs
- Problem of rising validation loss function: Overfitting

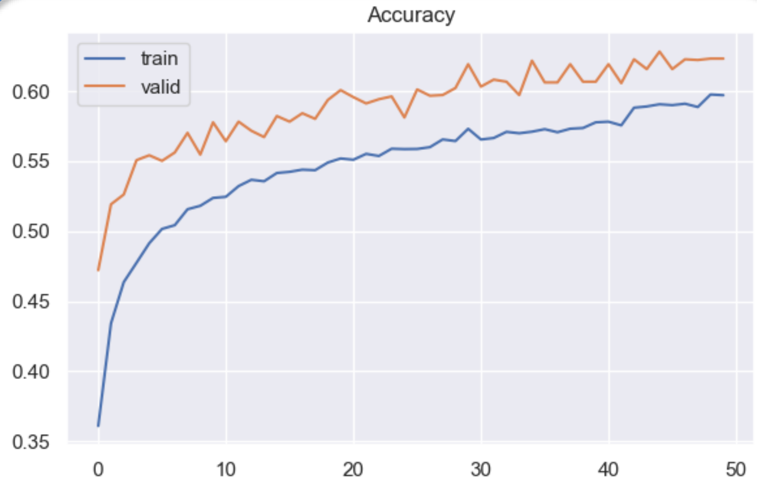
CNN

Agenda
Intro
Project
Product
Algorithms
Presentation
Conclusion

Overfitting Problem:

<u>Image Generator</u>
Instead of using static training data, it uses images from the trainings data that are randomly shifted, rotated, mirrored, scaled and sheared.

Early stopping and restoring weights after recognising stagnant validation accuracy / validation loss



Accuracy with Image Generator

CNN

Overfitting Problem:

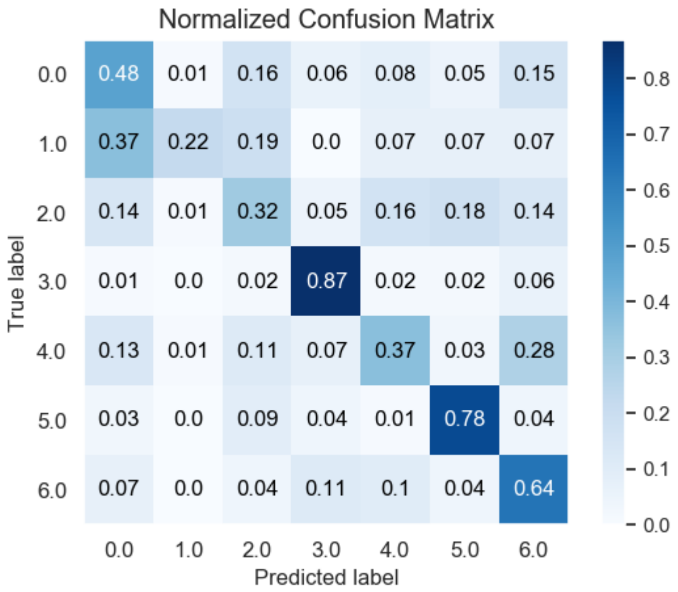
<u>Image Generator</u>
Instead of using static training data, it uses images from the trainings data that are randomly shifted, rotated, mirrored, scaled and sheared.

Early stopping and restoring weights after recognising stagnant validation accuracy / validation loss

Loss with Image Generator

CNN

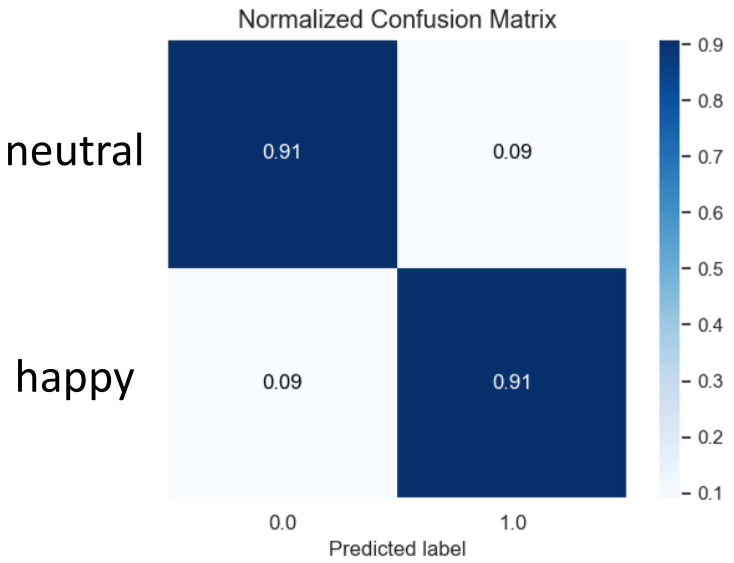
angry
disgust
fear
happy
sad
surprise
neutral



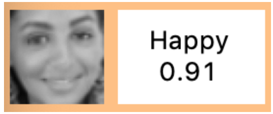
- 62% accuracy with 7 emotions
  - good with happy and surprise
  - trouble with disgust, fear and sad
  - disgust is almost never predicted
    - least represented in data set

CNN

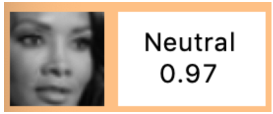
CNN trained for 2 emotions



- 91% accuracy with 2 emotions
  - many misclassifications are debatable



labeled as neutral

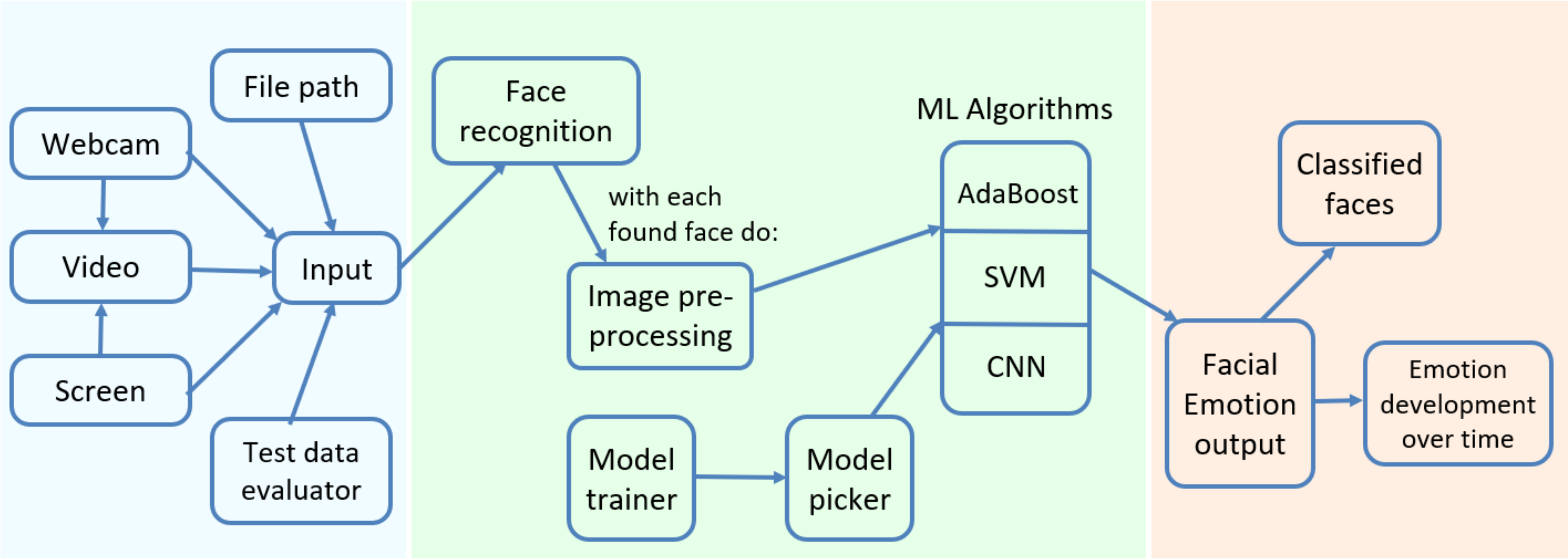


labeled as happy

Comparison

	Adaboost	SVM	CNN
7 Emotions	36,4 %	42,7 %	62 %
Happy / Sad / Neutral	56,7 %	56,7 %	78 %
Happy / Neutral	73,3 %	73 %	91 %

Product Presentation



Conclusion

Goals	Final status
Learn to work together	We established a frequent comunication and discussed the upcoming problems/ideas at least weekly
Research different ML methods	We collected different algorithms to tackle our given problem. We explained the idea behind our algorithms to the others in our meetings and wrote a brief overview for each algorithm in LaTeX.
Implement at least one ML algorithm in Python and optimize its accuracy	We implemented four algorithms which try to solve the initial problem. Even though we knew from the beginning that CNN seems to be the best choice we build other solutions to confirm our initial thoughts.

Responsibilities

Name	Tasks / Roles
Max	Project Manager, Git introduction, SVM implementation, SVM documentation, dataset
Giulia	Documentation of goals and progress, general machine learning documentation, dataset
Jonas	NN implementation, NN documentation, GUI, dataset
Timo	Classical learning documentation, AdaBoost documentation, adaboost implementation, dataset, presentation
John	KNN implementation, dataset

Thank you!

References:

For all references please have a look at „ML-Methods“

AdaBoost:

Introduction to AdaBoost https://towardsdatascience.com/understanding-adaboost-for-decision-treeff8f07d2851

scikit-learn: https://scikit-learn.org/stable/modules/ensemble.html https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html

Wikipedia: Boosting https://de.wikipedia.org/wiki/Boosting

SVM:

Martin Lotz: Mathematics of Machine Learning. Lecture Notes, Warwick (UK), 2020.

scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html https://scikit-learn.org/stable/modules/svm.html

Coursera: Machine Learning, by Stanford University https://www.coursera.org/learn/machine-learning/home/welcome

Wikipedia: Support vector machine https://en.wikipedia.org/wiki/Support_vector_machine

CNN

https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

https://brilliant.org/wiki/convolutional-neural-network

https://www.kaggle.com/gauravsharma99/facial-emotion-recognition