Can we use deep learning for a small dataset?

HLRS Summer School Trust and ML

Dr. Khatuna Kakhiani

26.7.- 28.7.2023, HLRS, Universität Stuttgart

Agenda - July 27, 2023

11:45 - 13:00 CEST

- Can we use deep learning for a small dataset? 1
  - Tutorial 1

11:45 - 13:00 CEST

- Can we use deep learning for a small dataset? 2
  - Tutorial 2: Image Classification with Convolutional Neural Networks (CNN)

Can we use Deep Learning for a small dataset?

Part 1

What is Deep Learning (DL) used for?

Increasingly use of DL

- Identify objects in images
- Transcribe speech into text
- Match news items
- Web search
- Content filtering on social networks
- Recommendations on e-commerce websites
- Microscopic Imaging; Medical Imaging
- Drug Discovery
- Search for new Functional Materials

- **AI-based content generators**
- **AI image generators, Chatbots**

Generating new data

Learning from Data

Machines Learn from Data

Data

- structured
- unstructured

Findable

Accessible

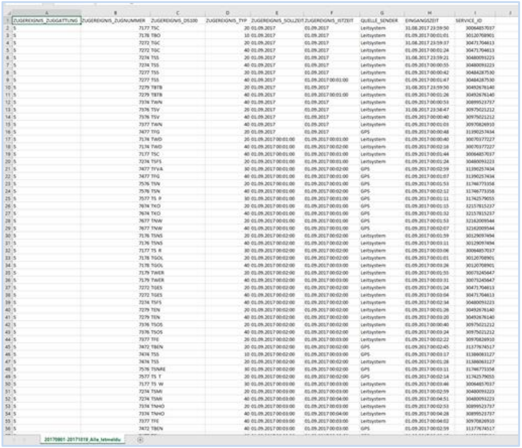
Interoperable

Reusable

- Collection of Raw Fact
- Unprocessed
- Recorded or stored

Data

- **Structured Data:**
  - organized and formatted
  - Examples: data held in spreadsheets and databases



- **Unstructured Data:**
  - no predefined construction or systemization
  - challenging to analyze
  - Examples: text, audio, images, videos & any combinations

Annotated Data

ML/DL models learn recurring patterns in **annotated data**

Data annotation:

- key part of the training dataset development for a supervised ML/DL
- adding metadata to a dataset - i.e., tags added

- Dataset for tutorial is annotated

Data

- TACO dataset- http://tacodataset.org/
- Contains images of waste in the wild
- User annotated - annotations in COCO* format
- Size: 1500 annotated images
- 60 categories of objects and 28 super categories

[*] https://cocodataset.org/#format-data for object detection

```
annotation{
    "id"                : int,
    "image_id"          : int,
    "category_id"       : int,
    "segmentation"      : RLE or [polygon],
    "area"              : float,
    "bbox"              : [x,y,width,height],
    "iscrowd"           : 0 or 1,
}
categories[{
    "id"                : int,
    "name"              : str,
    "supercategory"     : str,
}]
```

Learning Machines (LM)

The training of programs developed
by allowing a computer to learn
from its Experience,
rather than through manually coding the individual steps

Machine Learning (ML)

- Arthur Samuel (**1959**): “Field of study that gives computer ability to learn without being explicitly programmed”
- Tom Mitchell (**1998**): "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure **P**, if its performance at tasks in T, as measured by P, improves with experience E“

E = playing many games of chess

T = the task of playing chess

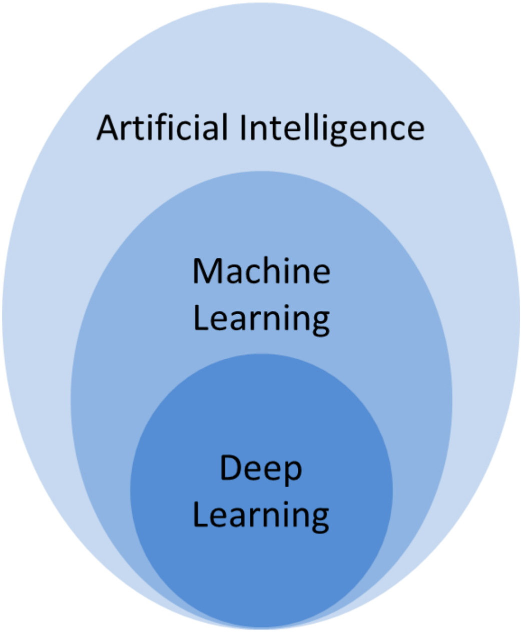
P = the probability that the program will win the next game

DL, ML and Artificial Intelligence

Techniques enable computer/machine to

- mimic human intelligence - Artificial Intelligence (AI)
- improve at task with experience (ML)
- train itself to perform a task (DL)

  (based on Deep Neural Networks (DNN))



The Learning Problems

- Supervised Learning
  - continues target variable
    - **Regression**
      - Prediction of age of a person based on a picture
  - categorical target variable
    - **Classification**
      - **Image classification**
      - Diagnostics
        - Microscopic imaging
        - Biomedical imaging
      - Data-Driven Computational Fluid Dynamics
      - Ocean/ Weather/Geosciences

- Unsupervised Learning
  - target variable not available
    - **Clustering**
      - Customer Segmentation
      - Astronomical data analysis
    - **Dimensionality reduction**

- Reinforcement Learning
  - categorical target variable
    - **Classification**
      - Optimized Marketing
  - target variable not available
    - **Control**
      - Driverless car

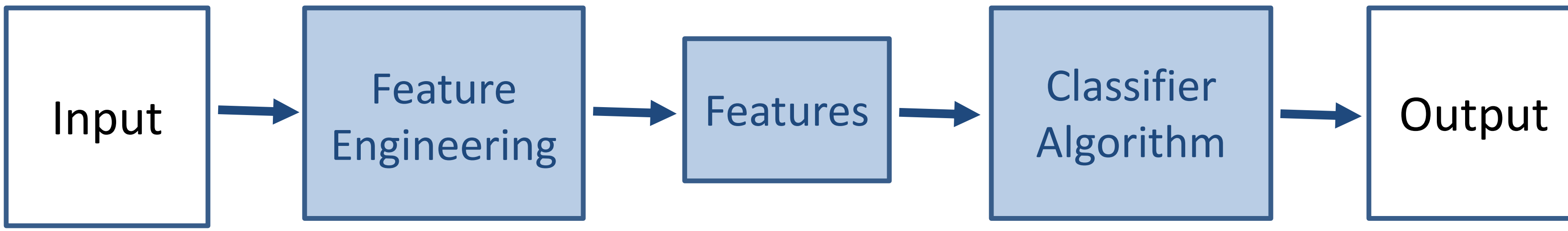
The Learning Problem

- **Supervised learning**
  - **Waste image classification**
- **Classification:** The goal is to predict discrete values
  - {Bottle, No bottle}
  - {1, 0}

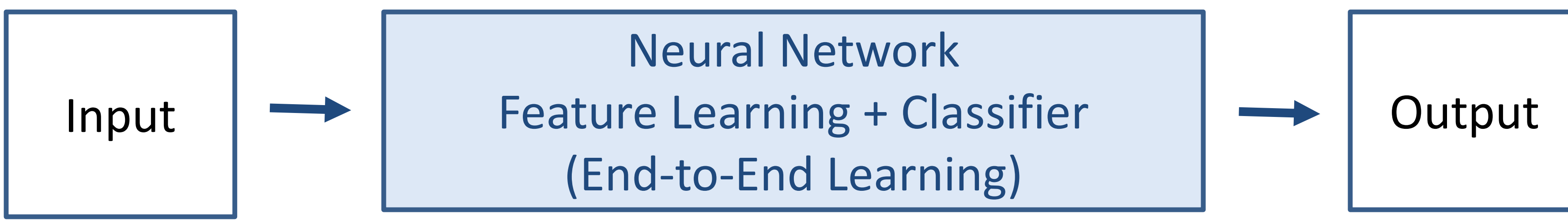
Why Deep Learning?

Simplified workflow

Classical Machine Learning



Deep Learning



Why Deep Learning?

- Works better on **unstructured data**
- **End-to-End training**
- **Model reuse**: learned features and trained models
  - transfer learning in case of a small dataset
- **Automatic Feature Extraction**
- **Eliminate domain-specific processing**
  - remove boundaries between CV, NLP etc.
  - unified set of tools for diverse problems
- Cons: Black-Box models

Data - Image

How do Computer Store Images?

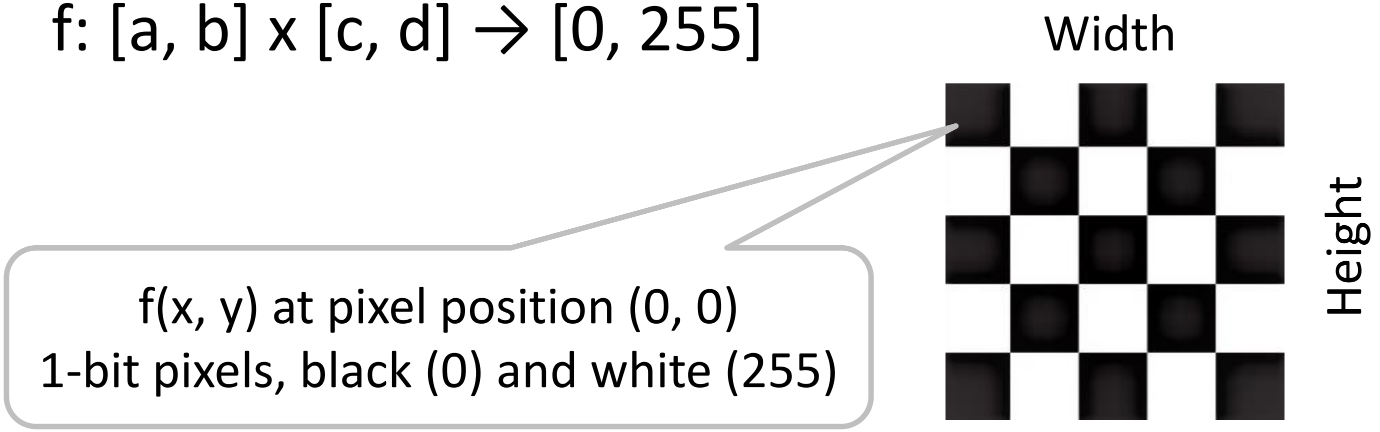
- **A matrices of numbers** representing **the discrete color** or **intensity values** present in every **image pixel**

- Every photograph, in digital form, is made up of pixels
- **Pixel** - smallest unit of information
- Each pixel is using a combination of Red, Green, Blue colors
- This is what we call an RGB image
- typically arranged in a 2-dimensional grid

Image color mode

A grayscale color mode

- An image, comprised of pixels, a function **f ?**
- Each 8 - bit pixel has its own value:
  i.e., intensity range [0, 255]; 0 - black & 255 - white
- f(x, y) → the color intensity at pixel position (x, y)
  assuming a rectangle with a finite range:
  f: [a, b] x [c, d] → [0, 255]



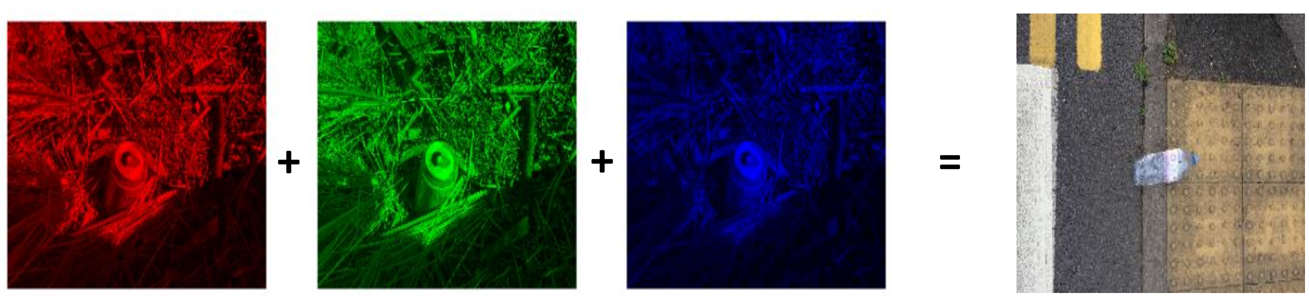
[

Image color mode

A RGB color mode

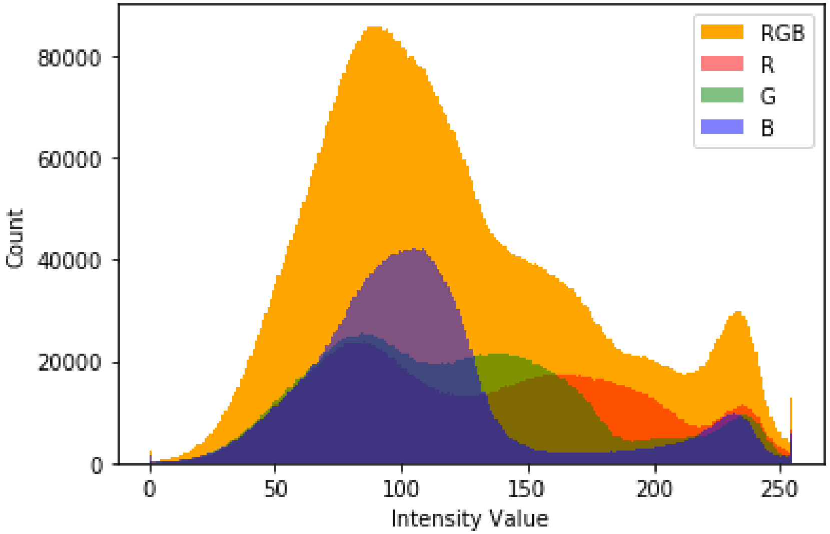
- RGB colors - a combination Red, Green, Blue channels
- Pixel range [0, 255] of each 3, 8 - bit color channel
- 256 * 256 * 256 = 16,777,216 combinations or color

R G B RGB



Histogram

- The distribution of pixel values of an image



- Image width x height: 2049 x 1537 (tutorial image)
- Color channel intensity values: Red [67 49 38 … 145 152 160]
  Green [64 46 35 … 119 126 134]
  Blue [71 53 42 … 96 103 109]

Tutorial

Tutorial - Image exploration

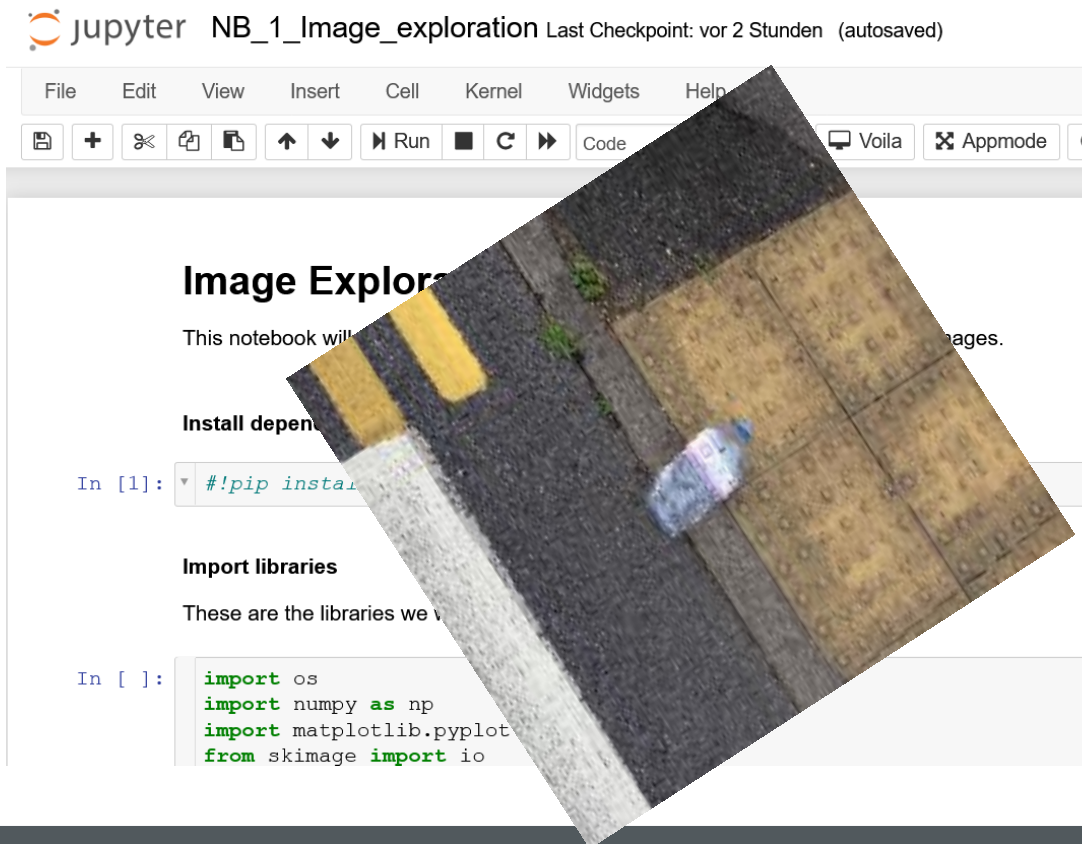
- **Hands-on:** Image exploration with Jupyter Notebook
- **Outcome:** Basic understanding of *image data,* ability to manipulate image instances and visualize results.

Tutorial

Jupyter notebook

10 min

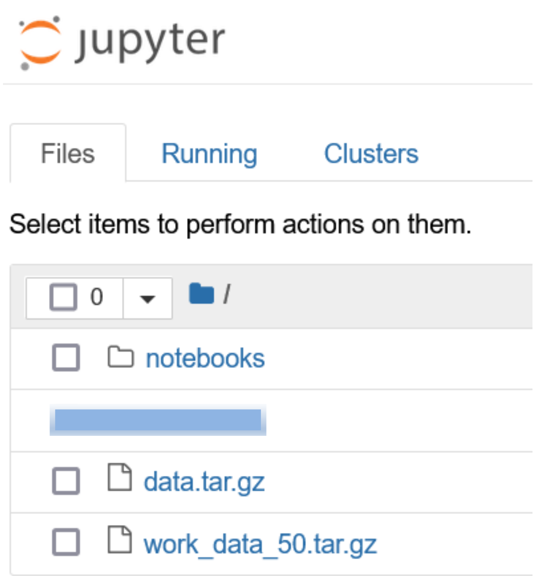
- Open a notebook notebooks/NB-Image_exploration.ipynb



Tutorial - Image exploration

- TACO dataset - http://tacodataset.org/
- Contains images of waste in the wild
- User annotated 1500 images

TACO dataset

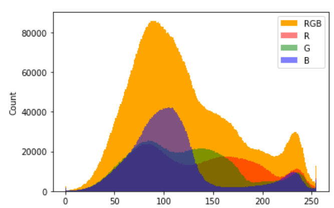


Tutorial - Image exploration

Tasks to complete in Notebook: NB-Image_exploration.ipynb

- Import libraries
- Load and handle one image data using PIL/Pillow
- Get original image size and color mode
- Split image from dataset into its RGB channels

- Explore Histogram of an Image



- Manipulate image instances

Tutorial - Image exploration

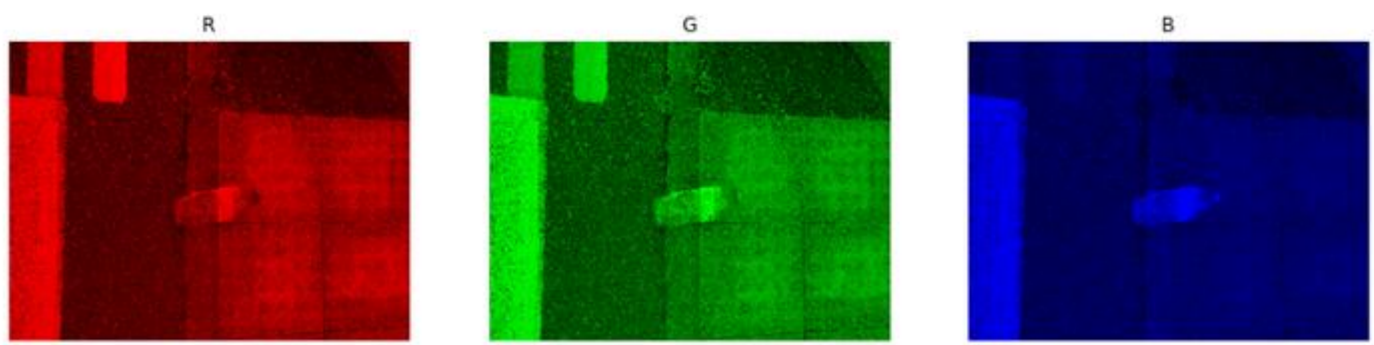
Example of a task:

```
In [ ]: #Exercise 5: display each color mode on a separat axes
fig, axes = plt.subplots(nrows=1, ncols=3, figsize=(15,5))

for v, c, ax in zip(["----", "G", "----"], range(3), axes):
    tmp_im = np.zeros(im.shape, dtype="uint8")
    tmp_im[:,:,c] = im[:,:,c]
    ax.imshow(tmp_im)
    ax.set_title(v)
    ax.set_axis_off()
```

to be completed

Expected result:



Deep Learning

Problem

- Huge amount of plastic waste
- Impact to wildlife and on humans
- Can we help build robots to classify waste automatically?

Source: https://www.onegreenplanet.org/

Data

- TACO dataset- http://tacodataset.org/
- Contains images of waste in the wild
- User annotated - annotations in COCO* format
- Size: 1500 annotated images
- 60 categories of objects and 28 super categories

[*] https://cocodataset.org/#format-data for object detection

```
annotation{
    "id"              : int,
    "image_id"        : int,
    "category_id"     : int,
    "segmentation"    : RLE or [polygon],
    "area"            : float,
    "bbox"            : [x,y,width,height],
    "iscrowd"         : 0 or 1,
}
categories[{
    "id"              : int,
    "name"            : str,
    "supercategory"   : str,
}]
```

Pop tab
Drink can
Single-use carrier bag
Plastic straw
Plastic Film
Paper cup
Plastic lid
Disposable plastic cup
Other plastic wrapper
Glass bottle
Plastic Film
Meal carton
Foam food container
Plastic drink bottle
Plastic bottle cap
Wrapping paper
Drink can

Learning from a small dataset?

- Are huge datasets for every kind of real-world problem feasible?
- What are Issues of small number of observations?
- How can one avoid overfitting and get accurate results?
- Are small datasets balanced?
- What are model’s limitations when drawing conclusions from its predictions?
- Is one aware about evaluation metrics for imbalanced data?

Image Classification Task

- **Given Dataset →** annotated RGB **images of litter**

- **The Task:**
  – predict a single label (or a distribution over labels to indicate confidence) for a given image

➢ turn numbers into a single label, such as *"bottle"*

The Learning Problem

- **Supervised learning**
  - **Waste image classification**
- **Classification:** The goal is to predict discrete values
  - {Bottle, No bottle}
  - {1, 0}

Image Classification Pipeline

- **Input:**
  - a set of **m** RGB **images of litter**
  - each labeled with one of **2 different classes {1, 0}**
- **Learning:**
  step of **training a classifier**, or **learning a model** using training set to learn what every one of the classes looks like
- **Evaluation:**
  classifier quality is evaluated by asking it to predict labels for a new set of images that it has never seen before

Input

- We distinguish 3 different sets of the **data**:

Train set	Validation set	Test set
• Model is trained • ~ 80% of the dataset	• Model is assessed • ~ 10% of the dataset	• Model is tested to make a prediction • ~ 10% of the dataset

- **The ground truth**: train, validation & test **label** sets

Learning a model

Configure the Model Architecture	Compile the model	Train the Model
• Artificial Neural Networks (ANN) • Multilayer Perceptron (MLP) • Convolutional Neural Networks (CNN)	• Loss (to measures how accurate the model is during training) • Optimizer (to minimize Loss with respect of parameters) • Metrics (to evaluate performance)	• Performance at Task improves with an Experience • Train to classify images • Track epochs, let model see every pictures many times; babysit process

Learning

- The **model train itself** from **experience E** with respect to some class of **tasks T** and performance **measure P**, if its performance at tasks in T, as measured by P, improves with experience E

E = perform a task multiple times (on train images & train labels)

T = the task to predict a single label for a given image

P = the probability that the trained model predict bottle

Evaluation

Evaluate

- Model performance is evaluated on validation set
- Trained model gives predictions on unseen data
- Chosen metrics suffice

Original Annotated Data

DL models learn recurring patterns in **annotated data**

- A set of 1500 images with annotations
- Original purpose: an object detection
- Annotations.json: a **nested dictionary** of data annotation

  Keys are mapped to another dictionary within original dictionary → preprocessing needed
- There are more annotations than images. Why?

Data Preprocessing

Create a simplified dictionary w.r.t. each image and its associated categories

```
Labels at instance {'file_name': 'batch_12/000019.jpg', 'category_ids':
{36, 5, 7, 45, 49, 22}, 'image_id': 320, 'height': 5312, 'width': 2988, '
category_names': {'Plastic bottle cap', 'Foam cup', 'Disposable food cont
ainer', 'Plastic film', 'Plastic utensils', 'Clear plastic bottle'}, 'sup
er_categories': {'Bottle', 'Bottle cap', 'Cup', 'Plastic container', 'Pla
stic bag & wrapper', 'Plastic utensils'}}
9.jpg', 'category_ids': {36, 5, 7, 45, 49, 22}, 'image_id': 320, 'height
```

Original

Data Preprocessing

- **Construct a binary classification problem**
  in a one vs all setting, i.e., does this image contain a specific supercategory (“bottle”) or not.

- **Split data**
  into three sets (**train**, **evaluation and test)** of **images and labels** to train, evaluate and test the model.

Data Preprocessing

- Summarize train, validation, and test data
- Notice the datasets are not balanced
  (e.g. proportion in supercategory < 50%)
- Choose evaluation metrics for imbalanced data

Training Number of instances: 1200, Proportion in supercategory: 19.50 %
Validation Number of instances: 150, Proportion in supercategory: 37.33 %
Test Number of instances: 150, Proportion in supercategory: 21.33 %

Data Preprocessing

Input parameters:

- Number of images
- Dataset images of varying sizes
- Number of RGB image channels (3)
- Pixel range [0, 255]

width x height in px: 2049 x 1537

Data Preprocessing

- Convert to gray scale:
  - color channel (3 → 1)
- Scale image size:
  - down-scaling of the width and height
  - uniform aspect ratio
  - most DNN models assume a square shape input image
- Normalize/standartize:
  - scale the pixel values to the range [0, 1]
  - each input parameter (pixel) has a similar data distribution
  - faster convergence while training the network

Deep Learning Tutorial

Tutorial - Data Preprocessing

skipped

- **Hands-on:** Dataset ex[...] preprocessing
- **Outcome:** Basic unders[...]ing of issues and challenges of image data preprocessing; ability to preprocess data for the image classifier training and evaluation.

Outlook

What can you do more?

Data Augmenting (DA)

Augment - artificially expand small labeled training datasets

- to prevent overfitting
- initial dataset is too small to train on
- to “squeeze” better performance from model

- Findable Accessible Interoperable Reusable

Data

DA Techniques

- **Geometric transformations**
  - Affine Transformations, simple linear transformations
  - shift, scale, rotate, flip, translate images…

- **Color space transformations**
  - change, intensify any color

- **Kernel filters** - sharpen or blur an image
- **Random Erasing** - delete a part of the initial image
- **Mixing images** - mix images with one another

Generative AI

- Involves training computers **to generate new data,** similar to data that it has been trained on, and to what a human might produce.
- Those algorithms can:
  - **improve the efficiency and accuracy of existing AI systems** (NLP, CV).
  - **produce various types of content, including text, imagery, audio and synthetic data**.

- **AI-based content generators**
- **AI image generators, Chatbots**
- Other useful tools that can make people's lives easier and grabbed the business headlines lately.

Tutorial

Requirements for the exercises

Deep Learning – part2

https://fs.hlrs.de/projects/par/events/2023/sst/commands-part2.txt

- Access your course account on Vulcan

  > ssh username@vulcan.hww.hlrs.de

- Recover **your** workspace

  > ws_list

  > MYSCR = $(ws_find wspart2)

  > cd $MYSCR (if **ws** path is stored) or

  > cd /lustre/nec/ws3/ws/username/**wspart2**

Requirements for the exercises

Deep Learning – part2

- Copy to your workspace:

  > cp -r /lustre/nec/ws3/ws/hpckkakh-exchangews/wsday2/***.tar.gz** ./

- In **your ws** directory unpack all ***.tar.gz** archives:
  > tar -zxvf data.tar.gz
  > tar -zxvf work_data_50.tar.gz
  > tar -zxvf notebooks.tar.gz

DONE !

Requirements for the exercises

- **data** - for Litter classification (*.json & co)
- **work_data_50** - image data as numpy array
- **notebooks** - tutorial exercises
- **solutions (you wont need hopefully)**

[*] http://tacodataset.org (dataset)

Requirements for the exercises

Options to work on exercises:

- **Jupyter Notebook**

and/or

- **Job scheduler** in terminal

Requirements for the exercises

To work with **Jupyter notebook**

- Open a **second** terminal window

- Connect Vulcan via **ssh** again:

  > ssh username@vulcan.hww.hlrs.de -q -D 8080

Requirements for the exercises

To work with **Jupyter notebook**

- Reserve computing time:

Max: until 18:00 CEST

> qsub -I -q R_sst -l select=1:node_type=hsw:**mem=100gb**,walltime=05:00:00

- Navigate to the workspace:

Special prompt of interactive batch job

s12345 n011101 000$ **ws_list**

> MYSCR = $(ws_find wspart2)

> cd $MYSCR (if **ws** path is stored/full path for wspart2)

(please check that you are in your workspace!)

Requirements for the exercises

To work with **Jupyter notebook** on **compute node &**

- Initialize your HPC working environment in your **ws**:

  > `. notebooks/modules.sh`

- To access the notebook, **copy URL**:

  > `http://n110010:8888/?token=...`

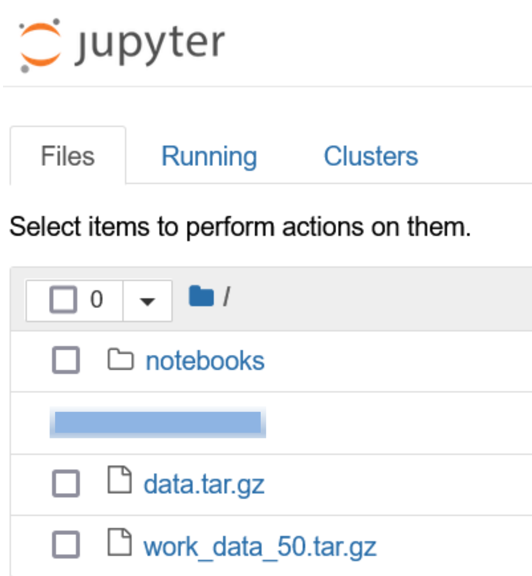
See next slide

Requirements for the exercises

To work with **Jupyter notebook**

- Launch browser **profiles** from the page “**about:profiles**”
- Choose **profile name** (i.e., Jupyter Notebook (JN))

”Launch profile in a new browser” or
”Profil zusätzlich ausführen”



- **Paste URL** in a JN browser profile:

http://n110010:8888/?token=...

You shall see in your browser

Requirements for the exercises

To work in **terminal** & HPC environment

> \> cd **notebooks**

- Submit job:

> \> **qsub job_test.pbs**

See next slide

- Check status:

> \> **qstat** # status of your job
>
> \> **qstat -q** # available queues

Requirements for the exercises

job_test.pbs

```
#!/bin/bash
#PBS -N  test
#PBS -q R_sst
#PBS -l  select=1:node_type=hsw
#PBS -l  walltime=00:03:00

cd $PBS_O_WORKDIR

# Load modules
module load python/3.10
…
type python
python  test.py  >  test.out
```

Directives

Loading software/API

User scripting