

Lab3: Image classification

XXXXXXX AI for Digital Health (2025/2)



Objective

- **Create & train** image classification model from teachable machine and pytorch library
- **Hyperparameter tuning** in model
- **Inference & evaluate** performant each model



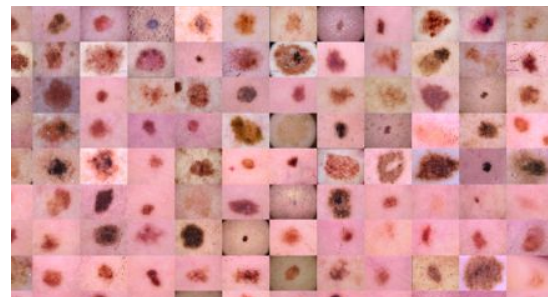
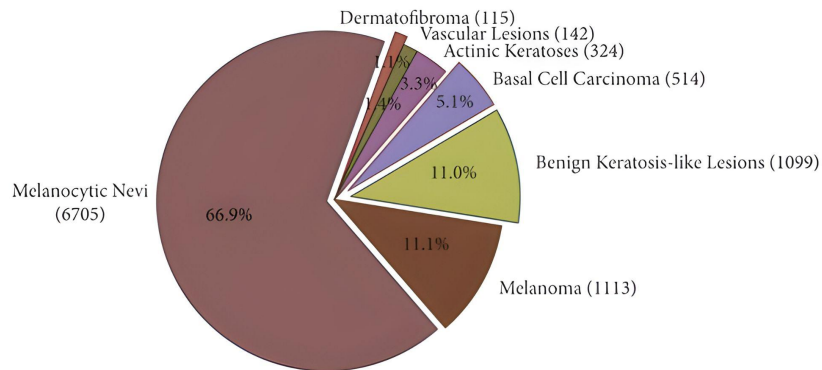
Material

- **Google Colab** is a free cloud-based platform that run jupyter notebook (Python code) in a browser, without needing local setup.
- **Teachable Machine** is a web-based tool by Google that trains machine learning models for images, audio, or poses without coding.
- **PyTorch, TensorFlow** is an deep learning library that enables building, training, and deploying neural networks using Python

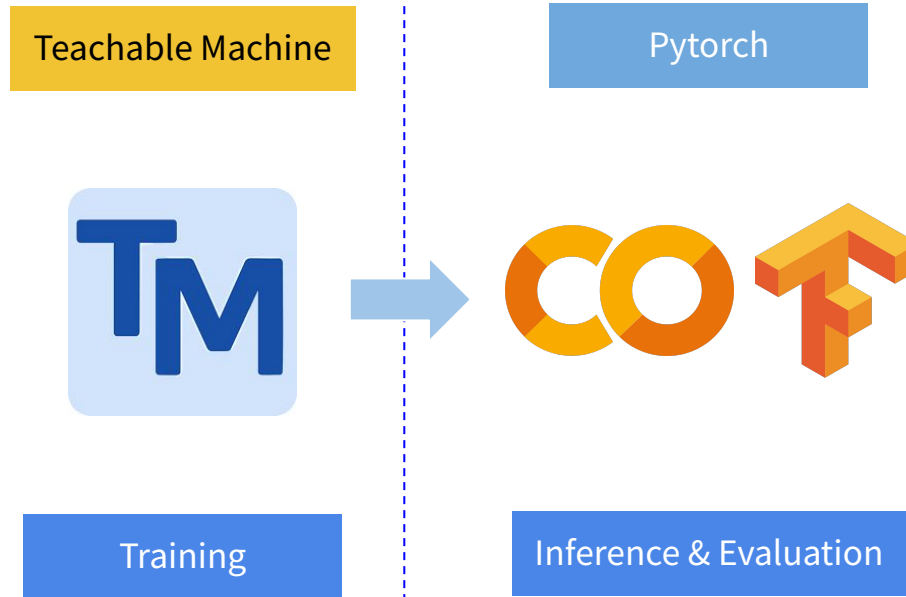


Dataset: Skin Cancer MNIST: HAM10000

- The dataset consists of 10015 images with 10013 labeled objects belonging to 7 skin cancer classes.
- The data contains image in JPG format and documents in JSON format
- In the experiment, we reduced the amount of data and formatted it to simplify the experiment.

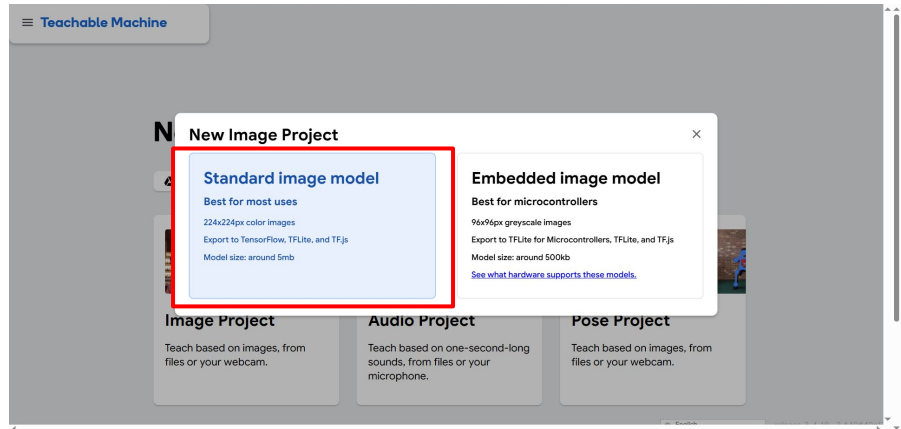
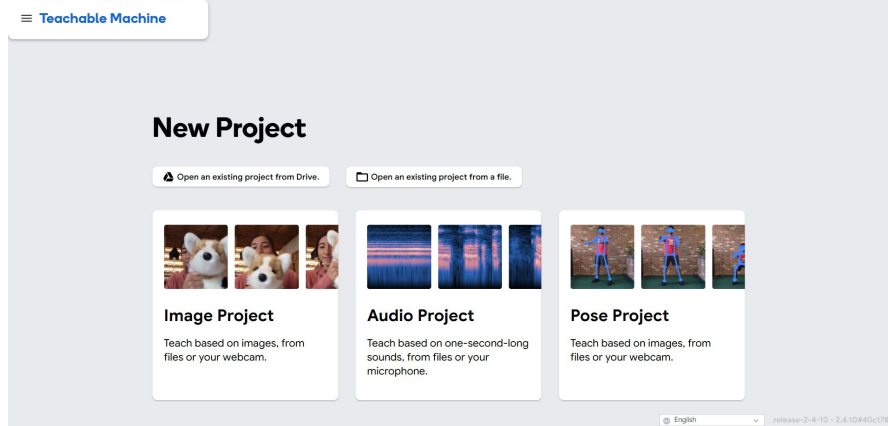


Lab3.1: MobileNet (Teachable Machine)



Lab3.1: MobileNet (Teachable Machine)

1) Create image project(standard image model) in Teachable Machine

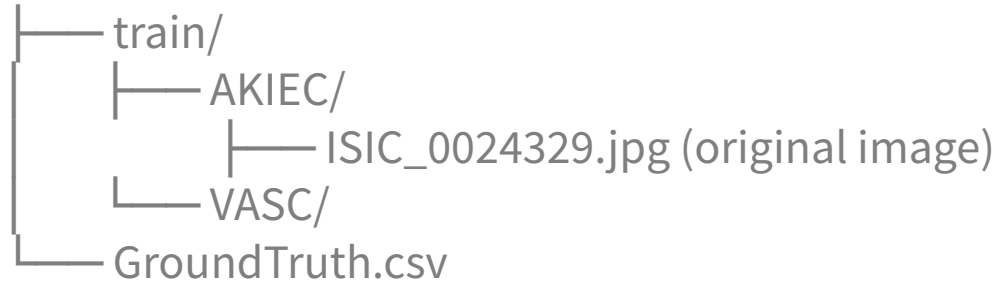


Lab3.1: MobileNet (Teachable Machine)

- 2) Add image sample and set label (Lab_3_1_HAM10000) into Teachable Machine

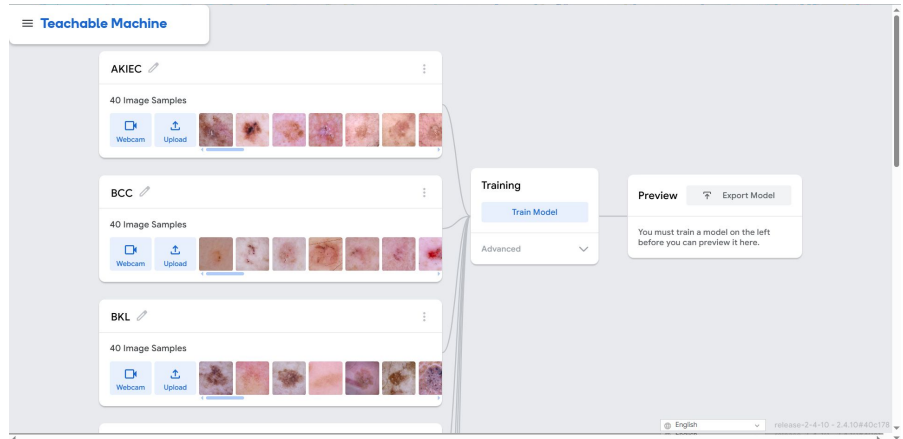
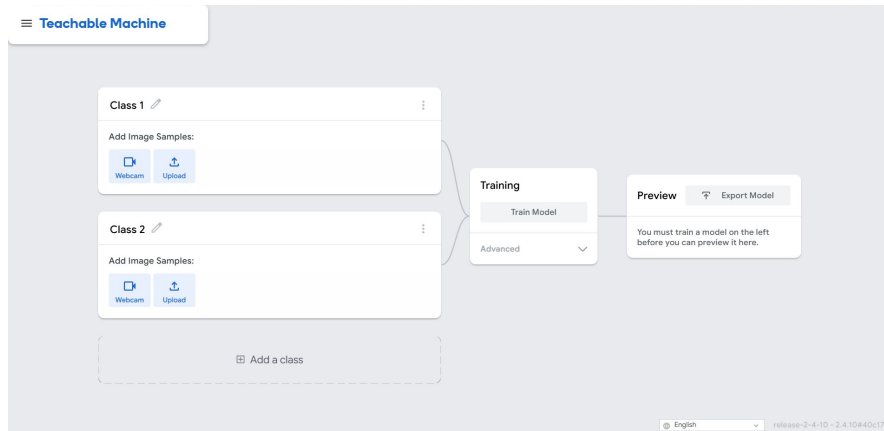
Lab_3_1_HAM10000 directory

Lab_3_1_HAM10000



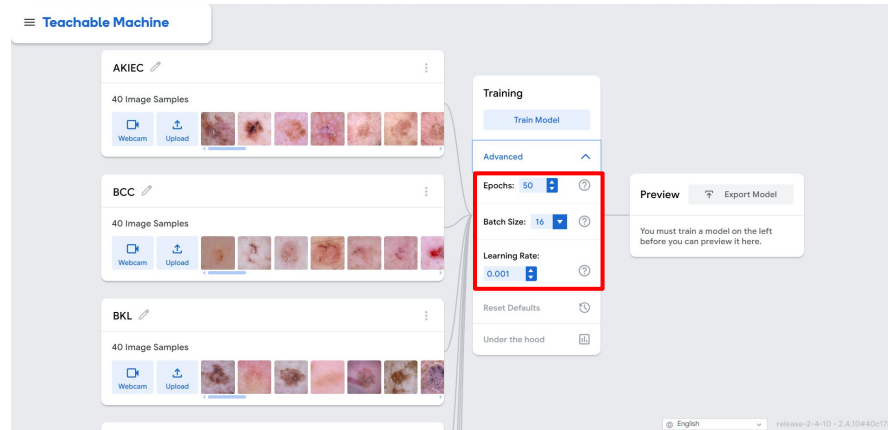
Lab3.1: MobileNet (Teachable Machine)

2) Add image sample and set label (code/Class03_IntroDL/Lab_3_1_HAM10000.zip) into Teachable Machine



Lab3.1: MobileNet (Teachable Machine)

- 3) Tune learning parameter & Train model
 - a) epoch: [5, 20, 50, 100]
 - b) learning rate: [0.00001, 0.001, 0.1]



Lab3.1: MobileNet (Teachable Machine)

4) Export model (tensorflow version)

The image consists of two side-by-side screenshots of the Teachable Machine web interface, illustrating the steps to export a trained MobileNet model as a TensorFlow version.

Left Screenshot: The main interface shows three datasets (AKIEC, BCC, BKL) on the left, training controls (Epochs: 50, Batch Size: 16, Learning Rate: 0.001) in the center, and a 'Preview' section on the right. In the 'Preview' section, the 'Export Model' button is highlighted with a red rectangle. Below it, an error message states: "There was an error opening your webcam. Make sure permissions are enabled or switch to image uploading."

Right Screenshot: A modal window titled "Export your model to use it in projects." is open. It shows three export options: "Tensorflow.js", "Tensorflow" (selected), and "Tensorflow Lite". Under the "Tensorflow" option, the "Download my model" button is highlighted with a red rectangle. Below the buttons, there is a section for "Code snippets to use your model:" with a "Keras" tab selected, displaying a Python code snippet for loading and using the model.

```
from keras.models import load_model # TensorFlow is required for Keras to work
from PIL import Image, ImageOps # Install pillow instead of PIL
import numpy as np

# Disable scientific notation for clarity
np.set_printoptions(suppress=True)

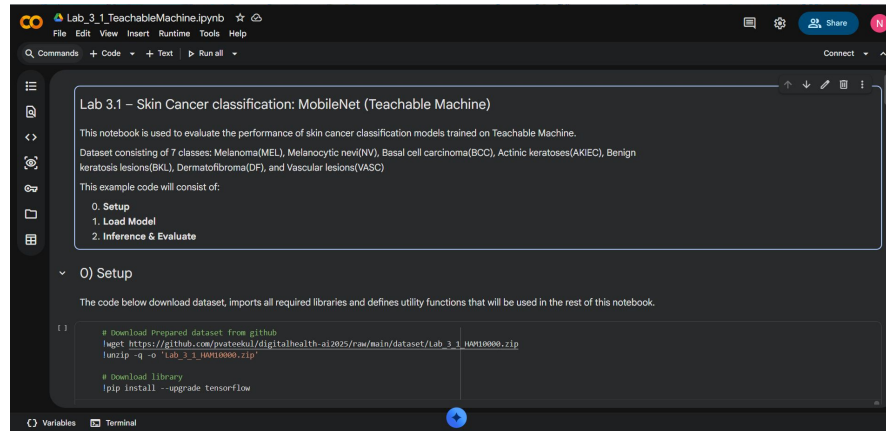
# Load the model
model = load_model('keras_Model.h5', compile=False)

# Load the labels
class_names = open('labels.txt', 'r').readlines()

# Create the array of the right shape to feed into the keras model
# The 'length' or number of images you can put into the array is
# determined by the first position in the shape tuple, in this case 1
data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)
```

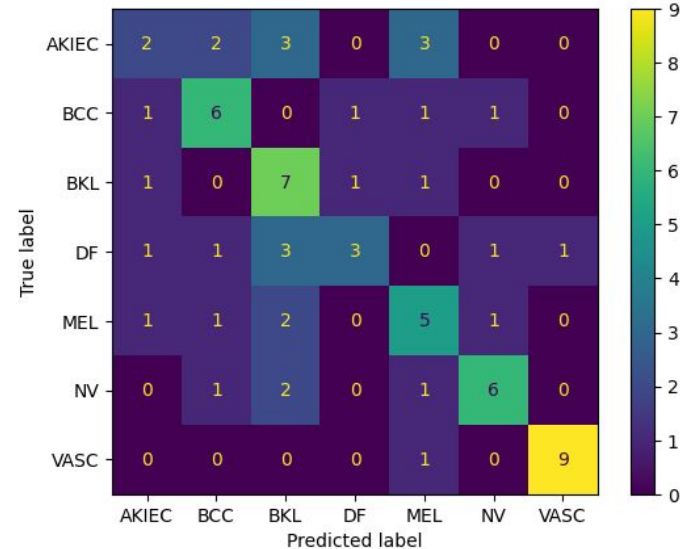
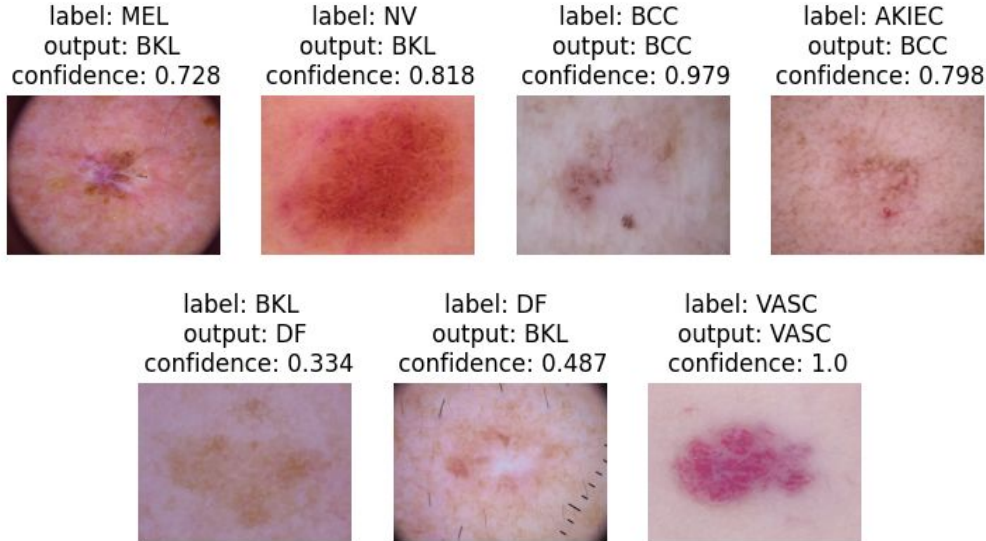
Lab3.1: MobileNet (Teachable Machine)

5) Open Lab_3_1_TeachableMachine.ipynb (in colab) and upload converted_keras.zip



Lab3.1: MobileNet (Teachable Machine)

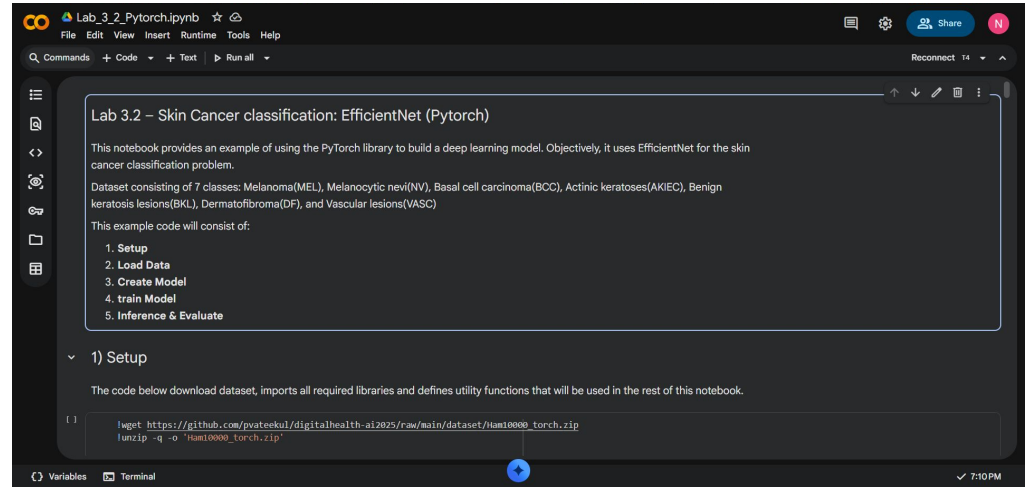
6) Run Lab_3_1_TeachableMachine.ipynb for evaluation & inference.



Lab3.2: EfficientNet (Pytorch)

Run Lab_3_2_PyTorch.ipynb (in colab)

- 1) Setup
- 2) Load Data
- 3) Create Model
- 4) train Model
- 5) Inference & Evaluate



The screenshot shows a Google Colab notebook interface. The title bar reads 'Lab_3_2_PyTorch.ipynb'. The left sidebar contains icons for file explorer, code editor, and runtime. The main area displays the notebook content, which includes a title 'Lab 3.2 – Skin Cancer classification: EfficientNet (Pytorch)', an introduction paragraph, a list of 7 classes, and a table of contents with 5 sections: 1. Setup, 2. Load Data, 3. Create Model, 4. train Model, and 5. Inference & Evaluate. The '1) Setup' section is expanded, showing a code cell with a command to download a dataset from GitHub and unzip it. The bottom of the interface shows 'Variables' and 'Terminal' tabs, and a status bar indicating '7:10 PM'.

Lab 3.2 – Skin Cancer classification: EfficientNet (Pytorch)

This notebook provides an example of using the PyTorch library to build a deep learning model. Objectively, it uses EfficientNet for the skin cancer classification problem.

Dataset consisting of 7 classes: Melanoma(MEL), Melanocytic nev(NV), Basal cell carcinoma(BCC), Actinic keratoses(AKIEC), Benign keratosis lesions(BKL), Dermatofibroma(DP), and Vascular lesions(VASC)

This example code will consist of:

1. Setup
2. Load Data
3. Create Model
4. train Model
5. Inference & Evaluate

1) Setup

The code below download dataset, imports all required libraries and defines utility functions that will be used in the rest of this notebook.

```
[ ] wget https://github.com/pvatekul/digitalhealth-ai2025/raw/main/dataset/Ham10000_torch.zip  
    unzip -q -o 'Ham10000_torch.zip'
```

