

Machine Learning – How it works



Cat

Burglar breaking
in a house

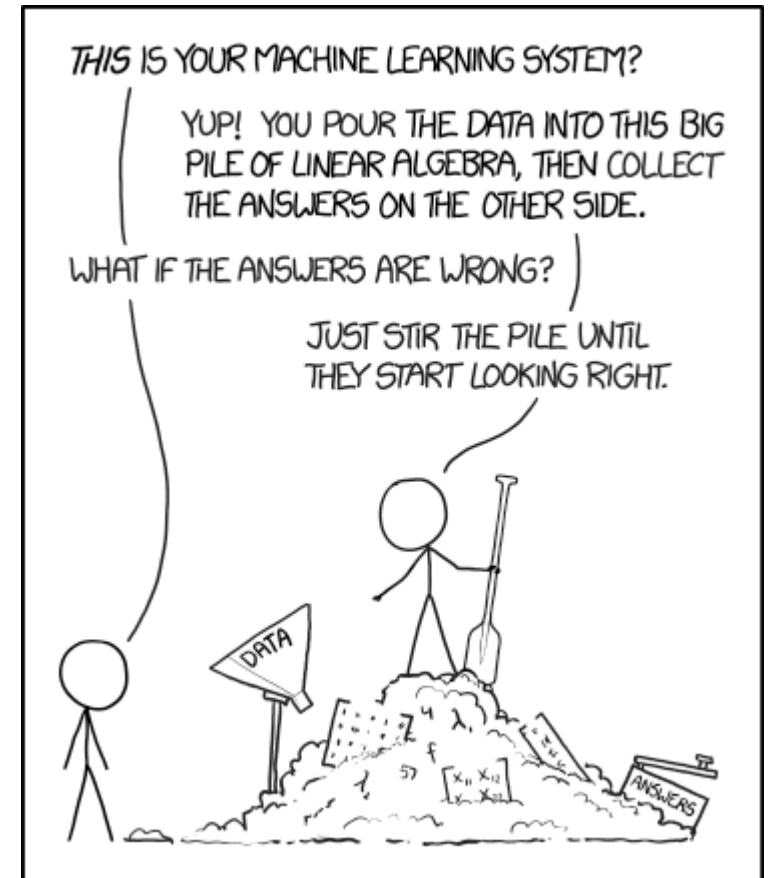
How does my computer do it ???

Machine Learning – How it works

- Traditional Software Development:
Software behavior determined by

Logic written by developers
- Machine Learning:
Software behavior determined by

Data/examples

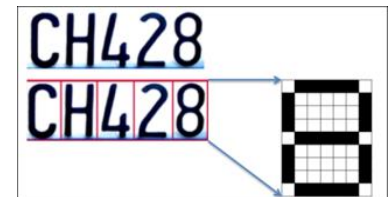
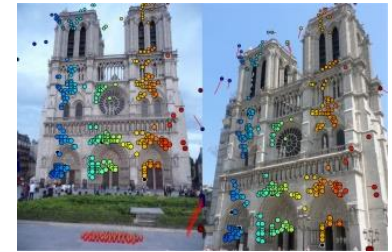


Machine Learning – How it works

Convert RGB pixels to meaningful numbers that can be understood by the machine (hand-crafted features)

=> Feature Extraction

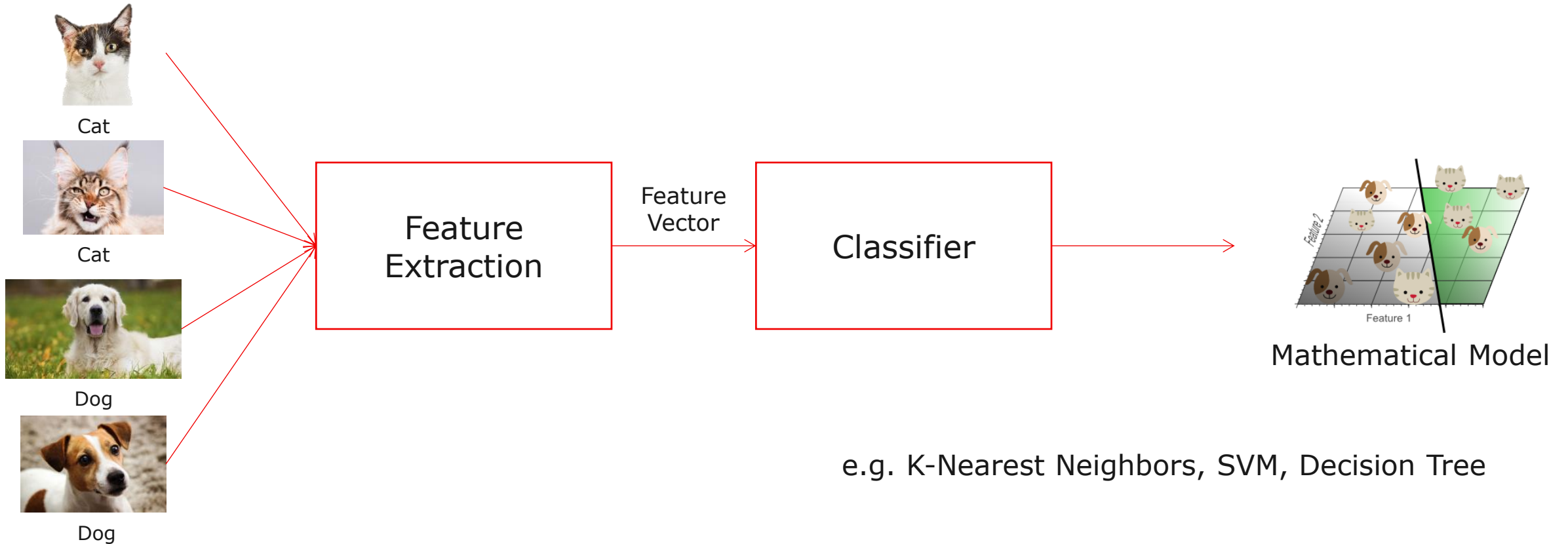
- Colors
- Shapes
- Points of interest
- Motion (video)
- Text/Numbers



Note: We can also build hand-crafted features for audio but that is not in the scope of this track

Machine Learning – How it works

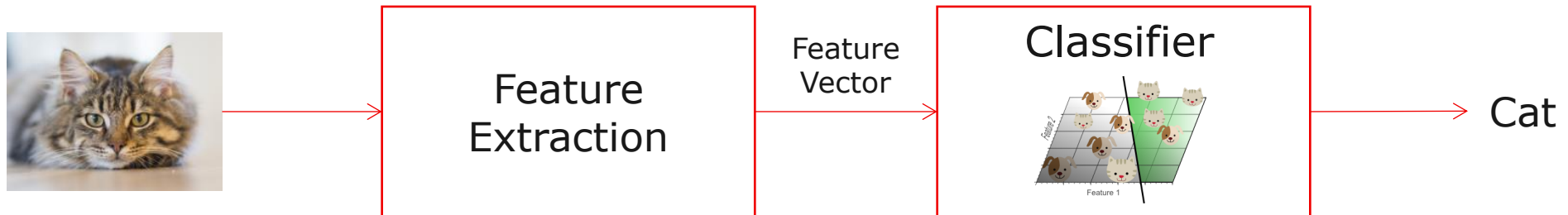
From meaningful features, we can learn patterns !
We feed what we call a “Classifier” with labelled data and it will build a mathematical model (Optimization). This is the training step.



e.g. K-Nearest Neighbors, SVM, Decision Tree

Machine Learning – How it works

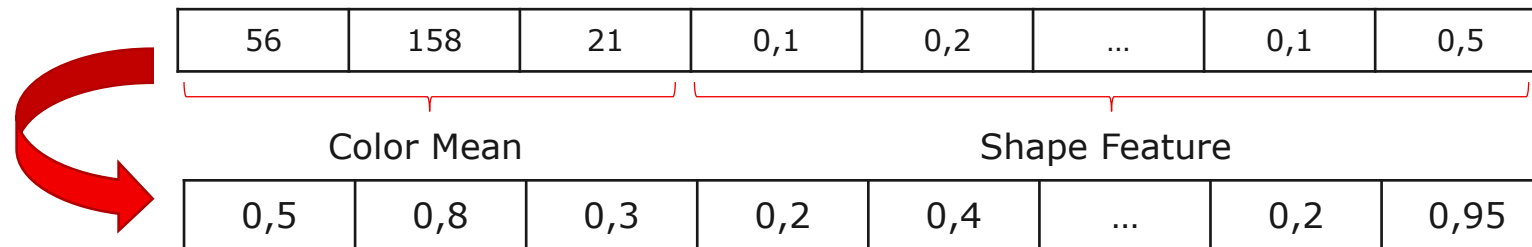
Once the classifier is trained (optimized mathematical model), we can Feed it with new samples and predict its label.



Machine Learning – How it works

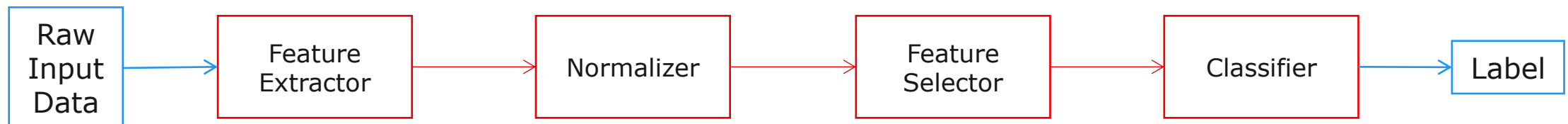
This pipeline is usually too simplistic to perform well

- Reducing the feature values is good for speed
- When concatenating feature vectors, some features can have larger values than others

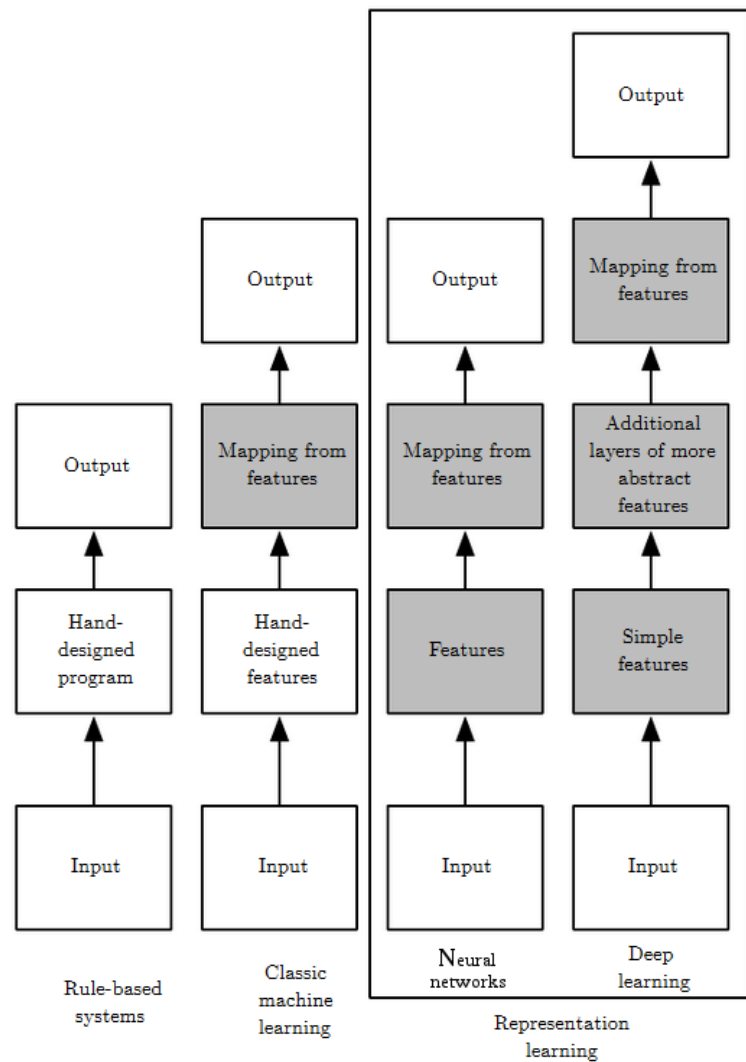


=> Normalize the feature vectors

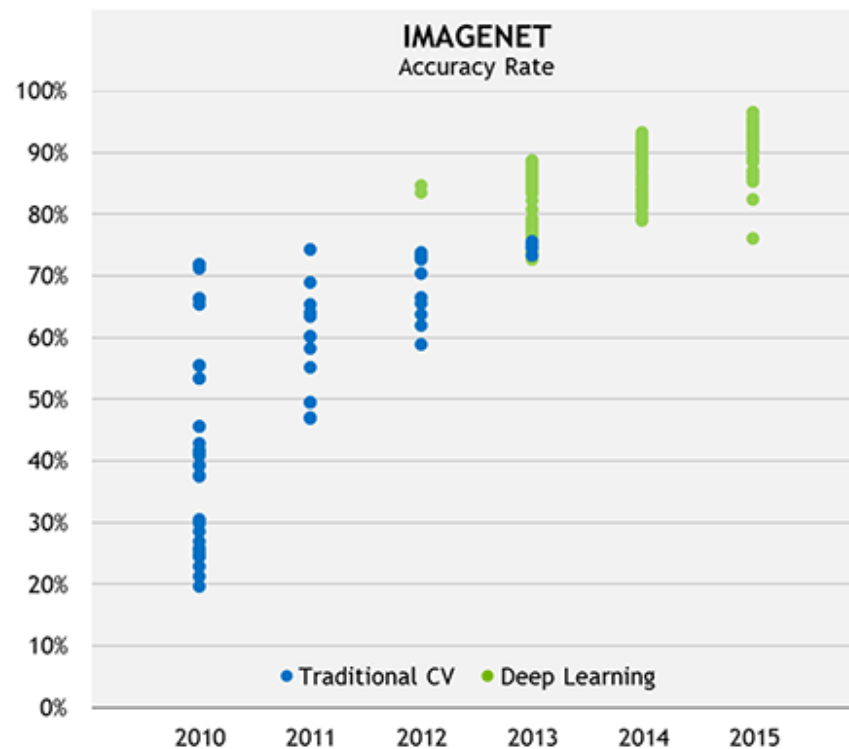
- Select the relevant features only! E.g. Principal Component Analysis



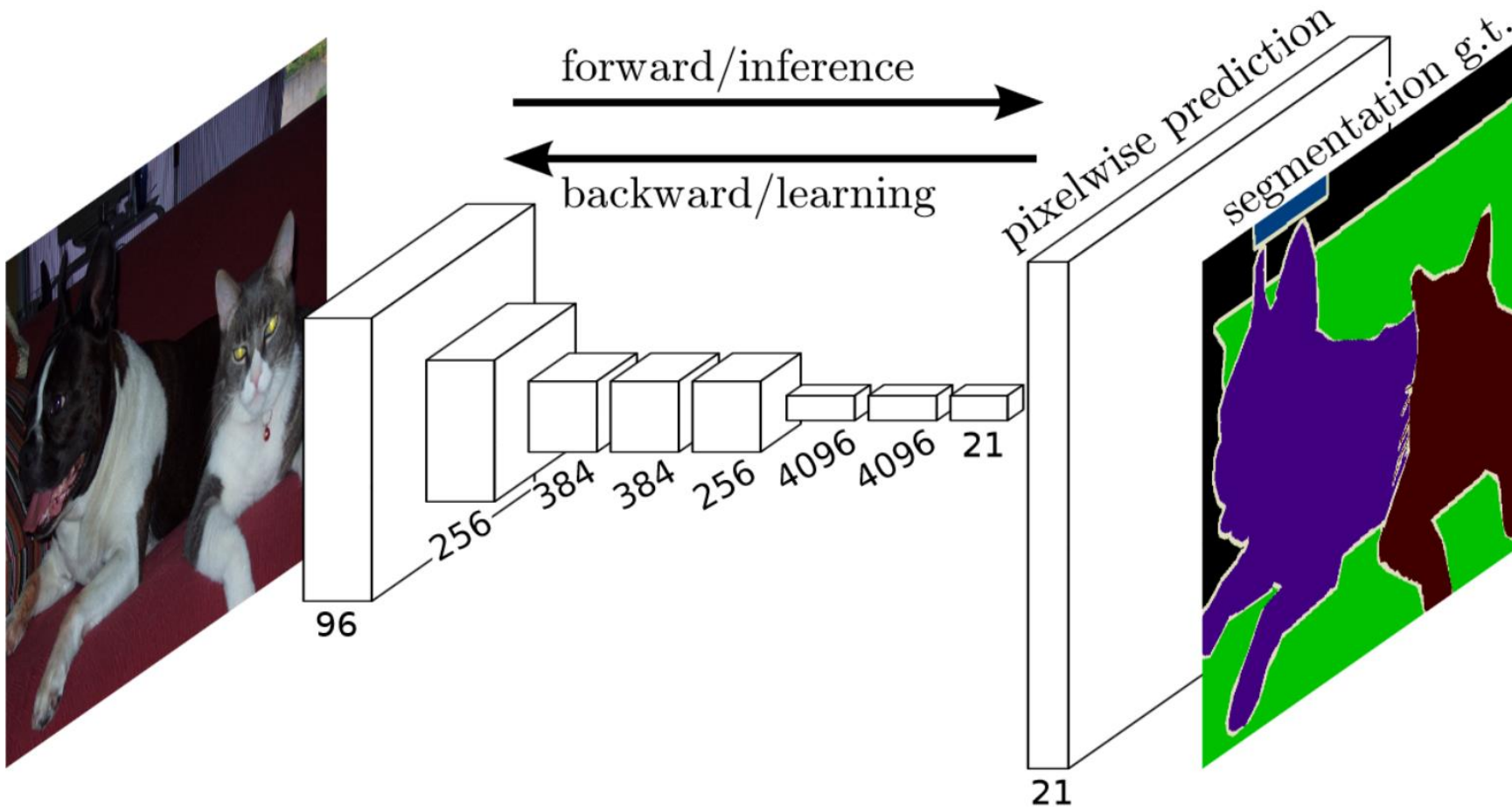
Machine Learning – How it works



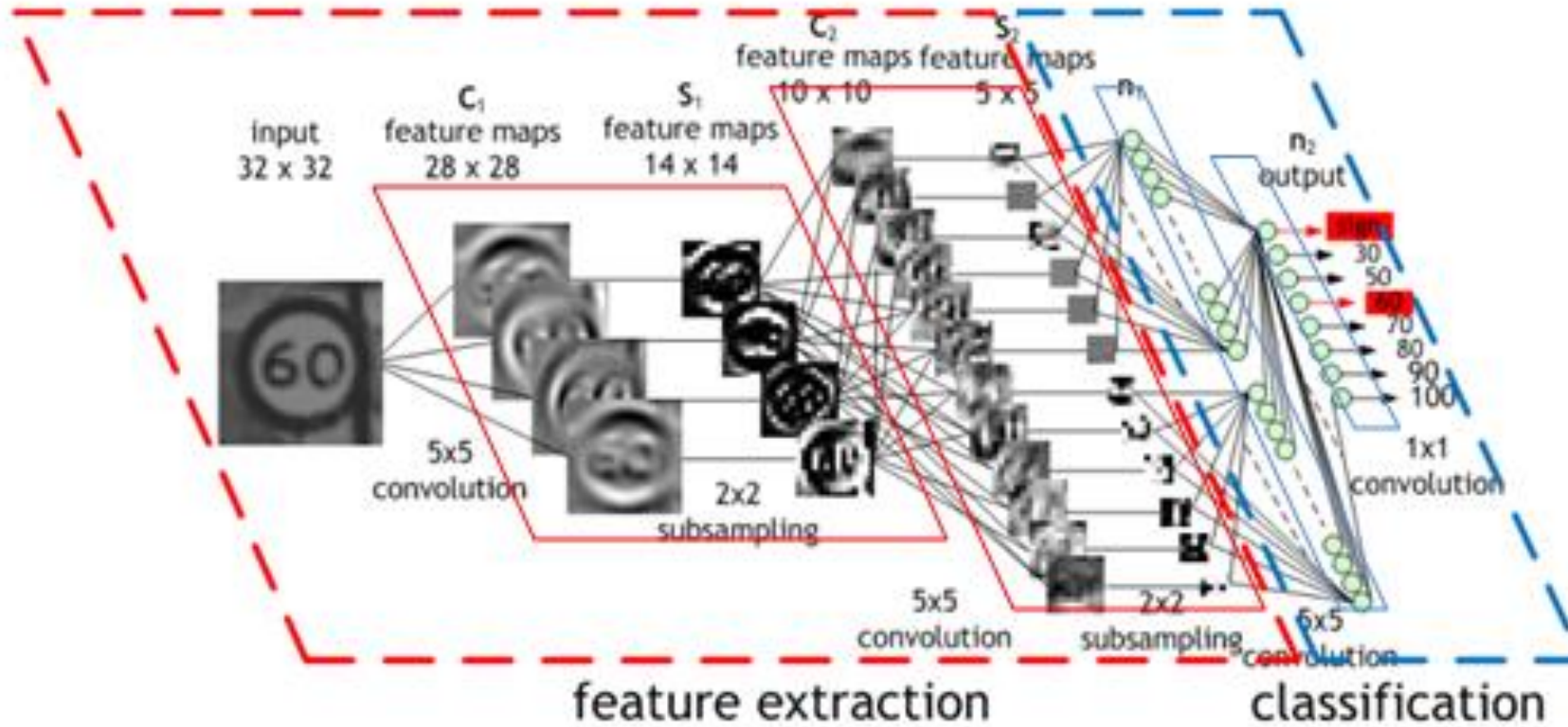
2015: A MILESTONE YEAR IN COMPUTER SCIENCE



Machine Learning – How it works



Machine Learning – How it works



Machine Learning – How it works

A mostly complete chart of

Neural Networks

©2016 Fjodor van Veen - asimovinstitute.org

Backfed Input Cell

Input Cell

Noisy Input Cell

Hidden Cell

Probabilistic Hidden Cell

Spiking Hidden Cell

Output Cell

Match Input Output Cell

Recurrent Cell

Memory Cell

Different Memory Cell

Kernel

Convolution or Pool

Perceptron (P)



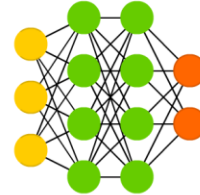
Feed Forward (FF)



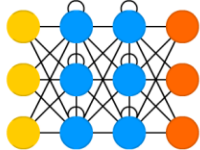
Radial Basis Network (RBF)



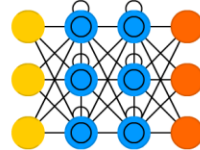
Deep Feed Forward (DFF)



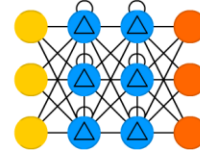
Recurrent Neural Network (RNN)



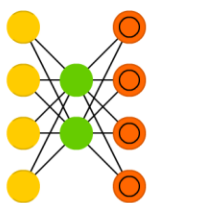
Long / Short Term Memory (LSTM)



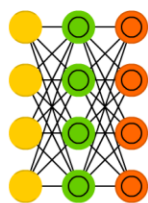
Gated Recurrent Unit (GRU)



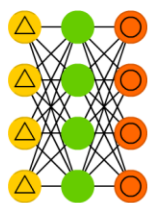
Auto Encoder (AE)



Variational AE (VAE)



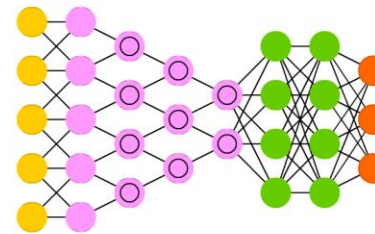
Denoising AE (DAE)



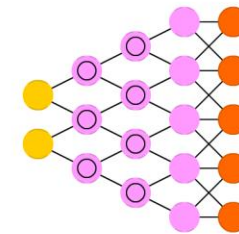
Sparse AE (SAE)



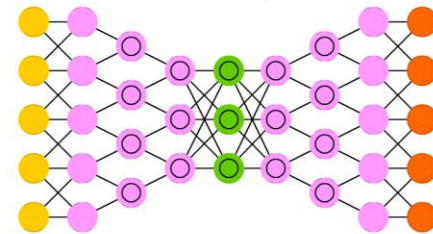
Deep Convolutional Network (DCN)



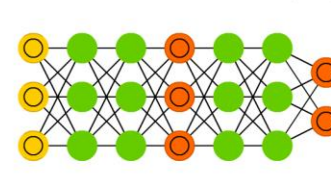
Deconvolutional Network (DN)



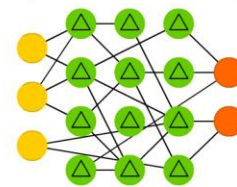
Deep Convolutional Inverse Graphics Network (DCIGN)



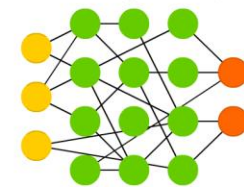
Generative Adversarial Network (GAN)



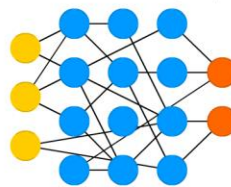
Liquid State Machine (LSM)



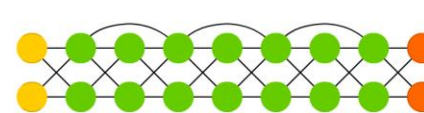
Extreme Learning Machine (ELM)



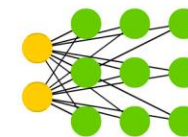
Echo State Network (ESN)



Deep Residual Network (DRN)



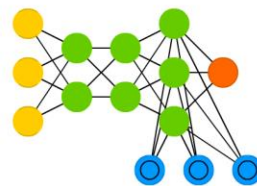
Kohonen Network (KN)



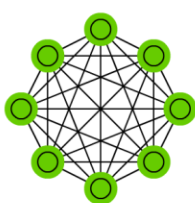
Support Vector Machine (SVM)



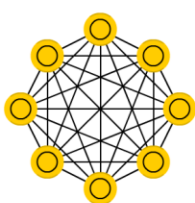
Neural Turing Machine (NTM)



Markov Chain (MC)



Hopfield Network (HN)



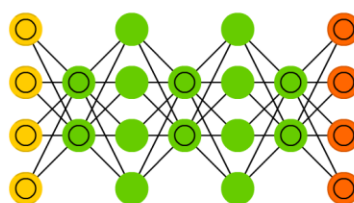
Boltzmann Machine (BM)



Restricted BM (RBM)



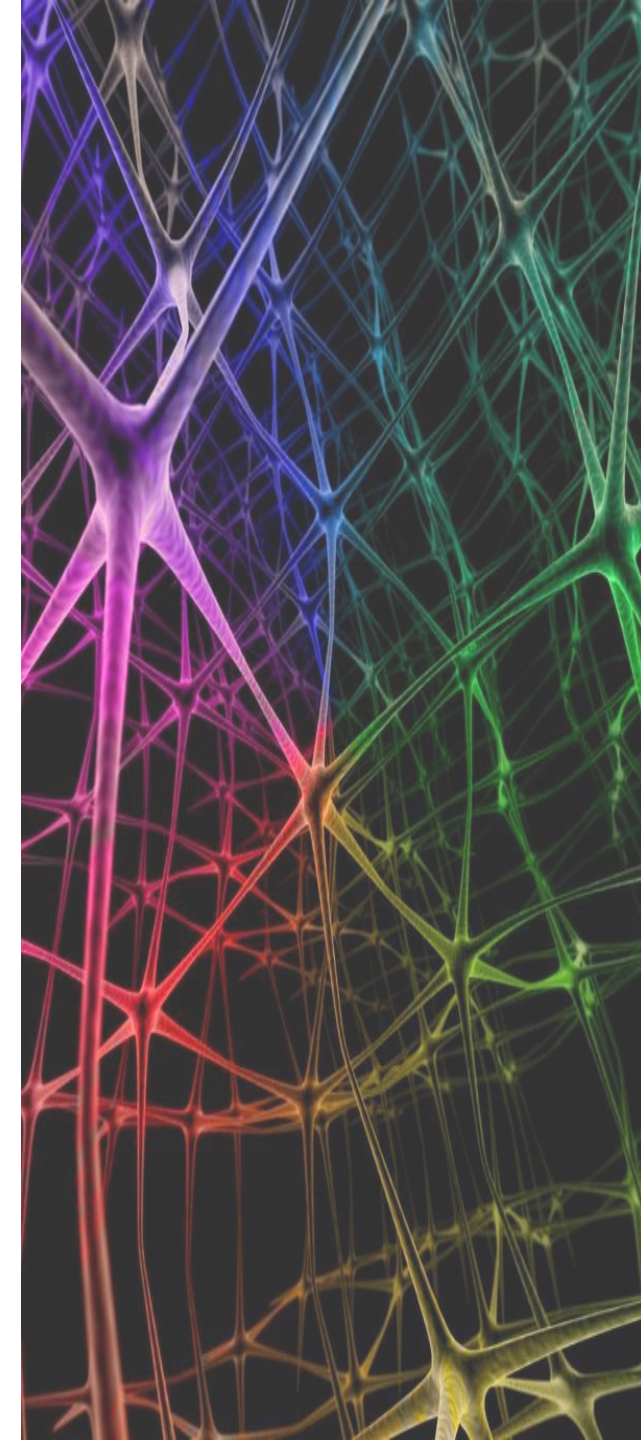
Deep Belief Network (DBN)





Questions?

About us? Barco? Machine Learning?



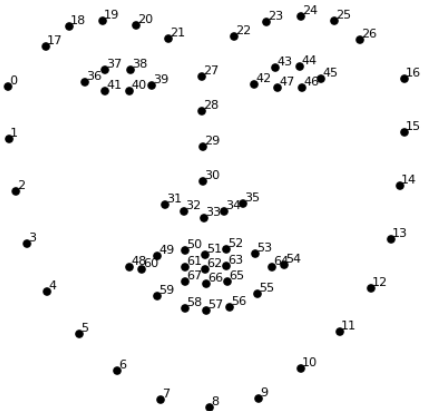
Hands-on – Introduction

Some Machine Learning, some coding and ... some fun!

Machine learning to detect facial landmarks

Machine learning to detect face configuration and emotion

Some code to overlay sprites on your face via your webcam

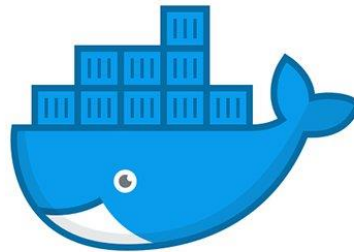
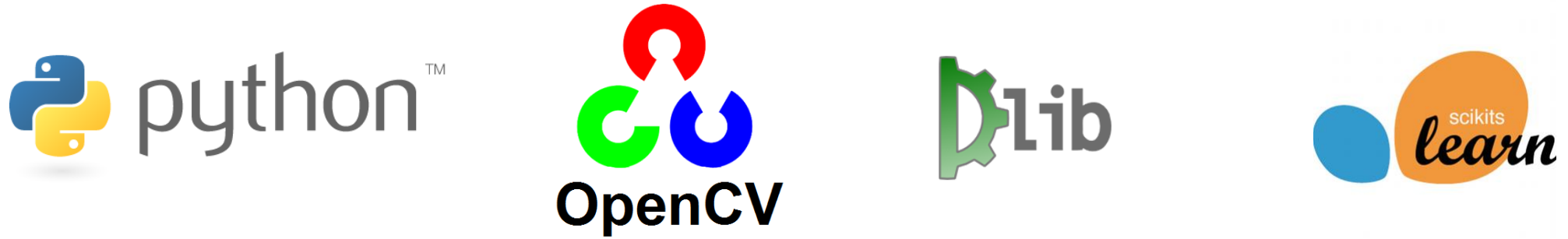


Open Mouth

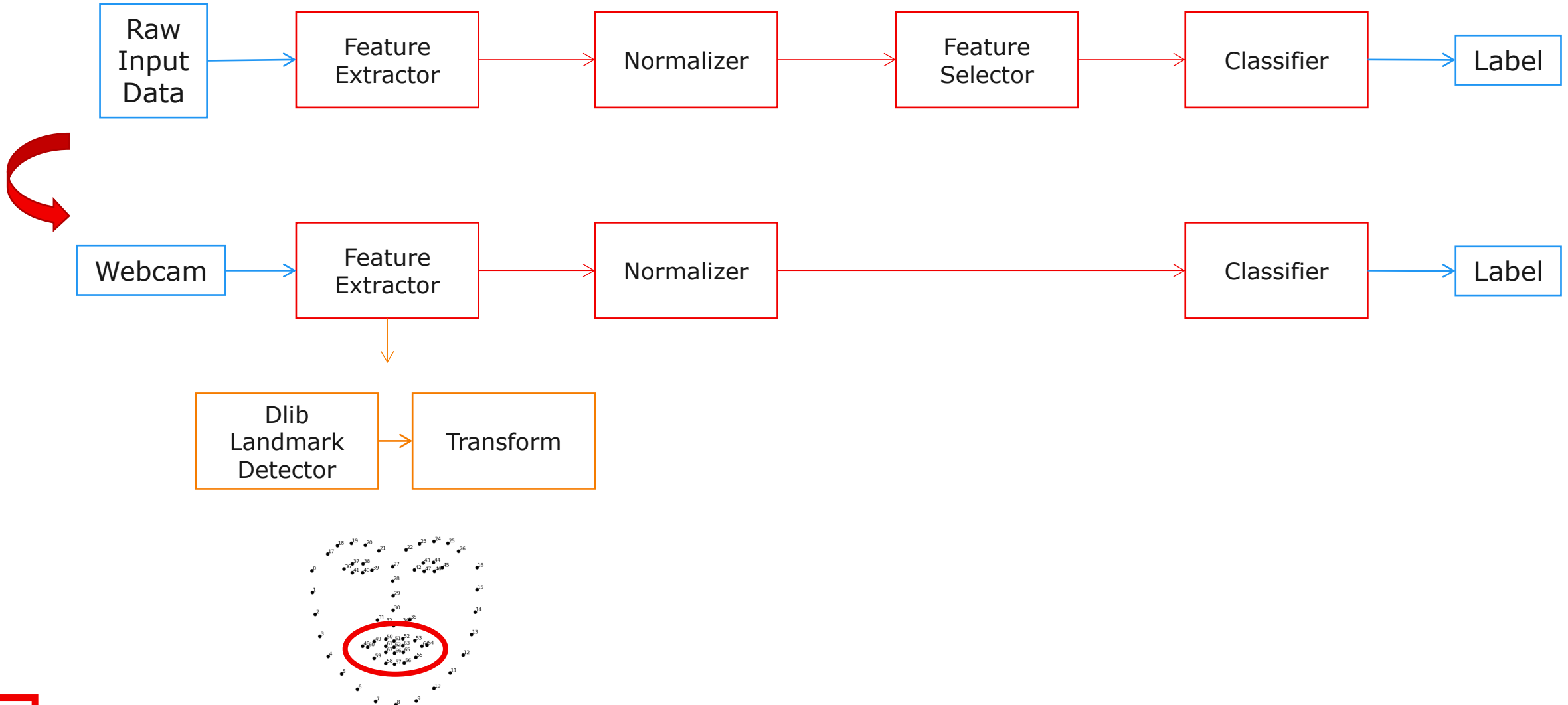


It is show time!

What will we use for this?



Same pipeline as before



- ▼ student-trip-lyon-master C:\Users\B\ul>
 - > docker
 - > examples
 - > sounds
 - > sprites
 - ▼ src
 - > imageoverlay
 - ▼ machinelearningsuite
 - __init__.py
 - classifier.py
 - configuration.py
 - featureprocessing.py
 - filter.py
 - landmarkdetector.py
 - machinelearningsuite.py
 - normalizer.py
 - predictorinterface.py
 - > tests
 - Barco_Facial_Landmarks.ipynb
 - jupyter_notebook_config.py
 - Landmarksreference.png
 - requirements.txt
 - run.sh
 - run_jupyter.sh
 - > External Libraries

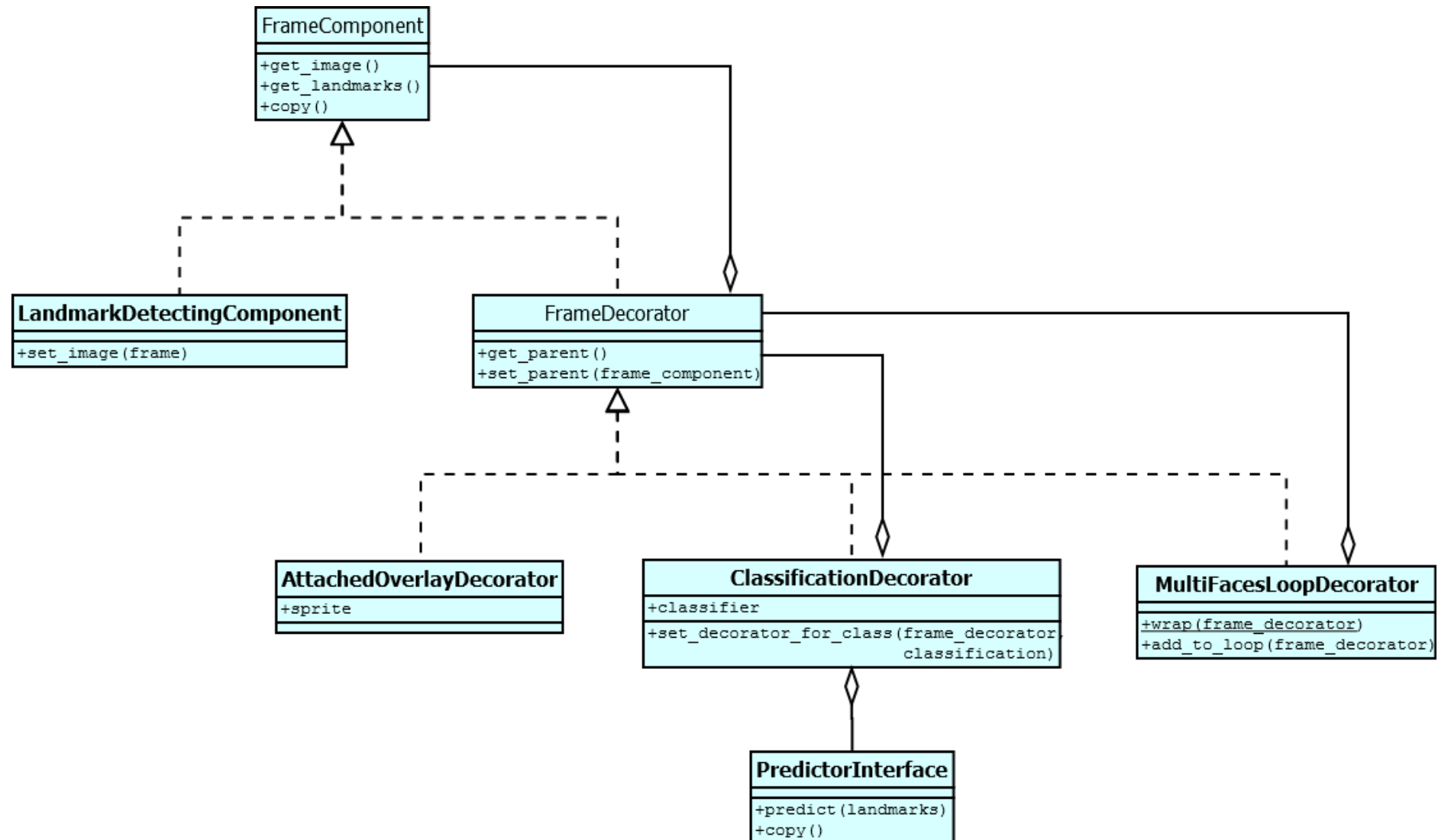
```
class Classifier:
    def __init__(self, configuration):
        self.configuration = configuration
        self.classifier = svm.SVC(kernel='linear', C=1000)
        # self.classifier = svm.SVC()
        # self.classifier = svm.LinearSVC()
        # self.classifier = svm.SVC(decision_function_shape='ovo')

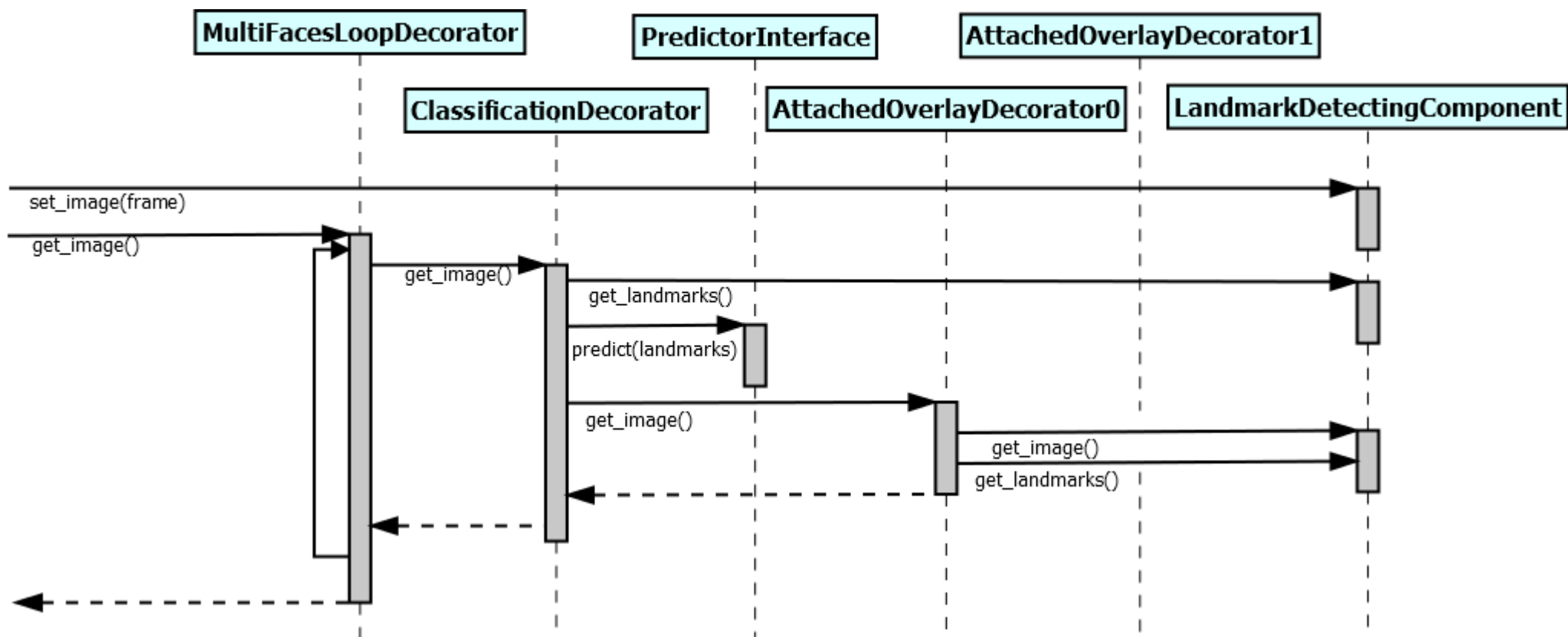
    def load_configuration(self):
        if self.configuration.classifier:
            self.classifier = self.configuration.classifier

    def save_configuration(self):
        self.configuration.classifier = self.classifier

    def train(self):
        X = self.configuration.data_values_normalized
        y = self.configuration.data_labels
        self.classifier.fit(X, y)
        self.save_configuration()

    def predict(self, data):
        return self.classifier.predict(data)
```



```
13     def predicting_example():
14         # Instantiate a new landmark detector
15         detector_data_path = '../data/shape_predictor_68_face_landmarks.dat'
16         landmark_detector = LandmarkDetector(predictor_file=detector_data_path)
17
18         # Create a frame component with landmarks
19         base_component = LandmarkComponent(landmark_detector)
20
21         # Instantiate and initialize the trained predictor
22         predictor = PredictorInterface('../examples/emotion.pkl')
23         predictor.initialize()
24
25         # Add decorator for the predictor
26         predictor_decorator = ClassDecorator(parent_component=base_component, classifier=predictor)
27         sunglasses = SpriteDecorator(base_file_name='../sprites/sunglasses')
28         eyes = SpriteDecorator(base_file_name='../sprites/eyes')
29         predictor_decorator.set_decorator_for_class(sunglasses, 0)
30         predictor_decorator.set_decorator_for_class(eyes, 1)
31
32         multifaces = AllFaces.wrap(predictor_decorator)
33
34
35
36
37
38
39
40
41
42
43         base_component.set_image(frame)
44         output = multifaces.get_image()
```

- Get docker image from:
docker pull bapha/student-trip-lyon
- Get source code from:
git clone <https://github.com/karmomoens/ml-in-cv.git>
- ./run.sh
- Have a look in ./examples


1. Show facial landmarks
2. Add overlay
3. Train classifier: Open/Closed mouth
4. Link overlay to classifier
5. Go crazy!

- Create a text overlay
- Face swap
- Draw a new sprite at run-time
- Create animations
- Program a (competitive) game (e.g. eating dots of the screen)
- Train a classifier for tilting your head in one or the other direction
- Create a nodding yes or no classifier
- Use the color of pixels to make a prediction (e.g. eye or hair color)
- Check out examples in dlib to label and train new object detectors and shape predictors


Thanks for attending! Hope you enjoyed it!



ENABLING BRIGHT OUTCOMES

 | youtube.com/BarcoTV

 | linkedin.com/company/Barco

 | twitter.com/Barco

 | facebook.com/Barco

