University of Stirling - Spring 2023

ITNPAI1 - Deep Learning for Vision and NLP (2022/3)

Assignment Summary

In this activity, you are required to apply the knowledge acquired in this module through the design and development of a complete project for deep learning-based image pattern recognition in an application to be defined by yourself. For this, you will need to perform the following **mandatory** steps:

1. Problem definition
2. GitHub repository
3. Dataset
4. Dataloader
5. Proposed solution
6. Experimental tests and evaluations
7. Quiz and Report
8. Demonstration session

1. **Problem definition**

You must choose a computer vision task (classification, detection, semantic segmentation, captioning, geotagging, etc) to be modeled from images collected in the context of two different cities (A and B).

- If the work is being carried out in pairs, **cities A and B must be the hometowns of each student**. In the case of individual work, city A must be your hometown and city B must be Stirling (or Edinburgh, if needed).
- The standard project recommendation is to focus on recognizing cars or trees, which are easier to identify and annotate. Other objects or phenomena can be adopted, but are subject to prior approval by the module instructor (Jefersson A. dos Santos). **It is not allowed to assemble datasets containing people. Other sensitive patterns, such as license plates, must be properly hidden.**

- Don't panic! We are aware that acquiring images *in situ* is an impediment for most students. The dataset can be assembled with images collected remotely or from public repositories. Just be careful with rights and permissions for using images found on the internet. Anyway, these factors must be taken into account for the problem definition.
- Think of interesting problems, but that are easy to assemble an image dataset. Although we encourage you to do interesting and engaging work, it shouldn't be too complex or time-consuming. Try to appropriately scale the time required for this step. Ask the instructors for advice, if necessary.

top

---

2. **GitHub repository**

Give your project a name, create a private GitHub repository with the name [Module Code] + [Project Name] and give access to the module instructors. Create a cover page with a description of your project. This empty notebook must be uploaded in the repository as well as the created dataset. The checkpoint date to perform this task will be two weeks after the publication of this notebook. This notebook should be updated and committed to the repository according to the checkpoint dates. The repository's update history will be used as a criterion for monitoring and evaluating the work. **Check the videos provided in the extra section on Canvas for more details on how to create your GitHub repository.**

top

---

3. **Dataset creation**

You must collect a minimum of **200 positive samples** from the study objects for each city (A and B). Note that, depending on the task being solved, it will also be necessary to:

(i) collect more samples - negative ones, for instance;

(ii) annotating each image, delineating objects or creating bounding boxes. Planning and executing this correctly is important to ensure effective training of deep learning-based models.

Your dataset can be assembled from one or more of the following ways:

- *M1* - Pictures taken by yourself on site (street view from cities A and B), with attention to anonymization issues (if it is the case). It is not allowed to assemble datasets containing people. Other sensitive patterns, such as license plates, must be properly hidden.
- *M2* - Aerial satellite/drone images obtained from GIS and remote sensing platforms or public repositories. Be careful with unusual file formats that may be challenging to manipulate using

basic image processing libraries. We recommend keeping or converting the images to jpg or png.

- *M3* - Pictures taken from other public available datasets. Remember you are not allowed to use datasets containing people or other sensitive patterns/objects.
- *M4* - Images crawled from the internet as a whole (social networks, webpages, etc), with special attention to use and copyrights.
- *M5* - Textual and metadata you may need in your project, with special attention to use and copyrights (as always!).

Important: If you collect the images on your own or from aerial imagery repositories, it will be necessary to keep the geographic coordinates. If you collect from specific websites, please retain the source links. This information should be placed in a .csv file and made available along with the final dataset.

---

4. **Dataloader**

Here you are required to implement all the code related to pre-processing, cleaning, de-noising and preparing the input images and metadata according to the necessary data structures as input to your pattern recognition module. We recommend using PyTorch or Tensorflow (with Keras) as a base, but you are free to use any library or platform as long as it is well justified in the final report.

top

```
# Write your dataloader code here. Create more code cells if you find it necessary
```

---

5. **Proposed solution**

This is where you should implement most of the code for your solution. Write the routines for training and predicting the models and any necessary intermediate steps. Post-processing functions must also be implemented here.

- Use good programming practices, modularizing and adequately commenting on your code. Code quality will be considered in the final assessment. Again, we recommend using PyTorch, but you are free to use any library or platform. You just need to justify that in the final report.

- You can use pre-trained models as backbones or any code available on the web as a basis, but they must be correctly credited and referenced both in this notebook and in the final report. Cite the source link repository and explicitly cite the authors of it. If you changed existing code, make it clear what the changes were. Make it clear where your own code starts and where it ends. Note that the originality percentage of the code will be considered in the evaluation, so use external codes wisely and sparingly. **Missconduct alert:** remember that there are many tools that compare existing source code and that it is relatively easy to identify authorship. So, be careful and fair by always properly thanking the authors if you use external code.

top

```
# Write your proposed solution code here. Create more code cells if you find it necessary
```

---

6. **Experimental tests and evaluations**

Here you must implement your code for training, testing and evaluating your solution. For this, the following code blocks (*E1*, *E2*, and *E3*) are mandatory:

- *E1* - Training the models. Implement code to call the dataloaders implemented for training your models. Make routines to test different parameters of your models. Plot graphs that illustrate how parameters impact model training. Compare. Train and select a model for each city (A and B) and justify. You should use half (50%) of the samples from each dataset for training and leave the other half for testing (50%).

top

```
# Write your codes for E1 here. Create more code cells if needed
```

- *E2* - Testing the models in the dataset. You must implement code routines to test the predictive ability of your models using half of each dataset intended for testing. **The model trained in city A must be tested in city A. The model trained in city B must be tested in city B.** Use the evaluation metrics (accuracy, F1-score, AUC, etc) that are most appropriate for your problem. Plot graphs that illustrate the results obtained for each city (A and B). Plot visual examples of correctly (true positive) and incorrectly (false positive) classified samples.

top

```
# Write your codes for E2 here. Create more code cells if needed
```

- *E3* - Testing the models crossing datasets. Here you must do exactly the same as in *E2*, but now training in one city and testing in the other. **The model trained in city A must be tested in city B. The model trained in city B must be tested in city A.** Use the same metrics and plot the same types of graphs so that results are comparable.

top

```
# Write your codes for E3 here. Create more code cells if needed
```

---

7. **Quiz and Report**

Answer the assessment quiz that will be made available on Canvas one week before the final deadline. Make a 2-page latex report using the IEEE template with a maximum of 1000 words. You can deliver the report in MS Word if you prefer. Your report should contain five sections: introduction, description of the proposed solution with justifications, results (here you can include the same graphs and pictures generated in this jupyter notebook), discussion of the results, and conclusion. Properly cite references to articles, tutorials, and sources used. A pdf version of your report should be made available in the project's github repository under the name "[project name] + _final_report.pdf".

top

---

8. **Demonstration**

Some projects (around 10%) will be selected for a mandatory demonstration. During the demo, you will be asked about implementation details and decisions that led to the design of the developed solution.

top