Deep Learning Bootcamp – Transfer Learning

Technische Hochschule Ingolstadt

KI-basierte Optimierung in der Automobilproduktion

Technische Hochschule Ingolstadt

Motivation

Image Recognition: Classification Based On Images

ImageNet **L**arge **S**cale **V**isual **R**ecognition **C**hallenge

- Established 2010

ImageNet

- ImageNet database: https://image-net.org/
- Consists of >14 Million annotated images of >20 000 categories
- ImageNet Large Scale Visual Recognition Challenge (***ILSVRC***):
  - Uses 1000 classes
  - Widely used as a benchmark for model architectures
  - Famous architectures designed for ***ILSVRC***:
    - Pascal VOC, Alexnet, Inception, ResNet
- Well-performing models must be able to differentiate a huge variety of different image classes

ImageNet

How are well-performing architectures developed?

- Usually: gradual improvements of previous approaches
- Some creativity and good ideas (batch-norm, gradient flow, residual connections)
- **Mostly: Huge amounts of data, Tons of computational power and time!**

→ It is very hard to discover/train similar performing architectures without the resources of Google, Facebook, …

→ **Idea: let's use their work to solve our simpler problem!**

Transfer Learning

Transfer Learning - Definition

- We want to use existing well-working approaches and trained models to a problem to solve a different, but related problem

- Usually: We want to transfer general knowledge to a more specific task

- **Examples:**
  - Dogs vs. Cats → Giraffes vs Elephants
  - Handwritten characters → Handwritten digits
  - ImageNet → Nearly every other image recognition task

Unambiguous defect (y=1) All labelers will agree

Ambiguous whether to classify as defect Even expert labelers disagree

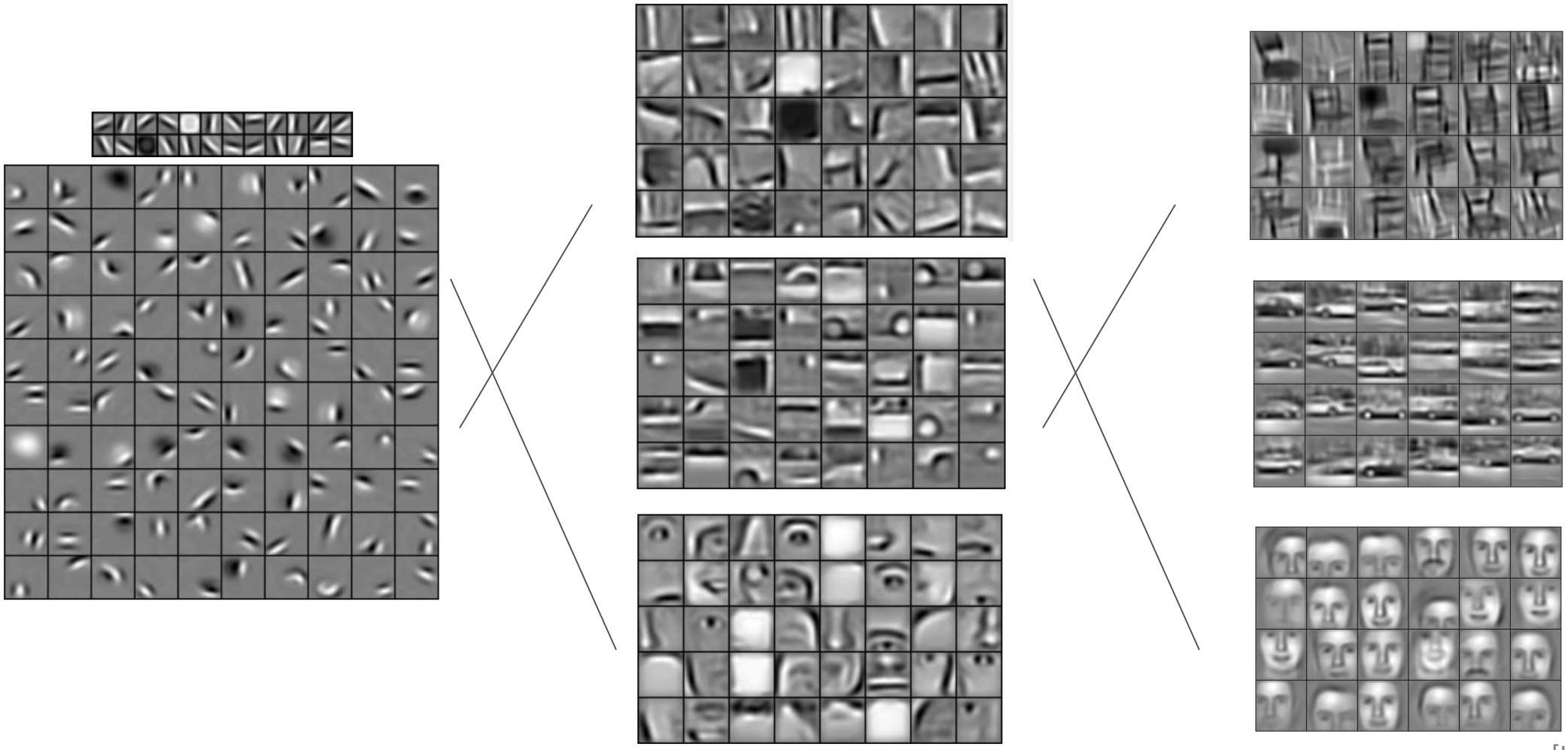
Unambiguous no defect (y=0) All labelers will agree

Convolutional Neural Nets are Feature Extractors

Low level features

Mid level features

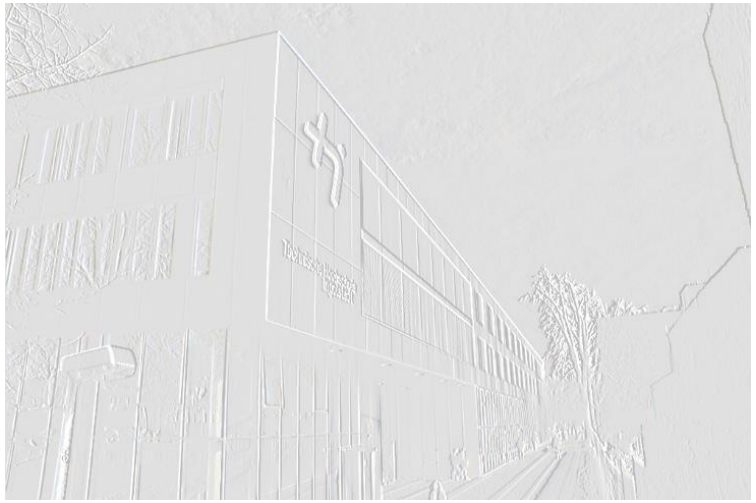
High level features



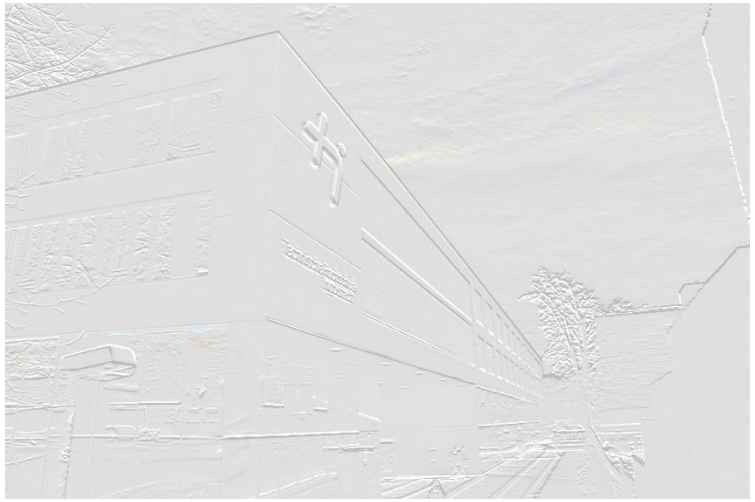
[Lee et. al, ICML 2009]

Transfer Learning – How can it work?

- Trained CNNs are excellent **feature extractors**
- Their way of feature extraction is similar to the human‘s visual system
- They extract meaningful representations from images, which makes classification easier



- Because the models are already pretrained, we need less data! The models are already „able“ to
  - Detect edges
  - Detect squares …

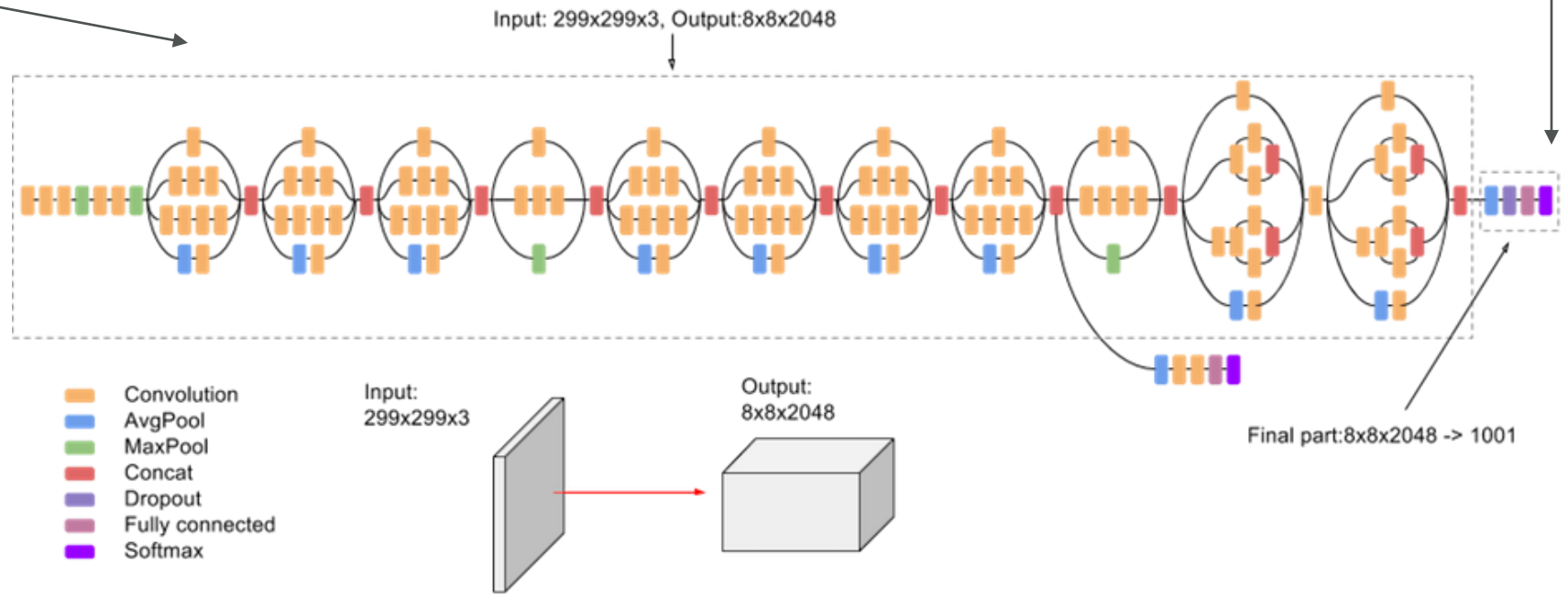


Transfer Learning - Usage

- Because of its huge variety, model architectures trained on ImageNet data are very often capable of solving a more special task
- How can we use these architectures?

We change the last layer of this part

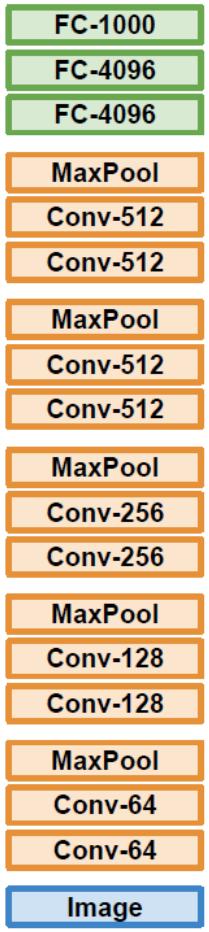
We want to keep this part (here lies the „intelligence“). This includes the trained weights!



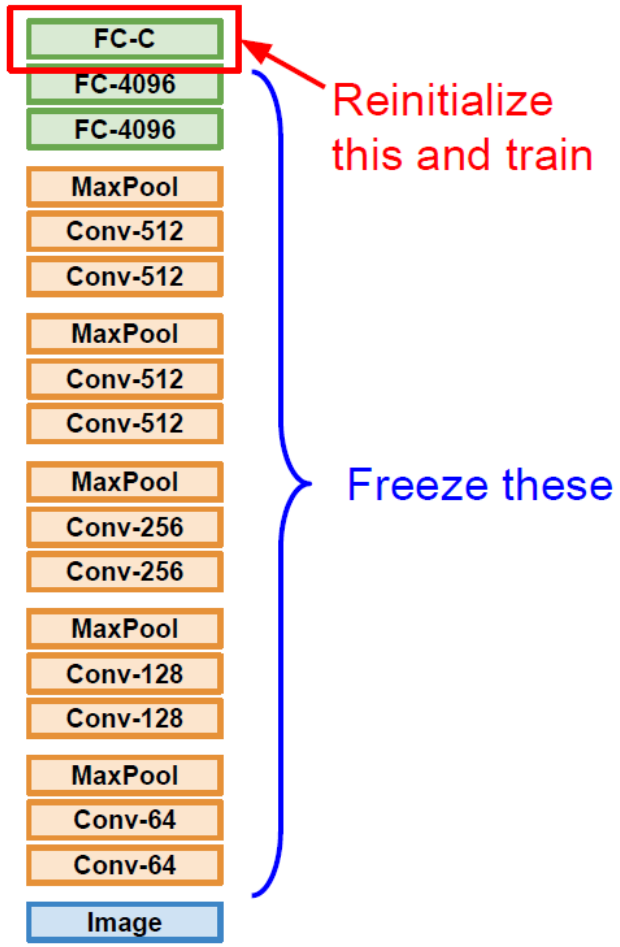
From: https://cloud.google.com/tpu/docs/inception-v3-advanced?hl=de

Transfer Learning

1. Train on large data set
(or even pre-trained net)



2. Fine-tune: Train the last layer(s) for c custom classes



Later layers act as good feature extractors

Freezing means not updating the weights (even if you would get gradients from backprop).

```
import torchvision.models as models
resnet18 = models.resnet18(pretrained=True)
alexnet = models.alexnet(pretrained=True)
squeezenet = models.squeezenet1_0(pretrained=True)
vgg16 = models.vgg16(pretrained=True)
densenet = models.densenet161(pretrained=True)
inception = models.inception_v3(pretrained=True)
googlenet = models.googlenet(pretrained=True)
shufflenet = models.shufflenet_v2_x1_0(pretrained=True)
mobilenet = models.mobilenet_v2(pretrained=True)
resnext50_32x4d = models.resnext50_32x4d(pretrained=True)
wide_resnet50_2 = models.wide_resnet50_2(pretrained=True)
mnasnet = models.mnasnet1_0(pretrained=True)
```

Transfer Learning - Usage

- How can we use these architectures?

```
tf.keras.applications.InceptionV3(
    include_top=True,
    weights="imagenet",
    input_tensor=None,
    input_shape=None,
    pooling=None,
    classes=1000,
    classifier_activation="softmax",
)
```

Usually, we use pretrained models and keep the weights

We only change the final layer
(„where the decision happens“)

Examples

Transfer Learning

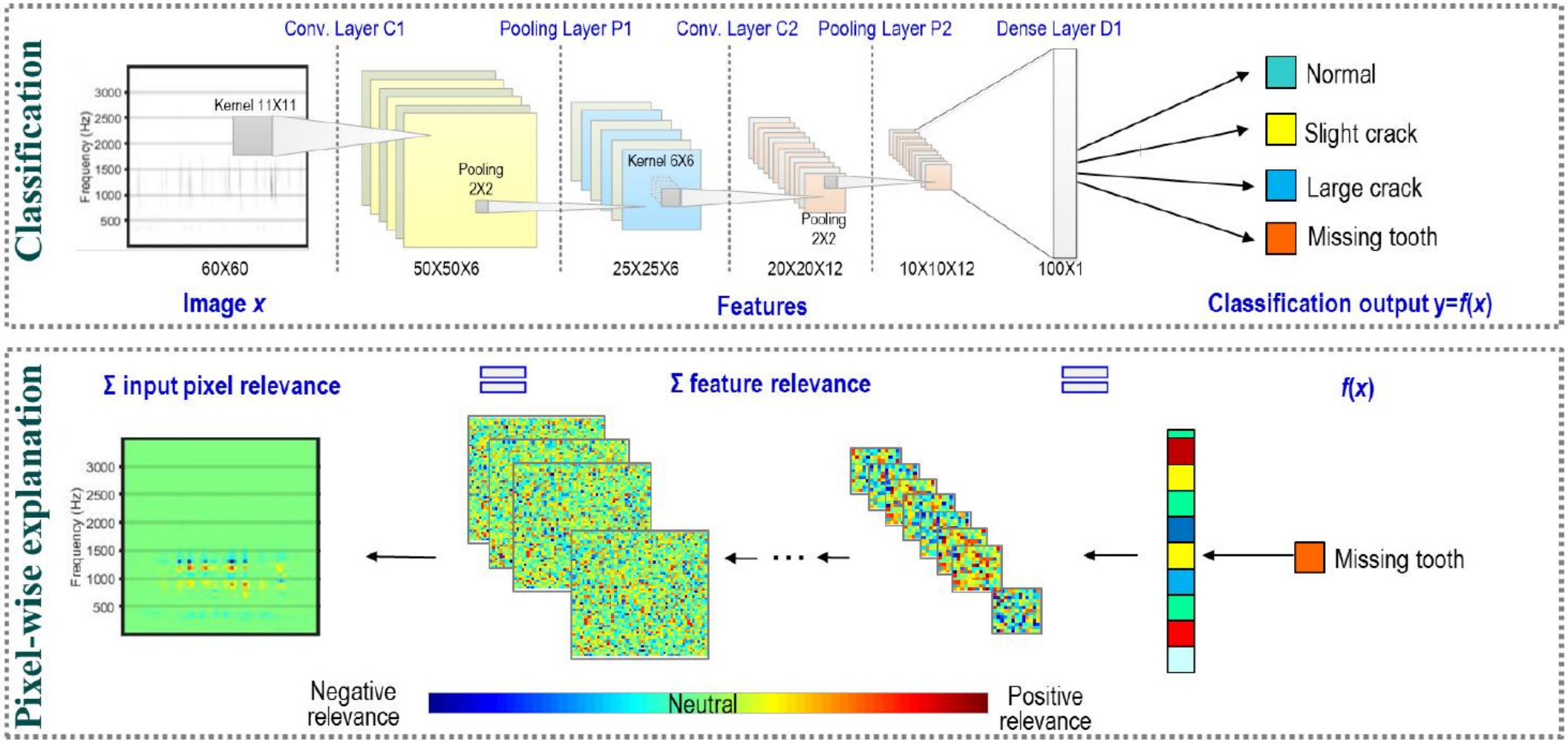
- Transfer Learning is not just in image classification a viable strategy

- Example: Classification of sensor values:
  - Usually: You design a model architecture specifically for [n_sensors, 1] sized inputs
  - It can be worth to try out a naive transfer learning approach
  - Try to resize your input into [x, y, 1] and act like it’s a pictures
  - Chances are high that this will work well

26th CIRP Life Cycle Engineering (LCE) Conference

Explainable Convolutional Neural Network for Gearbox Fault Diagnosis

John Grezmak [a], Peng Wang [a], Chuang Sun [b], and Robert X. Gao [a,*]



Highly Accurate Machine Fault Diagnosis Using Deep Transfer Learning

Siyu Shao, *Student Member, IEEE*, Stephen McAleer, Ruqiang Yan, *Senior Member, IEEE*, and Pierre Baldi, *Fellow, IEEE*

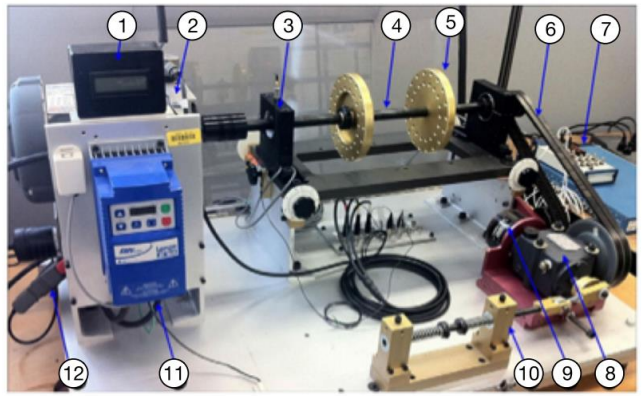
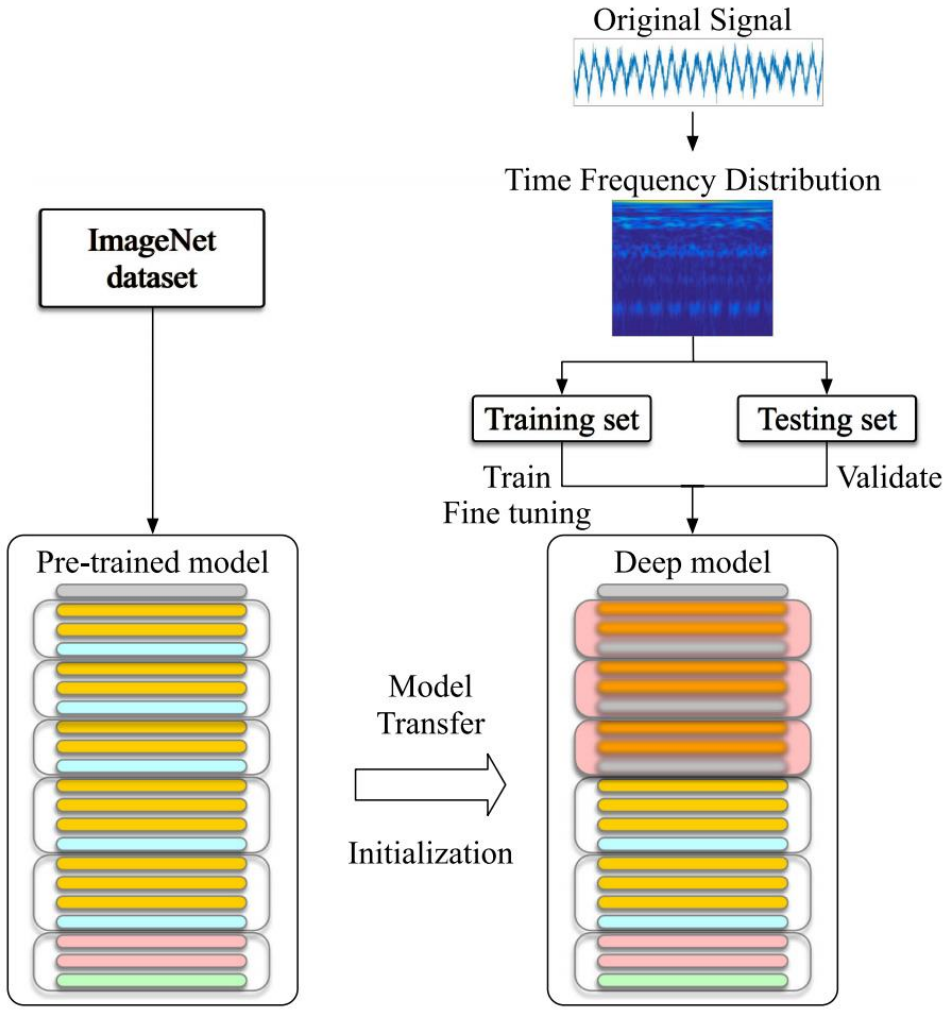
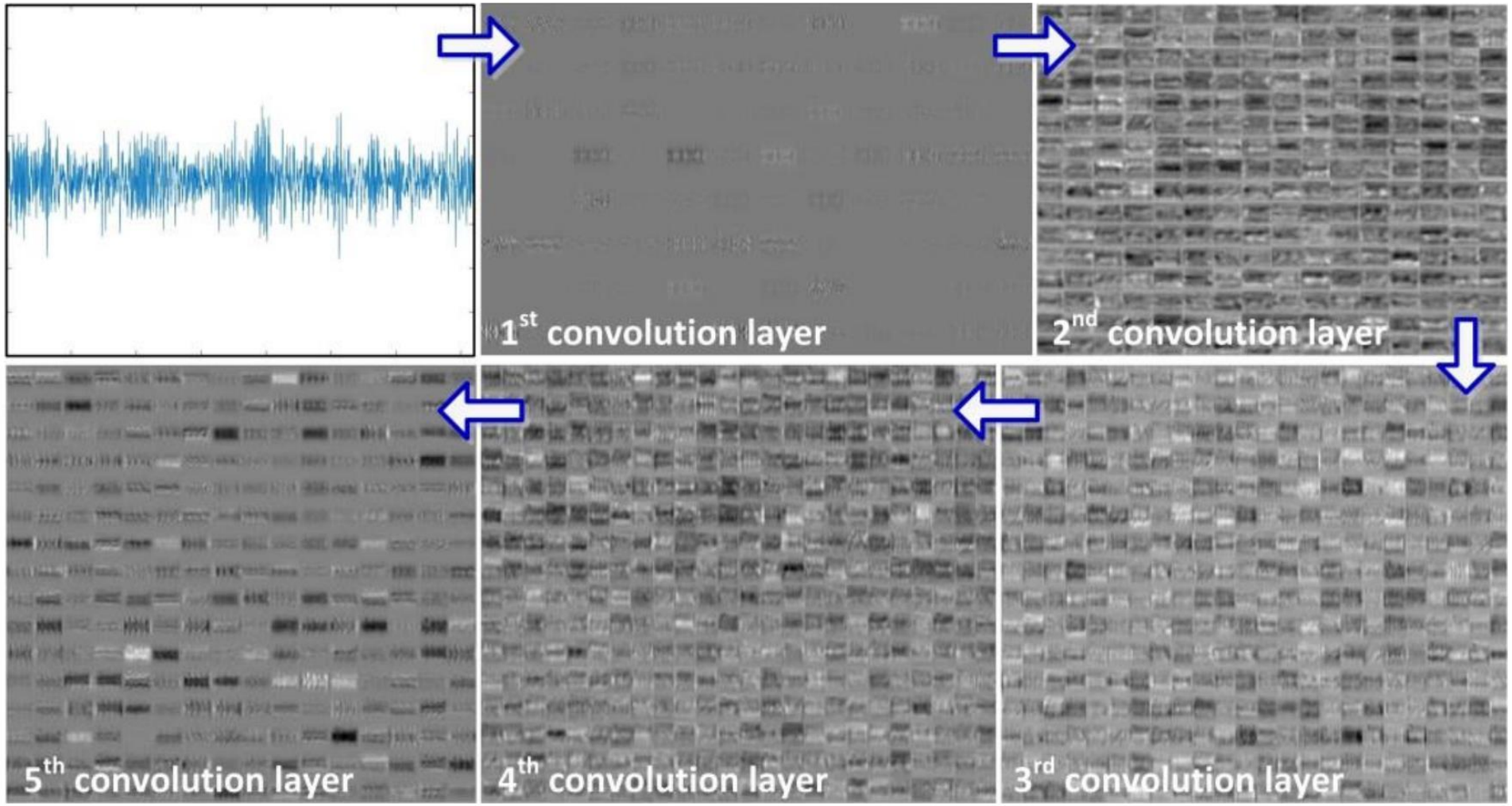


Fig. 4. Experimental facility. (1) Opera meter. (2) Induction Motor. (3) Bearing. (4) Shaft. (5) Loading Disc. (6) Driving Belt. (7) Data Acquisition Board. (8) Bevel gearbox. (9) Magnetic Load.(10) Reciprocating Mechanism. (11) Variable Speed Controller. (12) Current Probe [23].

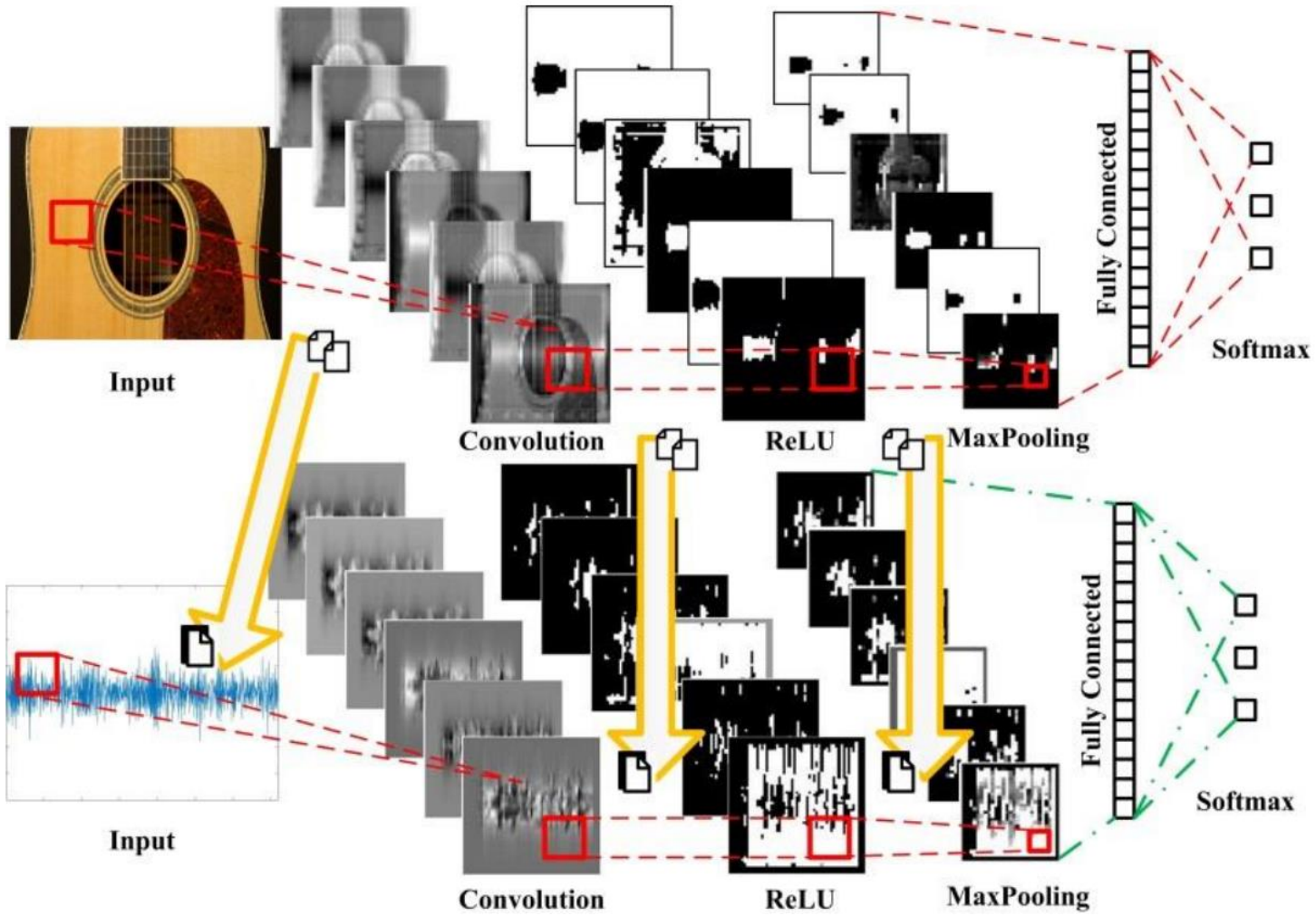


Examples: Crack detection



[Pre-Processing-Free Gear Fault Diagnosis Using Small Datasets with Deep Convolutional Neural Network-Based Transfer Learning, Cao et a., 2017]

Illustration of Transfer Learning for Gear Fault Diagnosis



[Pre-Processing-Free Gear Fault Diagnosis Using Small Datasets with Deep Convolutional Neural Network-Based Transfer Learning, Cao et a., 2017]

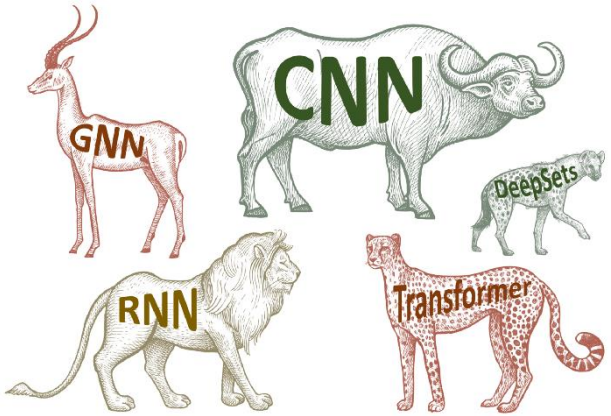
Transfer Learning – Wrap up

- Take away message: Most of the time, pretrained models are an excellent starting point for many (image) classification tasks

- If you are faced with a classification problem, you should try a transfer learning approach first
  - Look at "model zoos"

- If you only have a small amount of data, transfer learning is often your best chance!

- Even extremely naive approaches work more often than you might think




```python
import torchvision.models as models
resnet18 = models.resnet18(pretrained=True)
alexnet = models.alexnet(pretrained=True)
squeezenet = models.squeezenet1_0(pretrained=True)
vgg16 = models.vgg16(pretrained=True)
densenet = models.densenet161(pretrained=True)
inception = models.inception_v3(pretrained=True)
googlenet = models.googlenet(pretrained=True)
shufflenet = models.shufflenet_v2_x1_0(pretrained=True)
mobilenet = models.mobilenet_v2(pretrained=True)
resnext50_32x4d = models.resnext50_32x4d(pretrained=True)
wide_resnet50_2 = models.wide_resnet50_2(pretrained=True)
mnasnet = models.mnasnet1_0(pretrained=True)
```

https://towardsdatascience.com/geometric-foundations-of-deep-learning-94cdd45b451d