Mini Projects

Marco Willi

Goal

The mini-project aims to apply the learned knowledge. Students should be able to model a specific problem in *Computer Vision* using Deep Learning methods. They should demonstrate their ability to implement, train, and evaluate models with PyTorch and handle GPU resources. Students should be able to present and defend their work.

Administrative Details

You will work on the mini-project in groups of two. Groups must be formed and reported by 11.12.2023.

You may choose any topic from the list below.

The submission must be made by 14.01.2024 (end of day).

The project must be presented in the last lesson on 15.01.2024. The presentation should last a maximum of 10 minutes plus a maximum of 5 minutes for questions. You should guide through a Jupyter notebook.

Submission

A completed Jupyter notebook covering the following content should be submitted:

- Title
- Introduction
- Data
- Models
- Results
- Discussion

You may also use *.py files for code. These must also be submitted. You can submit a link to a Git repository. In any case, a Jupyter notebook must be submitted/contained, which shows the structure mentioned above and is fully executed with visible cell outputs.

Google Colab

For this project, you will need GPU resources. Each student will receive 20 CHF from the program for Google Colab credits. You can purchase 2 x 100 compute units with the "Pay as you go" option:

https://colab.research.google.com/signup

You can get the reimbursement form from the lecturer in paper form. You can then collect the amount in cash at the program administration with the completed form and receipt.

Tips for collaborating via Google Colab can be found in the example repository.

You do not have to use Google Colab.

Evaluation

- Submission: 0 - 5 points
  - Notebook structure according to the specification
  - General code quality/readability
- Data/Background: 0 - 5 points
  - Description of the problem
  - Description of the data
  - Data preparation for modeling
- Modeling: 0 - 10 points
  - Models and architectures described
  - Models chosen sensibly
  - Models correctly implemented
  - Model selection: models compared sensibly
- Results: 0 - 5 points
  - Presentation of results
  - Discussion of results
- Presentation: 0 - 5 points

  - Content: structure and graphics
  - Precision of language
  - Answering questions

- Difficulty of the problem (Bonus): 0 - 3 points

There is a joint grade per group.

Example Repository

An example repository can be found here: Link. Note: This repository is bare-bones and shows an unfinished mini-project and is only intended to give an impression of what you should submit. It also contains tips on setting up Google Colab.

Topic List

You can choose one of the following topics:

Image Classification: EuroSAT Land Use and Land Cover Classification

Goal: Develop a model to classify satellite images. You should consider RGB images and images with 13 spectral bands (see wiki) in modeling. There are 10 classes and 27,000 images.

Approach: Develop CNNs to model the data. Investigate various architectures and decide what works best. Compare pre-trained models with those you train from scratch. Use appropriate data augmentation techniques. Since the dataset is relatively small, you should be careful of overfitting and robustly compare different models. Compare RGB-only models with models that use the full 13 spectral bands.

Dataset: The two datasets can be found here: https://github.com/phelber/eurosat. There is a dataset in RGB format and a dataset with 13 spectral bands.

Difficulty/Effort: Small - Medium

EuroSAT: Land Use and Land Cover Classification with Sentinel-2

Figure 1: Source: Link

Semantic Segmentation: Underwater Imagery

Goal: Develop a segmentation model to segment underwater camera images. You should classify pixels into 8 classes on 1,500 images.

Approach: Develop your architecture based on your intuition and knowledge from the course. Read the paper Islam et al. (2020) and implement one of the architectures presented (SUIM-Net RSB or SUIM-Net VGG) and compare with your architecture. Since the dataset is relatively small, you should be careful of overfitting and robustly compare different models.

Dataset: Available here: https://irvlab.cs.umn.edu/resources/suim-dataset.

Difficulty/Effort: Medium

Object Detection: from Ground Up

Goal: Implement object detection with an end-to-end deep learning approach.

Approach: Define a model type, such as YOLO (Vx), CenterNet, and implement it. Compare various hyperparameters.

Dataset: Use the Pascal VOC dataset for training.

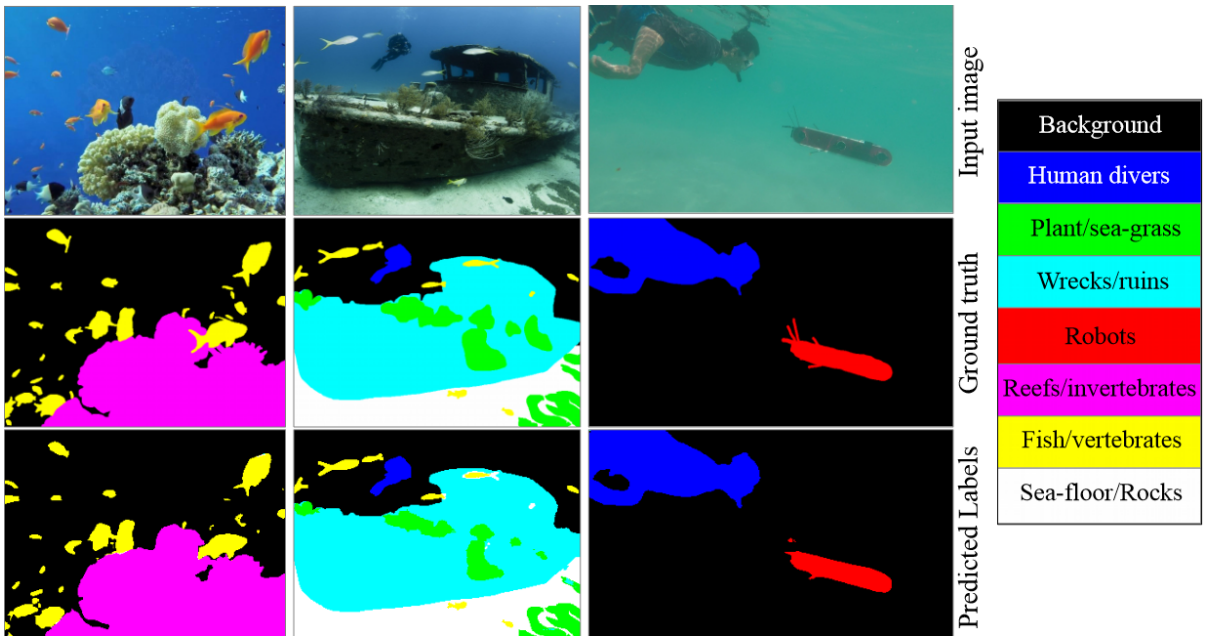

Figure 2: Source: Link

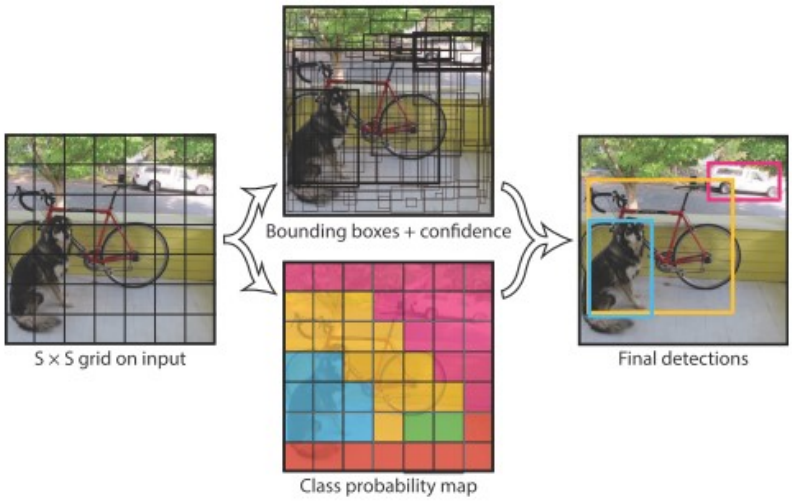

Figure 3: Source: Redmon et al. (2016)

Difficulty/Effort: Medium to Very High (depending on the choice of methods).

Image-to-Image: Steganography

Goal: Develop a model to hide image A in image B. Image B should remain as unchanged as possible, and image A should be extracted from image B as accurately as possible.

Approach: Define an encoder model that hides image A in image B. Define a decoder model that extracts image A from image B. Train the models end-to-end. You can be inspired by Baluja (2017).

Dataset: Use an image dataset and generate the training data yourself. You can use the COCO 2017 dataset.

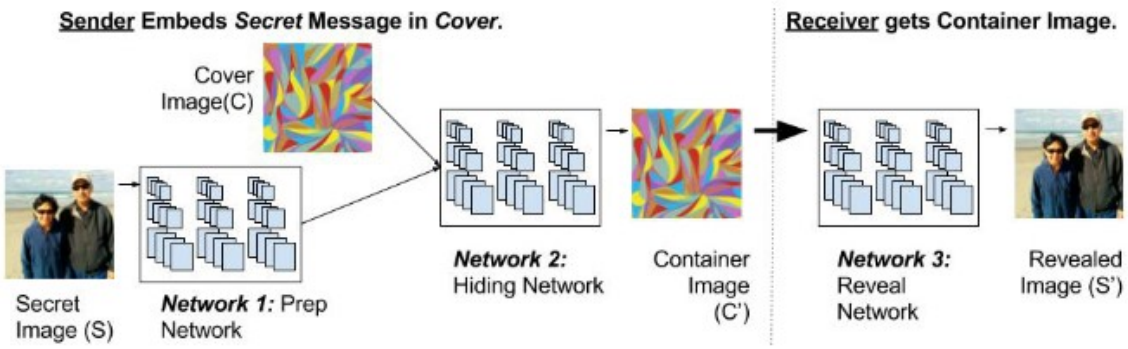

Figure 4: Source: Baluja (2017)

Difficulty/Effort: Medium - High.

Image-to-Image: Super Resolution on Faces

Goal: Develop a model that converts a low-res image of a face to a higher resolution.

Approach: Define an image-to-image architecture and train the model with appropriate data. You can assume fixed input and output resolutions. Implement the method from Dong et al. (2015). Test other, own variations of the architecture.

Dataset: Use images of faces. You can download up to 70,000 images from here: ffhq.

Difficulty/Effort: Small - Medium

Baluja, Shumeet. 2017. "Hiding Images in Plain Sight: Deep Steganography." *Advances in Neural Information Processing Systems* 2017-Decem (Nips): 2070–80. https://papers.nips.cc/paper_files/paper/2017/file/838e8afb1ca34354ac209f53d90c3a43-Paper.pdf.

Figure 5: Source: ffhq

Dong, Chao, Chen Change Loy, Kaiming He, and Xiaoou Tang. 2015. "Image Super-Resolution Using Deep Convolutional Networks." arXiv. http://arxiv.org/abs/1501.00092.

Islam, Md Jahidul, Chelsey Edge, Yuyang Xiao, Peigen Luo, Muntaqim Mehtaz, Christopher Morse, Sadman Sakib Enan, and Junaed Sattar. 2020. "Semantic Segmentation of Underwater Imagery: Dataset and Benchmark." arXiv. http://arxiv.org/abs/2004.01241.

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. "You Only Look Once: Unified, Real-Time Object Detection." *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 2016-Decem: 779–88. https://doi.org/10.1109/CVPR.2016.91.