Topic: Machine learning II KIUA1006

Compulsory II: Title: *Deep Vision — Building and Analysing a CNN from Scratch and with Transfer Learning*

Duration: ≈ 5 hours total (divided into 2 parts)

Deadline: 30 October at 23.59

Goal:
Implement, train, and analyse a CNN for image classification, compare it with transfer learning using a pre-trained network, and explain the architectural and performance differences.

Main deliverables: A one PDF file that contains all sub-deliverables in each part (no more than 4 pages)

Learning Objectives

By the end of this assignment, you should be able to:

1. Understand and implement the structure of a CNN (convolutional, pooling, activation, and fully connected layers).
2. Train and evaluate CNNs on real-world image datasets.
3. Apply **transfer learning** using a pre-trained model (e.g., ResNet, VGG, MobileNet).
4. Analyse model performance, feature maps, and overfitting/regularization techniques.
5. Write a concise technical report explaining the architecture and results.

Part 1: Building a CNN from Scratch

Goal: Use the **CIFAR-10 (https://www.cs.toronto.edu/~kriz/cifar.html)** dataset (available in both PyTorch and TensorFlow).
It contains **60,000 color images (32×32 pixels)** across **10 classes**, such as airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

You may also use a **subset** (e.g., 2–5 classes) if your system is slow or you want faster experiments.

Tasks:

1. Load and preprocess the dataset (normalization, train/test split).
2. Build a CNN from scratch using PyTorch or TensorFlow/Keras
    - 3 convolutional layers with ReLU and max pooling

    - Dropout and batch normalization encouraged
3. Train the model for 10–15 epochs.
4. Record training and validation accuracy/loss.

Deliverables:

- Model architecture summary (code of the model)
- Accuracy and loss plots (graphs)
- Short commentary on underfitting/overfitting observations (text)
- Visualize some feature maps or activation outputs from your CNN's first layer (2-4 images)

Part 2 : Transfer Learning

Tasks:

1. Load a pre-trained model (ResNet18, VGG16, or MobileNetV2).
2. Freeze convolutional layers and train only the classifier head for the same dataset.
3. Optionally fine-tune the last convolutional block.
4. Compare accuracy and training time to your custom CNN (the initial CNN created in Part 1).

Deliverables:

- Description of the chosen pre-trained model (text and some figures if possible)
- Comparative results table (custom CNN vs pre-trained)
- Discussion: *Why does transfer learning perform better/worse? (text)*