Final Project – 52025 – Using Aerial Imagery for forecasting indicators

Asaf Nissan & Eldad Riklin; HUJI 09\2024

link for full git

Abstract: We processed Sentinel-2 satellite imagery, aligning it to specific areas of interest using metadata and shapefiles. The preprocessed data was then used to develop a predictive model for socio-economic indicators, leveraging a modified ResNet-50 architecture and Grad-CAM for interpretability. Despite these advanced techniques, the model's performance was suboptimal, indicating the need for further improvements in feature selection or model design.

Data collection

Data Extraction of Aerial Imagery

we processed Sentinel-2 satellite imagery obtained from the Copernicus Open Access Hub, which provides 10-meter resolution data with four channels (Red, Green, Blue, and Near-Infrared). This choice of imagery was driven by its availability and suitability for land cover analysis, despite the existence of higher resolution datasets (e.g., 2-meter imagery) that only offer three spectral bands, which limits their utility for certain applications (As seen in Fig. 2, the Near-Infrared (NIR) channel highlights vegetation areas, rendering them in blue, which allows us to easily distinguish regions rich in plant life).

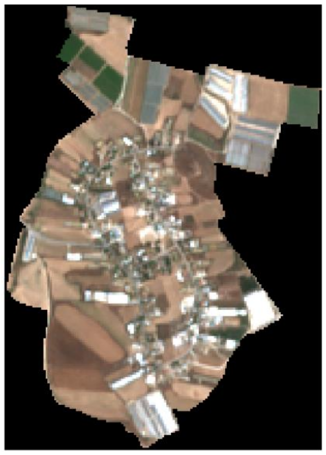


Fig 1. No'am: standard RGB

The data extraction pipeline started with automating the download and unzipping of Sentinel-2 imagery. Each dataset was stored as a compressed `.zip` file, which was then unzipped to reveal folders containing the images and associated metadata. These metadata files, stored in `.xml` format, were crucial for conducting radiometric corrections necessary to convert raw image values into physical reflectance values.

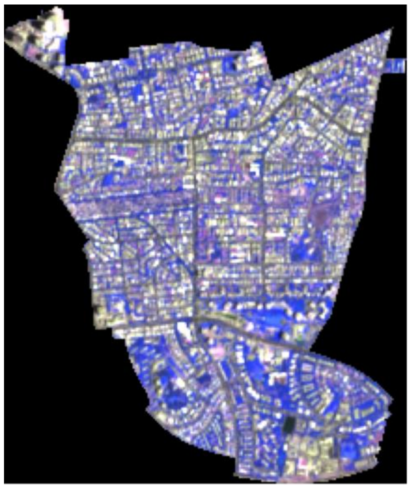


Fig. 2: Givatayim, RGB with blue as NIR band.

Using Python's `xml.etree.ElementTree`, we parsed the `.xml` metadata to extract quantification values, reflectance conversion factors, radiometric offsets, solar irradiance values, and special values such as NODATA and saturated pixel indicators. These values were essential to ensuring the radiometric accuracy of the imagery, allowing for correct interpretation of the satellite data.

Data Extraction of Aerial Imagery

For the spatial division of the area of interest, we used shapefiles from OpenStreetMap (OSM) for municipal boundaries (e.g. Fig. 2) and from DataGov for statistical areas (e.g. Fig. 1). Using aerial imagery metadata, we extracted the footprint (Fig. 3) of each image and calculated its coverage over each polygon. We then selected the imagery raster for each area by applying a mask based on maximum coverage.

Radiometric corrections were applied to the image bands, masking out NODATA and saturated pixels (using 98%-2% filtering). Corrected bands were then normalized for visualization, with the RGB channels being aligned to ensure proper overlay and visual consistency.

Finally, the processed data, including polygons and corrected imagery, was saved in pickle files for efficient storage and future retrieval.

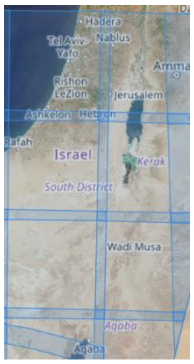


Fig. 3: footprints

Data Preprocessing and Model Development

Used libraries

To effectively utilize the satellite imagery for forecasting socio-economic indicators, we initiated the data preprocessing phase by setting up the computational environment with essential libraries such as NumPy, Pandas, OpenCV, and PyTorch. These libraries facilitated numerical computations, data manipulation, image processing, and deep learning model development, respectively.

Data Loading and Integration

Taking Our dataset consists of preprocessed Sentinel-2 satellite images stored in pickle files, where each file contains a dictionary mapping statistical areas to NumPy arrays. We opted not to include results for municipalities due to poor performance, primarily caused by the large variance in size (e.g., very large municipalities with small populations compared to statistical areas that only represent inhabited regions).

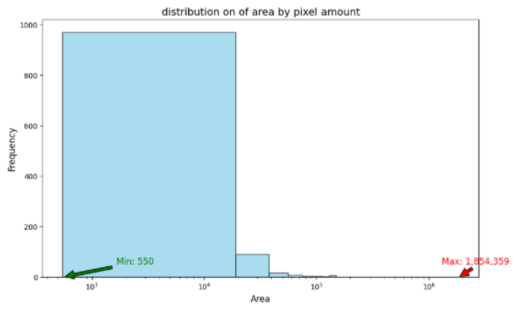


Fig 4. Area variability (pixel = 10m)

Simultaneously, we acquired socio-economic scores and population data for each statistical area from official sources. This response variable data included the socio-economic index (The index takes continuous values from -3 to 3, with a centered distribution, Fig. 5) and population figures, which were essential for our predictive modeling. To calculate population density, we also obtained the area size for each statistical region.

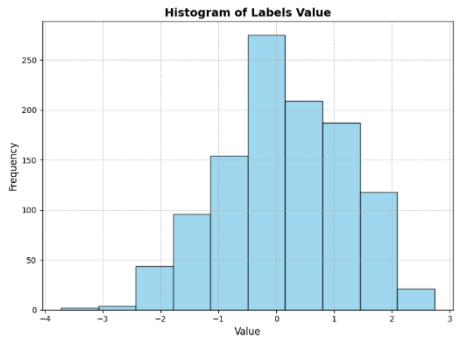


Fig 5. Labels Histogram

at the end, only the socio-economic index was examined due to the poor results of the density (potentially caused by a poor model).

Data Alignment and Label Creation

To ensure consistency between the imagery data and the socio-economic indicators, we implemented a filtering mechanism. This involved matching the statistical area codes from the satellite imagery data with those in the socio-economic dataset. We created functions to verify these matches and extract the corresponding labels, resulting in a dictionary that linked each statistical area's image data with its socio-economic score and population density.

Image Preprocessing and Standardization

Given the variability in image sizes due to differing statistical area boundaries, we needed to standardize the images for model input. We developed functions to adjust image sizes by either randomly cropping larger images or padding smaller ones with zeros to reach a uniform dimension of 224x224 pixels (recall: Each pixel contains four channels and represents a 10-meter resolution in reality).

Data Augmentation

To enhance the robustness of our model and mitigate overfitting, we applied data augmentation techniques. Specifically, we introduced random rotations to the images, increasing the diversity of our dataset. This approach helped the model generalize better by exposing it to various orientations and reducing sensitivity to the positional variance of features within the images.

Dataset Construction and Splitting

We encapsulated our preprocessed images and their associated labels into a custom PyTorch Dataset class. This structure facilitated efficient data handling and streamlined integration with PyTorch's DataLoader for batch processing. We then partitioned the dataset into training and testing sets using an 80-20 split (for validity).

Model Architecture and Training

For the predictive model, we selected a ResNet-50 architecture due to its proven effectiveness in image analysis tasks. To accommodate the four-channel input (including the NIR band), we modified the first convolutional layer of the network. Additionally, we adjusted the final fully connected layer to output a single value, aligning with our regression objective of predicting continuous socio-economic indicators.

We employed the Mean Squared Error (MSE) loss function to quantify the difference between the model's predictions and the actual values. The Adam optimizer was chosen for its efficiency in handling sparse gradients and adaptability to different data distributions. Training was conducted over multiple epochs (10), with the model iteratively learning to minimize the loss function through backpropagation and weight updates.

Model Evaluation

Error Evaluation

After training, we assessed the model's performance (Fig 6.) using the test dataset. The validation loss provided insights into the model's predictive accuracy and ability to generalize. To further interpret the model's decision-making process, we applied Gradient-weighted Class Activation Mapping (Grad-CAM). This technique enabled us to visualize the areas of the satellite images that the model found most influential for its predictions. The Grad-CAM heatmaps highlighted critical features such as vegetation density, urban structures, and other land cover elements associated with socio-economic conditions.

As shown in Fig. 6, the model's performance was poor, with a score of 0.3. Furthermore, we observed no clear relationship (Fig. 9, Fig. 8) between the error—represented by the percentage of non-black pixels—and the inclusion of additional visible areas. This indicates that the model is not effectively learning the relevant information from the data.

Despite the model's overall suboptimal performance, it still demonstrates some predictive power, particularly in affluent areas. This can be seen in Fig. 6, where it appears that the model's predictions improve in wealthier areas. Additionally, examining the Grad-CAM heatmaps reveals that the model has occasionally captured underlying patterns in the data. An example of this can be seen in Fig. 10, where a park in Givatayim contributed to the model's performance. In contrast, in poorer areas, the model sometimes based its predictions on black parts of the image.

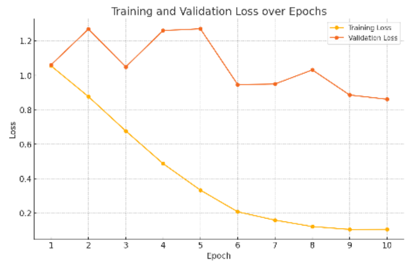


Fig. 6

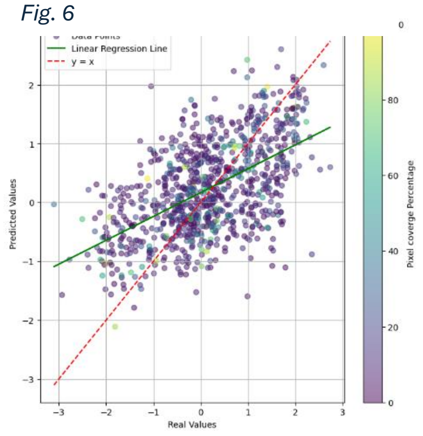


Fig. 7 R^2: 0.310

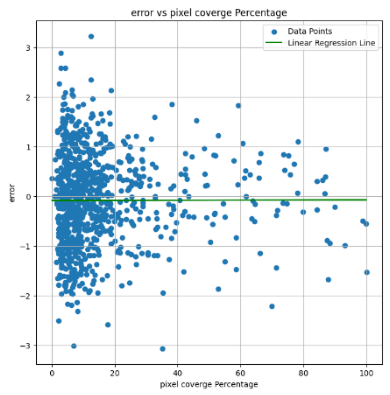


Fig. 8 R^2 : 0.000

Conclusion

In conclusion, despite applying advanced techniques such as data preprocessing, augmentation, and leveraging the NIR channel to capture vegetation and land-use details, the model's performance remained poor, as evidenced by the score of 0.3. The Grad-CAM visualizations, while offering some interpretability, did not reveal a clear connection between the regions the model focused on and the errors it produced.

This suggests that the model is not effectively learning the critical information required for accurate socio-economic forecasting from the satellite data. The lack of improvement with the inclusion of additional visible areas further indicates that the current approach may require

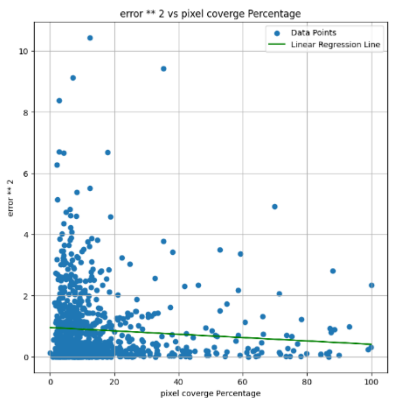
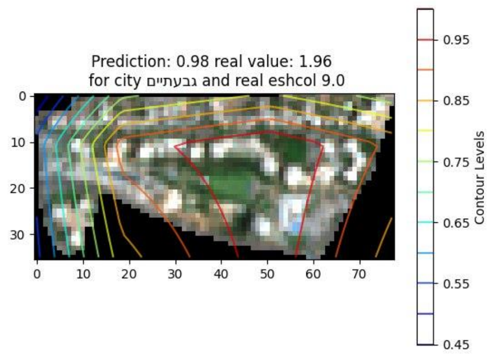


Fig. 9 R^2 : 0.006

significant adjustments, such as refining the model architecture, feature selection, or exploring alternative methodologies, to achieve meaningful results.



Prediction: 0.98 real value: 1.96
for city גבעתיים and real eshcol 9.0
Contour Levels
0.95
0.85
0.75
0.65
0.55
0.45

Fig. 10

References

- Copernicus Data Space Ecosystem (Official website)
- Image Contrast Enhancement by Percentile Stretch (wiki)
- Dimitrovski, I., Kitanovski, I., Kocev, D., & Simidjievski, N. (2023). *Current trends in deep learning for Earth observation: An open-source benchmark arena for image classification*. ISPRS Journal of Photogrammetry and Remote Sensing. Available at Papers with Code and UKIM Repository.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep Residual Learning for Image Recognition*. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 770–778. IEEE. https://doi.org/10.1109/CVPR.2016.90