Hands-on AI 1

Unit 5 (Assignment) - Convolutional Neural Networks

Topics: Convolution, Linear, Activation, Subsampling

Goal:
Understand the basic building blocks of a "classic" deep learning image classification system and how it works.

Tasks:
This notebook, a week-related Youtube playlist, and a few required exercises and an optional exercise between us and you. Your assessment of this material, to make a decision on which exercise to do, is based on your understanding of the material and your ability to explain it to others.

TIPS TO USE THIS NOTEBOOK

The notebook is designed to run top-to-bottom. Some cells are based on data files stored in /content/data/ or /content/output/ or /content/drive/MyDrive/... If the cell fails, it might mean that the required data hasn't been created yet. Try running earlier cells in the notebook. If you're using Google Colab, you can mount your Google Drive and save/load data there if you want to keep it between sessions. Some cells are marked as optional, meaning they can be skipped but are not required for the assignment. Cells marked as "Exercise" are required exercises.

Suggestion: Before you start, make a copy of this notebook and rename it to **yourlastnamefirstinit_week5**. Example: **smithj_week5**

```
!mkdir -p /content/data
!mkdir -p /content/output
!mkdir -p /content/models
!mkdir -p /content/data/cifar10
!mkdir -p /content/data/mnist
!mkdir -p /content/data/fashion
!mkdir -p /content/data/cifar100
!mkdir -p /content/data/stanfordcars
!mkdir -p /content/data/imagenet
!mkdir -p /content/data/tinyimagenet
!ln -s /content/drive/MyDrive/ /content/drive
!ln -s /content/data /content/drive/MyDrive/colab_data
!ln -s /content/output /content/drive/MyDrive/colab_output
!ln -s /content/models /content/drive/MyDrive/colab_models
!ln -s /content/drive/MyDrive/Colab\ Notebooks /content/drive/MyDrive/colab_notebooks
!ln -s /content/drive/MyDrive/Colab\ Notebooks/utils /content/utils
```

```
!apt install libcairo2-dev pkg-config python3-dev && pip install pycairo
```

Exercise 1

Below is the data from the previous lecture, using the cell/code **(Tools) (Unit 5 Data)** to map the 3d coordinates and the camera parameters of the dataset of the image below into 2d pixel locations.

```
# (Tools) (Unit5 Data)
# Visualize the 3d points and camera locations
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
import numpy, pickle, matplotlib.pyplot, os, cv2, math, sys, io, PIL.Image, PIL.ImageDraw
from google.colab.patches import cv2_imshow
pickle_data = pickle.load(open('/content/data/tinyimagenet/0.pickle', 'rb'))
print(pickle_data.keys())
print(pickle_data['image'].shape)
matplotlib.pyplot.imshow(pickle_data['image'])
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['camera_intrinsics']
```

```
array([[ 1.        ,  0.        ,  0.        ],
       [ 0.        ,  1.        ,  0.        ],
       [ 0.        ,  0.        ,  1.        ]])
```

```
pickle_data['camera_extrinsics']
```

```
array([[ 0.        ,  0.        ,  1.        ,  0.        ],
       [ 0.        ,  1.        ,  0.        ,  0.        ],
       [-1.        ,  0.        ,  0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        ,  1.        ]])
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['point_cloud']
```

```
array([[ 0.        ,  0.        ,  1.        ,  0.        ],
       [ 0.        ,  1.        ,  0.        ,  0.        ],
       [-1.        ,  0.        ,  0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        ,  1.        ]])
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['point_cloud_color']
```

```
array([[ 0.        ,  0.        ,  1.        ,  0.        ],
       [ 0.        ,  1.        ,  0.        ,  0.        ],
       [-1.        ,  0.        ,  0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        ,  1.        ]])
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['image']
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['image_name']
```

```
'/content/drive/MyDrive/colab_data/tinyimagenet/train/n02481823/images/n02481823_0.JPEG'
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['image_size']
```

```
(512, 512)
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['bounding_box']
```

```
array([[ 0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
         0.        ,  0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        ,  0.        ,  1.        ,
         1.        ,  1.        ,  1.        ],
       [ 0.        ,  0.        ,  1.        ,  1.        ,  0.        ,
         0.        ,  1.        ,  1.        ],
       [ 1.        ,  1.        ,  1.        ,  1.        ,  1.        ,
         1.        ,  1.        ,  1.        ]])
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['camera_intrinsics'] @ pickle_data['camera_extrinsics'][:3,:4]
```

```
array([[ 0.        ,  0.        ,  1.        ,  0.        ],
       [ 0.        ,  1.        ,  0.        ,  0.        ],
       [-1.        ,  0.        ,  0.        ,  0.        ]])
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['camera_intrinsics'] @ pickle_data['camera_extrinsics'][:3,:4] @ pickle_data['point_cloud']
```

```
array([[ 0.        ,  0.        ,  1.        ,  0.        ],
       [ 0.        ,  1.        ,  0.        ,  0.        ],
       [-1.        ,  0.        ,  0.        ,  0.        ]])
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['camera_intrinsics'] @ pickle_data['camera_extrinsics'][:3,:4] @ pickle_data['point_cloud'] / pickle_data['point_cloud'][3,:]
```

```
array([[ 0.        ,  0.        ,  1.        ,  0.        ],
       [ 0.        ,  1.        ,  0.        ,  0.        ],
       [-1.        ,  0.        ,  0.        ,  0.        ]])
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['camera_intrinsics'] @ pickle_data['camera_extrinsics'][:3,:4] @ pickle_data['point_cloud'] / pickle_data['point_cloud'][3,:] * 100
```

```
array([[ 0.        ,  0.        ,  1.        ,  0.        ],
       [ 0.        ,  1.        ,  0.        ,  0.        ],
       [-1.        ,  0.        ,  0.        ,  0.        ]])
```

Starting Code/Output

```
# Visualize the 3d points and camera locations
```

```
# Create a 3d coordinate system, draw the camera in perspective and the point cloud from the image below, then
# save a new image to /content/output/3d_points_view.jpg (optional: if you want to use it in your report)
```

```
pickle_data['camera_intrinsics'] @ pickle_data['camera_extrinsics'][:3,:4] @ pickle_data['point_cloud'] / pickle_data['point_cloud'][3,:] * 100 + 256
```