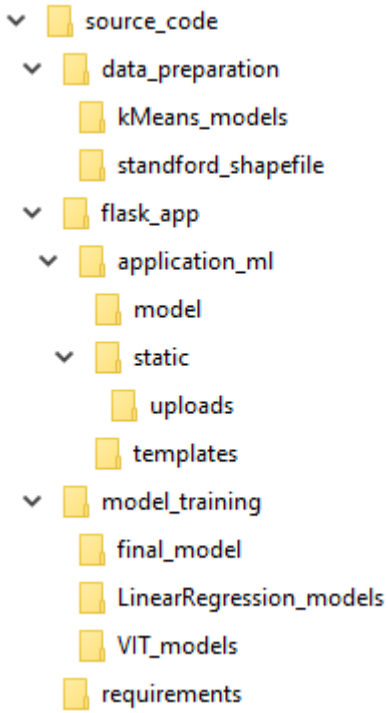
LUMEN DATA SCIENCE 2022
Technical documentation

GeoGuesser AI Agent

Zagreb, May 2022

1. Source code directory structure

Our main directory that contains all important source code is called `source_code` and contains three main subdirectories: `data_preparation`, `model_training` and `flask_app`.



Inside `data_preparation` directory is all source code for initial data analysis and preparation as well as files needed for running it. It's contents include:

- `kMeans_models` - a directory with the k-means models we used for finding regions
- `standford_shapefile` - a directory with Croatia's shapefiles
- `Data_analysis_and_preparation.ipynb` - notebook used for data analysis and preparation
- `*.csv files`, `HR_poly.pkl` (Croatia's area in polygons), `outliers.txt` (a list of removed outliers)

Directory `model_training` contains source code for all models' training and evaluation, the final model pipline and directories that have saved models:

- `final_model` - directory contains all 9 final models stored in .pth format
- `LinearRegression_models` - directory contains second layer linear regression models stored in .h5 format
- `VIT_models` - directory contains vision transformer first layer model stored in .h5 format
- `.py` files for training and evaluation
- `final_model_pipeline.ipynb`, `model_performance_analysis.ipynb` - notebooks for our model pipeline and it's performance analysis
- `test_predictions_weighted_average.csv` - predictions on test dataset (Photomath)

Directory `flask_app` contains the Flask app developed in Python.
Directory `requirements` contains `.whl` files for installing certain required libraries.

2. Jupyter Notebook

To run our `.ipynb` notebooks it is required to have **Jupyter**[1] software and **Python** installed. Python version used in the implementation is 3.8.2.

Jupyter Notebook can be started from the terminal by positioning yourself to desired directory and write `jupyter notebook`. Localhost will be started in default browser. There you can select the notebook you wish to view and/or run. To run the cell you can press the play button or keyboard shortcut *CRTL + Enter*.

It is important to note that some notebooks (`source_code/model_training/final_model_pipeline.ipynb`) were run on **Kaggle**[2] in order to reduce execution

[1] https://jupyter.org/
[2] https://www.kaggle.com/

time (and later downloaded) so their outputs cannot be seen.

3. Python

For Python files and notebooks it is required to have Python 3 (we used version 3.8.2.[1]) and pip[2] package installer.

3.1. Libraries

Required libraries to run Python code are specified in `source_code/requirements.txt` file. To install required libraries it is recommended to create a virtual environment. Virtual environments are a common and effective technique used in Python development. Each environment can use different versions of package dependencies and Python. To create a virtual environment follow these steps:

1. Using the terminal position yourself inside the `source_code` directory
2. `pip install virtualenv` (Windows) or `python3 -m pip install -user virtualenv` (Unix/macOS) to install virtual environment
3. `python3` (Unix/MacOS)/`py` (Windows) `-m venv env` to create a virtual environment
4. `env\Scripts\activate` (Windows) or `. env/bin/activate` (Unix/MacOS) to activate the virtual environment
5. In the activated enviroment run `pip install -r requirements.txt` to install required libraries

[1] https://www.python.org/downloads/
[2] https://pip.pypa.io/en/stable/installation/

4. Model training

Python scripts for model training and evaluation are located in `source_code/model_training`. To run the script position yourself inside the mentioned directory and run:

-> for training:

```
python <name_of_training_script>.py --data_dir "<path_to_dataset_directory>"
[ --saved_model "<model_name>.h5" --model_dir "<path_to_model_load_directory>"
--poly_dir "<path_to_data_prep_directory>" --model_s_dir
"<path_to_model_save_directory>" --class_label "<class_label_number>" ]
```

-> for evaluation:

```
python <name_of_evaluation_script>.py --data_dir "<path_to_dataset_directory>"
--saved_model "<model_name>.h5" [ --model_dir "<path_to_model_load_directory>"
--poly_dir "<path_to_data_prep_directory>" --model_s_dir
"<path_to_model_save_directory>" --class_label "<class_label_number>
--evaluate_on "test|val|train" --mode "separate|group" ]
```

where

- `saved_model` is used to continue training on last saved model - import saved model
- `model_dir` is used to set the directory to load model from
- `data_dir` is used to set the data directory
- `poly_dir` is used to set the data preparation directory containing csv files
- `model_s_dir` is only for training and used to set the directory to store model to
- `class_label` is only for second layer models used to filter csv file based on label -> args: 0,1,2,3,4,5,6,7
- `evaluate_on` is used only for evaluation and is used on train, val or test data -> arguments: test, val, train

- `mode` is used only for evaluation and defines how will the class be calculated -> arguments: separate, group

If you run within `source_code/model_training`:

- For scripts `GeoGuess_pytorch_8_classes_single_images*.py` and `VIT_pytorch_8_classes_single_images*.py` the only mandatory argument is `data_dir` because the dataset with images is not in the zip (because of large size).
- For scripts `GeoGuess_pytorch_8_subclasses_single_images*.py` and `LinearRegression_8_subclasses_single_image*.py` argument `class_label` must be provide alongside `data_dir`.
- If you are running `*evaluation.py` scripts you have to provide arguments `data_dir`, `saved_model`.
- All other arguments are optional.

If you want to continue training on our models they are in their respective directories: `final_model, VIT_models, LinearRegression_models`.

5. Model evaluation and test set prediction

The notebook which is used to evaluate our model and for prediction on real test dataset provided by Photomath is `source_code/model_training/final_model_pipeline.ipynb`. The two parts of the notebook dedicated to those tasks are:

1. Model's evaluation process on test dataset
2. Model's prediction pipeline on real test dataset (Photomath)

To run this notebook we used Kaggle since our personal computers do not have strong computing power. The dataset with images is not in the zip so the `DATADIR`(1.) / `TESTDIR` (2.) variable in cells marked with *#editing needed* will have to be set in order to run the notebook properly. It is important that cells are executed in their

order.

6. Flask app

To showcase our models prediction performance, we developed a REST application in Flask. To run the application:

1. Start a virtual environment (as described in 3.1.) inside the `source_code/flask_app` directory and install requirements specified in `source_code/flask_app/application_ml/requirements.txt` file
2. Position yourself to `source_code/flask_app/application_ml` and run command `python main.py`
3. Follow the link in the output or open your browser to `http://127.0.0.1:5000/` and you will see the homepage as on Figure 6.1
4. Now you can upload one or multiple images, click submit and you get the prediction as seen on Figure 6.2

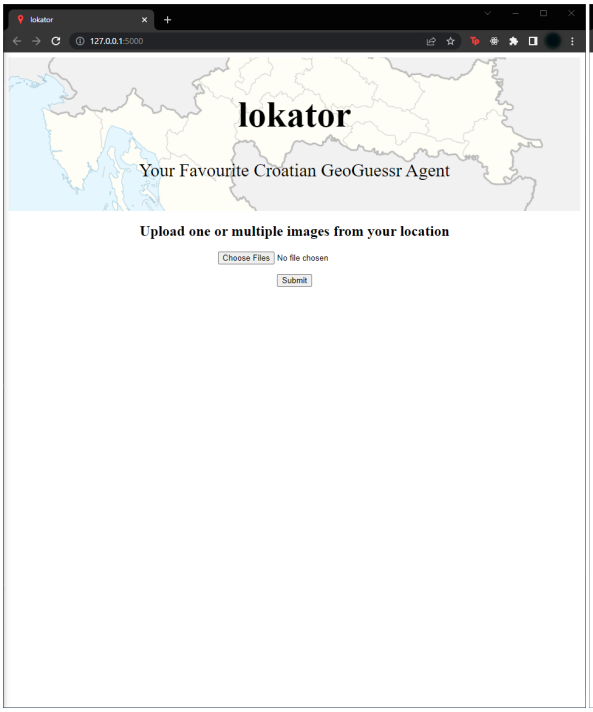


Figure 6.1: App homepage

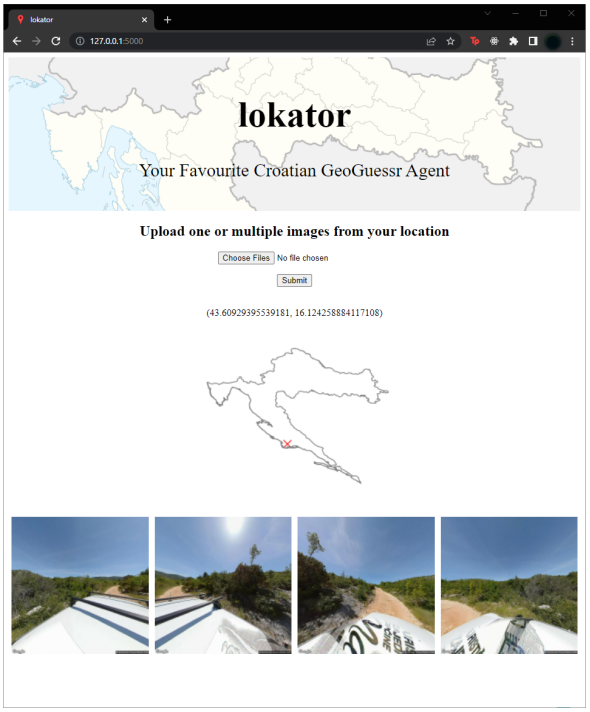


Figure 6.2: App prediction

Apart from the web interface it is possible to send curl requests with arbitrary number of images to get a prediction:

```
curl http://127.0.0.1:5000/predict -F "files[]=@<path_to_image1>"
[ -F "files[]=@<path_to_image2>" ...]
```