Land Use and Land Cover Mapping: Utilizing CNNs for Segmentation and Classification of 64x64 Sentinel-2 Image Patches

Final Report for ENM 5310, Spring 2023

Richard Ren*

April 2023

1 Introduction and Literature Review

Land use and land cover mapping is used to identify and classify the different types of land use (e.g. residential, commercial, agricultural, and industrial use) as well as land cover (e.g. forests, grasslands, wetlands, or bare soil), helping to understand socioeconomic and ecological processes.

Often times, this data is obtained through remote sensing – which can include wavelengths in different wavelengths of the electromagnetic spectrum, including visible light, infrared radiation, microwaves, and radio waves. This is often known as *multispectral* data (representing all spectral bands, including those beyond the visible spectrum), a subset of which is known as *visual* data (visible to the human eye and typically represented with the RGB color model). There exist many open and publicly available satellite data sources from LANDSAT, Sentinel-2, Sentinel-1, etc. – many of which can be accessed through repositories such as the Copernicus Open Access Hub.

Given that $\tilde{2}3\%$ of global human-caused greenhouse gas emissions come from land usage such as agriculture and urban expansion [1], accurate land-cover maps form a baseline for environmental modeling and risk analysis.

Previous literature has demonstrated that improved projections of land-use and land-cover classes help modelers understand climate change mitigation potential in the U.S. Great Plains Wetlands [2], assist in conversation planning and habitat restoration for freshwater biodiversity [3] and Califronian sage scrub [4], and improve hydrological modeling and planned management of water resources, especially with issues related to flood risk and water quality in the Upper Mississippi River Basin [5]. Furthermore, commercial startups such as Land IQ have utilized time-series remote sensing data to assist local governments in enforcing compliance on agricultural water usage regulations through the measure of evapotranspiration [6].

In the past, there has been a wide usage of quickly classifying land cover via classical machine learning algorithms such as maximum likelihood classification, k-nearest neighbors, and k-means clustering [7]. Other proposed techniques include self-organizing maps, artificial neural networks with cellular automata, and stochastic spatial random forests **8-9**. The application of machine learning to has enabled mapping land use and land cover at an unprecedented scale, detection socioeconomic trends throughout time (e.g. understanding deforestation), and detailed spatial projections of future land-use and land-cover change.

Even newer modern deep learning techniques utilize convolutional neural networks (CNNs) and graph neural networks to process spatial information, and utilize recurrent neural networks or long-short memory networks to process time series data and detect changes [8] – essentially taking advantage of spatial, spectral, and temporal priors in combination with strong learning and feature expression abilities to achieve superior performance. Notable deep learning methods that have achieved high accuracy include DenseNet, UNet, DeepLab, and SegNet [9].

*Computer Science Department of the University of Pennsylvania; author can be reached at renrich@wharton.upenn.edu or renrich@seas.upenn.edu

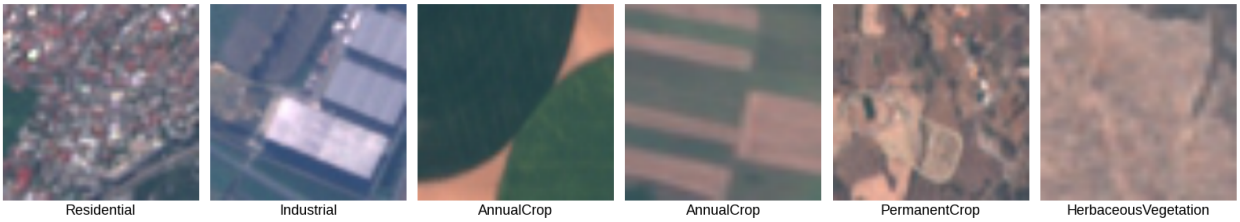

Figure 1: Visualization of labeled examples in the EuroSAT dataset.

We aim to focus on investigating the viability deep learning models in predict urban features in different cities, both within and outside of 64x64 square patches generated by the European Space Agency (EUA) Sentinel-2 satellite.

2 Problem Statement

2.1 LULC Classification of 64x64 Images

Let's denote the input image as I, which has a spatial size of H x W and consists of C channels (e.g., RGB or multispectral bands). The goal of the CNN is to generate an output map Y, where each entry y_{ij} corresponds to the predicted class label for the pixel or region located at position (i, j) in the input image. Mathematically, F is the CNN model that must be trained to map the input image I to the output map Y. We test this on EuroSAT data, an already cleaned dataset.

2.2 LULC Segmentation within 64x64 Satellite Images

Given an (H, W, C) remote sensing image, where H is the height, W is the width, and C is the number of channels of the image, a CNN segmentation algorithm F: image(H,W,C) $\rightarrow$ class(H,W) will output a mask image of the same size representing different land use and land cover classes. The mask image is an integer array, whereby 0 denotes background pixels and 1, ..., N denote the pixels occupied by different land use and land cover classes, with each individual class labeled by a unique integer value. We test this in both Ankara, Turkey and Paris, France.

3 LULC Classification of 64x64 Images

3.1 Dataset

The EuroSAT dataset consists of 27,000 high-resolution (30m per pixel) satellite images covering 13 spectral bands of the Sentinel-2 satellite sensor. The images cover ten land use and land cover classes: annual crop, forest, herbaceous vegetation, highway, industrial, pastures, permanent crop, residential, river, and sea and lake. Labeled examples are shown in Fig. 1.

3.2 Architecture: ResNet-50

Common Convolutional Neural Network (CNN) Layers Overview

Convolutional layers apply a set of kernels to an input volume of data, performing the convolution operation at each locations. A 2D convolution operation between an input matrix (or image) I and a kernel matrix K is given by:

$$(I * K)_{i,j} = \sum_m \sum_n I_{m,n} K_{i-m,j-n}$$

Pooling layers are used to reduce the spatial dimensions – and the most common variant is max pooling, where a maximum value is selected for a given "kernel" that represents a rectangular window.

Commonly used for computer vision, successive convolutional and pooling layers take far less parameters than their fully connected counterparts, and usually impart priors of spatial invariance, stationarity, locality, and compositionality.

ResNet Overview

Often, deep neural networks face the vanishing or exploding gradient problem, where the gradients become very small or very large during backpropagation, causing weights to update too little (slow convergence issues) or too much (numerical stability and divergence issues) – leading to a situation where a deeper network may see degraded accuracy despite not overfitting. The ResNet architecture was introduced in to enable deep neural networks that counter this vanishing/exploding gradient problem.

The ResNet utilizes residual blocks, where an input x is modified through multiple convolutional layers to become $F(x)$. Afterwards, at the end of the residual block, the final formulation is expressed as $F(x)+x$. The re-addition of the input is known as a "shortcut connection" (or "skip connection") because of the identity mapping skipping multiple layers.

There are two large advantages to the ResNet architecture:

1. Intuitively, the convolutional layers that form $F(x)$ is calculating some residual of the input that will be re-added to the layer. This enables the model to learn the identity function when needed, as well as small variations over the identity function (rather than the entire transformation).

2. The shortcut connections in ResNet makes the vanishing or exploding gradient issue far less likely. To show this, assume that there is an input x with an output x' with a series of forward propagation layers $f(x)$; the forward propagation of a residual block looks like:

$$x' = f(x) + x$$

The backpropagation through this residual block can therefore be expressed as:

$$\frac{\partial x'}{\partial x} = 1 + \frac{\partial f(x)}{\partial x}$$

Because the gradient $\frac{\partial x'}{\partial x}$ has term Ï; at least part of the gradient will not vanish or explode as it is backpropagated through the network.

The ResNet architecture experimentally saw far increased accuracy gains compared with other techniques at its introduction, and residual connections are often used in modern transformer blocks such as BERT and GPT.

ResNet-50

ResNet-50 is a 50-layer deep neural network architecture that consists of an initial convolutional layer with 64 filters, a 3x3 max-pooling layer, and four stages of convolutional layers with residual connections. The first stage contains 3 residual blocks with 64 filters, the second stage has 4 blocks with 128 filters, the third stage has 6 blocks with 256 filters, and the final stage has 3 blocks with 512 filters. Each block has 1x1 and 3x3 convolutional layers followed by batch normalization and ReLU activation. The shortcut connections are implemented using 1x1 convolutions with the appropriate number of filters; these layers are meant to reduce the depth of feature maps and "condense" information before the next convolution layer. The network ends with global average pooling, a fully connected layer with the number of output classes, and a softmax activation.

3.3 Training Objective and Optimization

We utilize the pretrained ResNet-50 convolutional neural network and fine-tune it for this satellite data classification task. We utilize a 80-20 train test split for our data. We utilized the randomresizedcrop, randomhorizontalflip, and randomverticalflip transforms on training data.

Stochastic gradient descent with minibatches is used for training. Given a training set of N labelled datasets in a batch, with image x_i and associated class label y_i for i = 1,...,N, the training objective is defined as:

$$L = \frac{-1}{N} * \sum_i (y_i * log(p(x_i)) + (1 - y_i) * log(1 - p(x_i)))$$

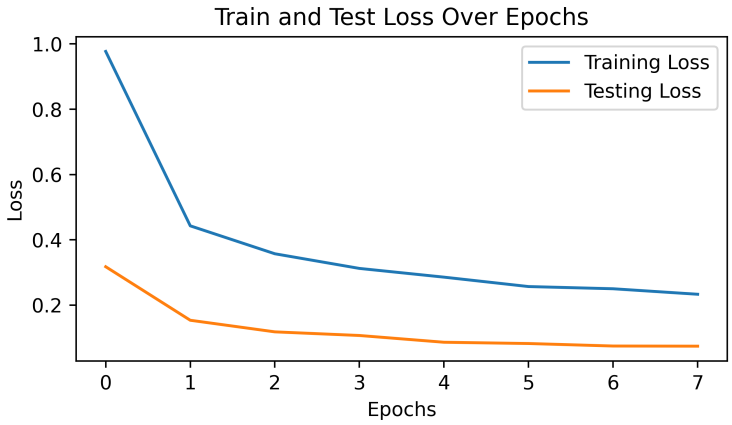

Figure 2: Plot of loss function and testing accuracy over training epochs.

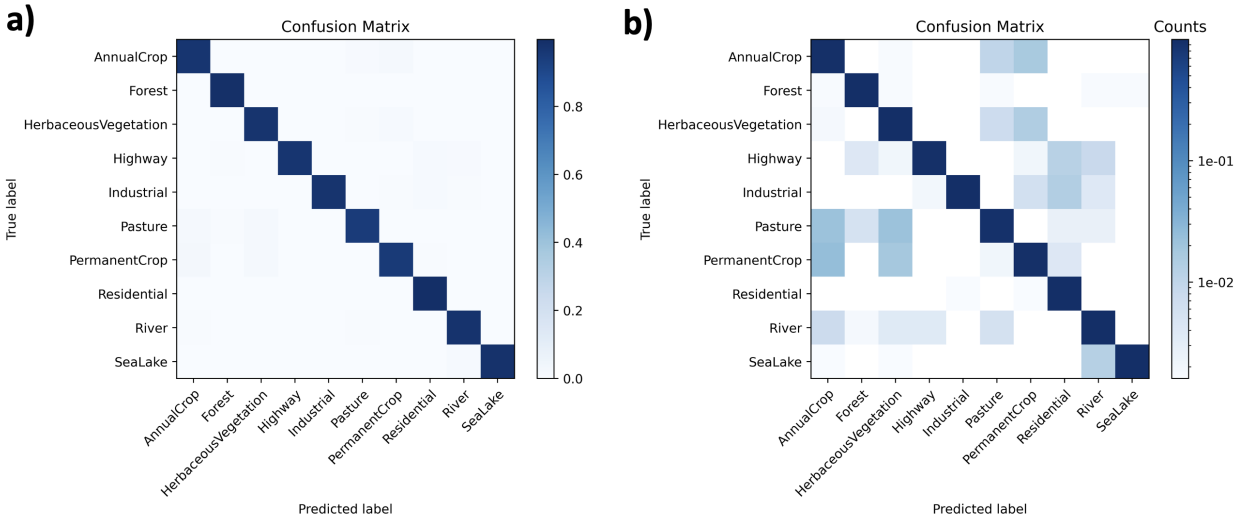

Figure 3: Classification matrices, in (a) absolute and (b) logarithmic scale.

where $p(x_i)$ is the predicted probability of the image x_i belonging to the correct class, given the network's parameters.

3.4 Results

We find that the model has extremely high testing accuracy, surpassing even the training data in Fig. 2. We hypothesize this is because the testing dataset contains slightly "easier" examples than the testing dataset. As shown by the confusion matrix in Fig 3a, the model seems to perform very well.

A classification matrix can be defined with $C = [c_{ij}]_{KxK}$ where K is the number of prediction classes and c_{ij} is the number of instances with true class label i and predicted class label j. We show both absolute classification matrix C in Fig. 3a as well as logarithmic classification matrix $log(C)$ in Fig. 3b to emphasize small changes that may not be visible in C.

4 LULC Segmentation within 64x64 Images

4.1 Dataset

Sentinel-2 Satellite Data

We use S2MSI2A satellite data, a type of satellite data product provided by the Copernicus program's Sentinel-2 mission. The S2MSI2A product type is a Level-2A product, which means that it has undergone atmospheric correction and includes surface reflectance values; it is often used in land cover classification, vegetation monitoring, crop mapping, and urban planning. We choose latitude and longitude coordinate ranges for Ankara, Turkey and Paris, France – our two areas of interest – to be (39.83 to 40.01, 32.53 to 33.03) and (48.67 to 49.04, 1.90 to 2.84) respectively. We extract 10 meter spatial resolution imagery, only using bands 2, 3, and 4 which correspond with red, green, and blue. We also take 20 meter spatial resolution SCL imagery (which gives minor details on water, clouds, vegetation, bare soil, and urban areas). This can be seen for Paris and Ankara in Fig. 4a and 5a respectively.

European Urban Atlas

We utilize the European Urban Atlas (EUA), which provides detailed land use and land cover courtesy of the European Environment Agency in collaboration with the European Commission's Joint Research Centre. We use the EUA maps for Ankara and Paris, condensing them until they meet our coordinate ranges of interest. It is notable that while there exist 27 different classifications, these classifications tend to be very imbalanced in their distribution. We rasterize these vector maps to create a segmentation mask that covers every pixel, helping to complete our training and testing dataset. This can be seen for Paris and Ankara in Fig. 4b and 5b respectively.

Training and Testing Patch Generation

We generate 64 by 64 patches of satellite data containing red, blue, green, and SCL data, and discard images with any NaN values as well as cloud cover limit above 15% (where cloud cover limit is determined by SCL classification). This can be seen for Paris and Ankara in Fig. 4c and 5c respectively. While the SCL classification cloud cover limit is not perfect, this should help get rid of excessively cloudly data. We then utilize a 80-20 train test split for our data and test each model on the city it was trained on.

4.2 Architecture: UNet

The UNet was first introduced in [10] for biomedical image segmentation tasks. It uses an encoder-decoder structure, helping the network condense its spatial information into many feature channels while then expanding those features to assign a class probability to each pixel; this allows the network to propagate context information to higher resolution layers.

The original U-Net (named after its U-like shape) consists of four encoder and decoder blocks, where each encoder halves the spatial dimensions and doubles the feature channels, while each decoder block doubles the spatial dimensions and halves the feature channels.

- Each of the four encoders uses two 3x3 unpadded convolutions (alongside ReLU activations), a 2x2 max-pooling operation with stride 2 for downsampling.
- The "bridge" between the encoder and decoder consists of two 3x3 convolutions (alongside ReLU activations).
- Each of the four decoders start with a 2x2 "up-convolution" that halves the feature channels, concatenated with a skip connection feature map from the encoder block, then two 3x3 convolutions (alongside ReLU activations).
- Toward the final layer, a 1x1 convolution is used to map each feature vector to the desired number of classes.

We utilize two trained U-Nets based on an EfficientNetB0 backbone and fine-tune it for the Paris and Ankara land segmentation tasks.

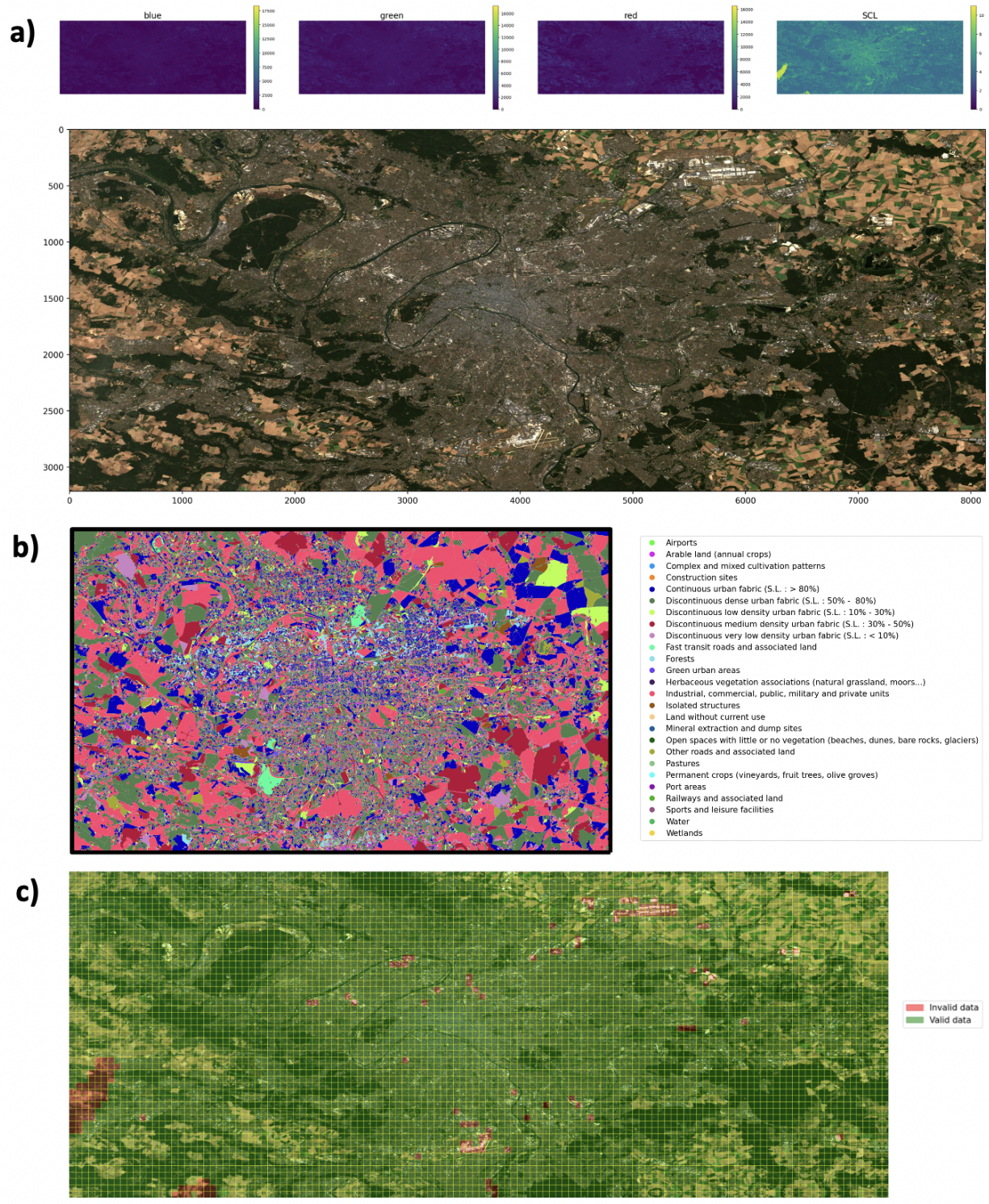

Figure 4: Visualization of (a) Sentinel-2 10x10m satellite data and (b) semantic EUA classification data utilized for area of interest encompassing Paris. (c) Certain data patches contained too much cloud cover and were discarded.

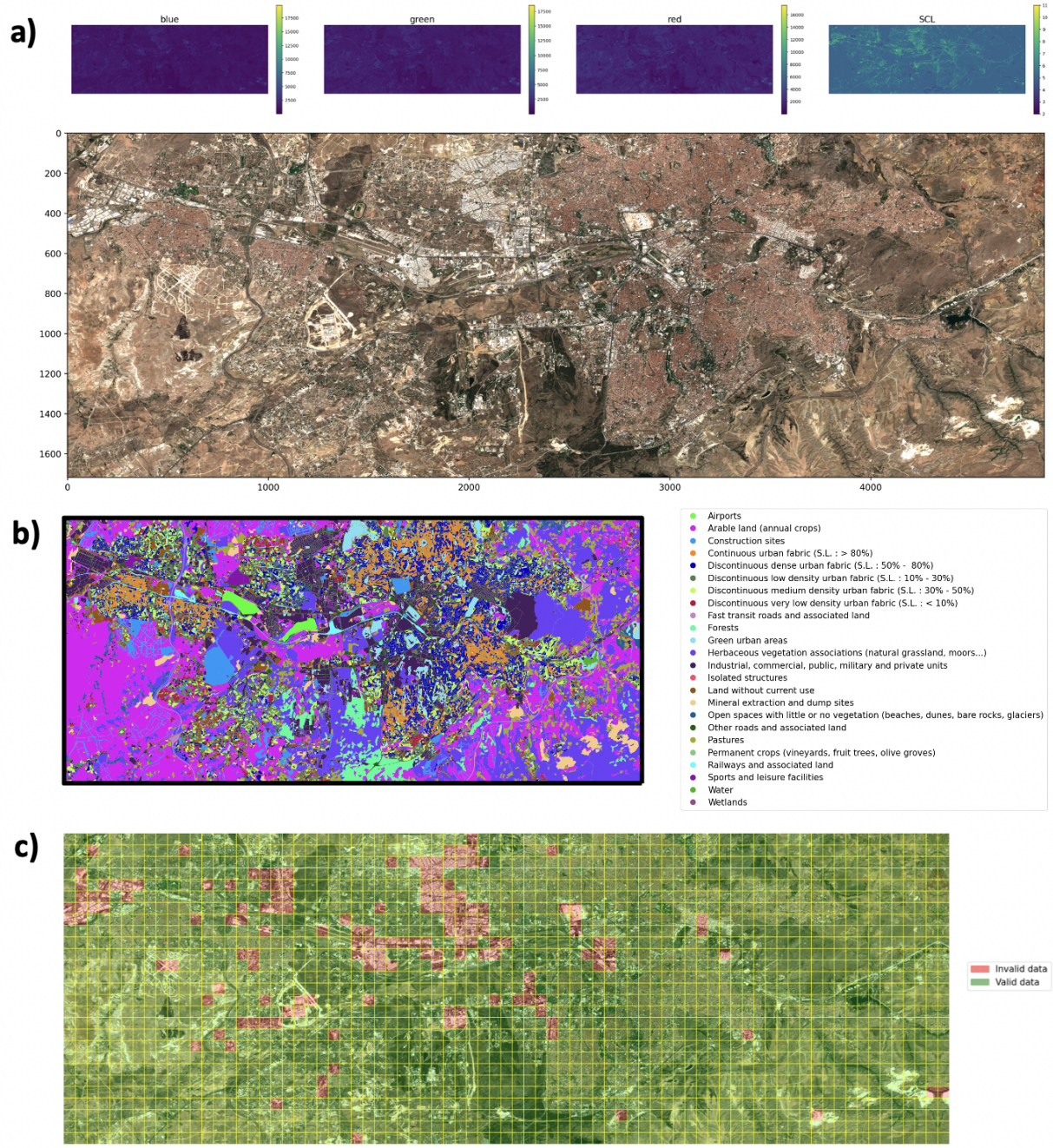

Figure 5: Visualization of (a) Sentinel-2 10x10m satellite data and (b) semantic EUA classification data utilized for area of interest encompassing Ankara. (c) Certain data patches contained too much cloud cover and were discarded.

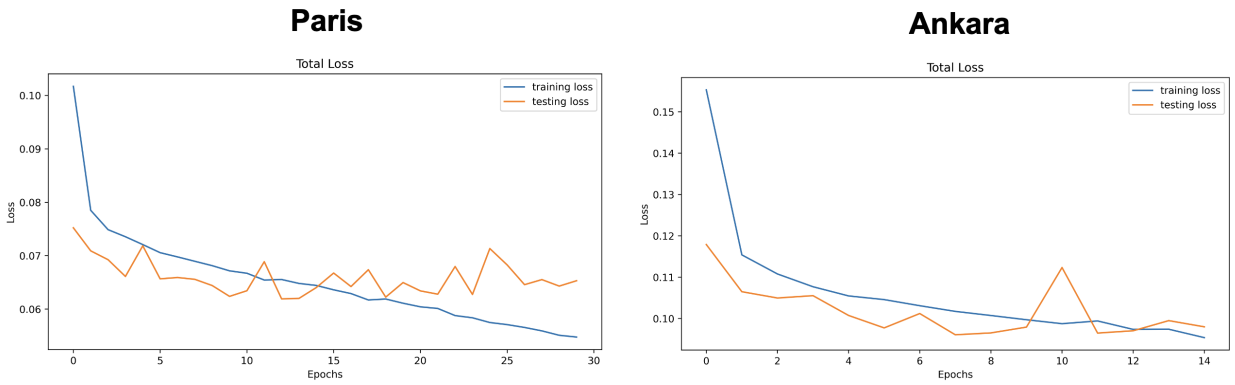

Figure 6: Plot of loss function and testing accuracy over training epochs.

4.3 Training Objective and Optimization

The cross entropy loss is computed between predicted probabilities of each pixel and the one-hot encoded ground truth layers, as common in a multi-class segmentation problem.

Given an image domain Ω, let $p_k(x)$ denote the predicted probability of pixel x belonging to class k, where $x \in \Omega$ and $k \in \{1, \ldots, K\}$. The softmax function is applied to the output logits of the network, resulting in a probability distribution over K classes for each pixel:

$$p_k(x) = \frac{e^{z_k(x)}}{\sum_{j=1}^{K} e^{z_j(x)}}$$

where $z_k(x)$ is the output logit for class k at pixel x.

Now, let $l(x)$ be the true class label of pixel x, and $w(x)$ be the weight associated with pixel x. $w(x)$ can be set unequally to give more importance to certain pixels or classes; we set $w(x)$ to 1 for all pixels. We can then define the weighted categorical cross-entropy loss function E defined as:

$$E = \sum_{x \in \Omega} w(x) \log \left(p_{l(x)}(x) \right)$$

4.4 Results

As one can see, the testing loss seems to falter while the training loss decreases, indicating an overfit model. The absolute and logarithmic classification matrix are shown in Fig. 7 and indicate that the model very weakly learns the correlations. However, many classes do not show up often in the dataset, and even the classes that do seem to have a noisy and unreliable classification.

Fig. 8 indicates that the model may have learned representations relatively well. At times, the CNN segmentation model struggles with details, possibly due to the poor resolution of the photos or the overfit nature of the model.

5 Conclusion

This was my first time working with satellite data, and I'm grateful for the opportunity to work on this (as well as Professor Perdikaris's flexibility with allowing me to focus on one of the subprojects I proposed, when the project ended up being more difficult than I anticipated). As one can probably tell, the first part was far easier and simply involved taking a cleaned dataset and using a ResNet architecture; the second part of the project took a lot of effort and debugging as it was downloading Sentinel-II images from scratch and cleaning them on my own to create compiled satellite dataset.

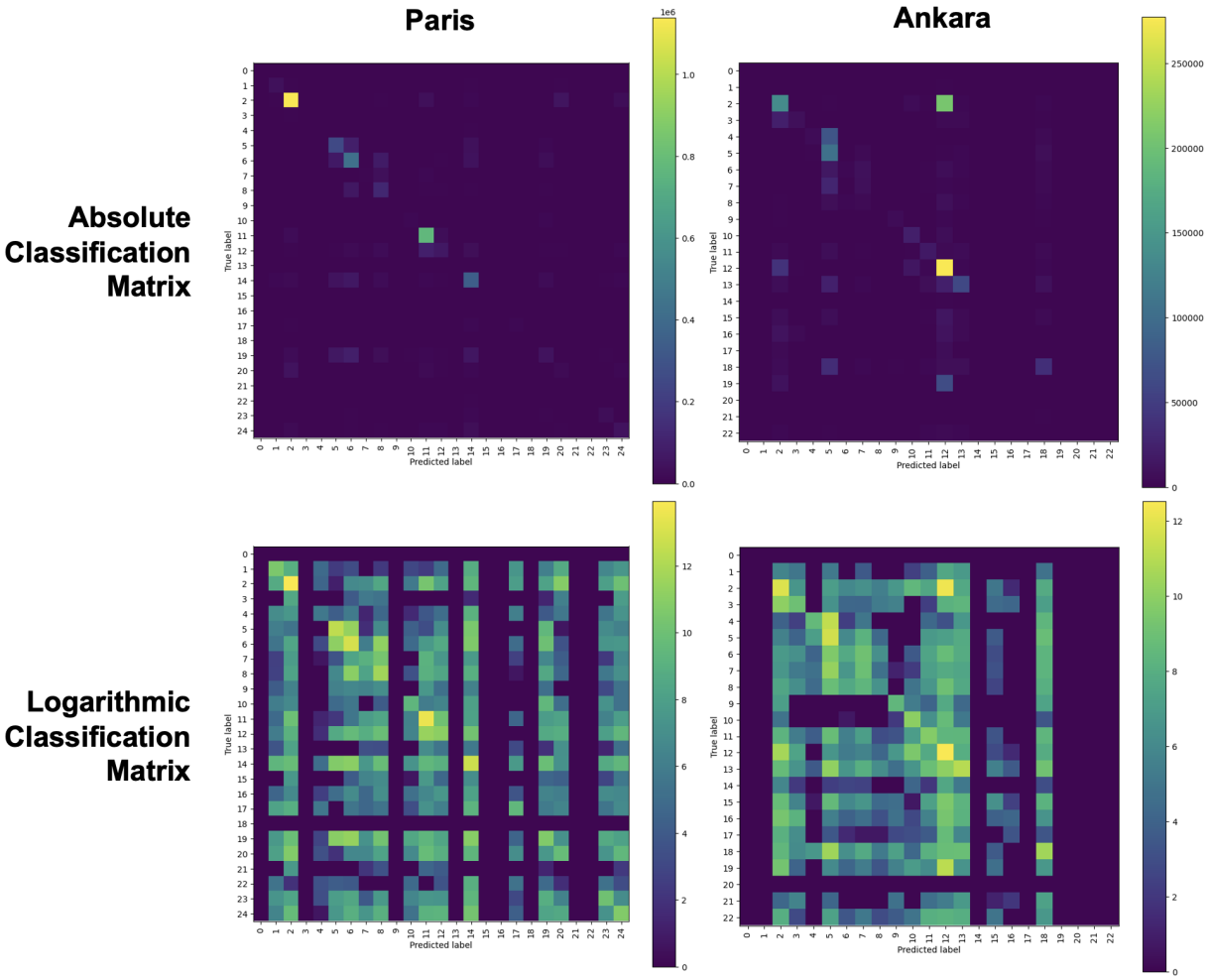

Figure 7: Classification matrix, where each pixel is a classification task.

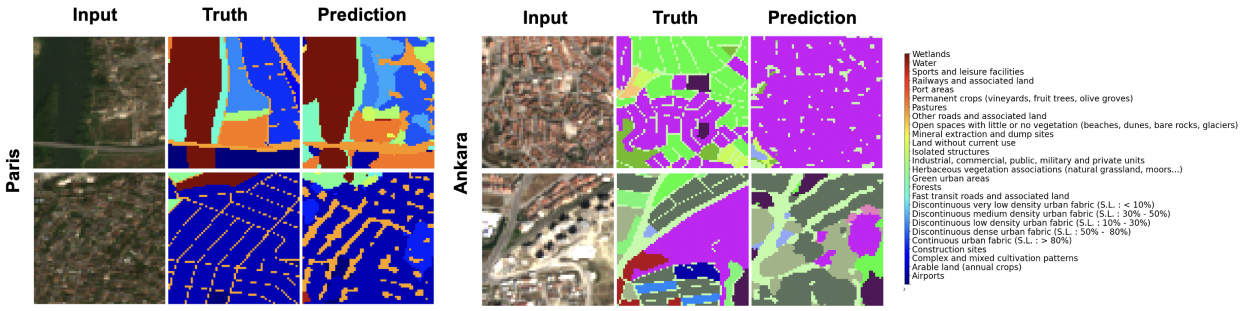

Figure 8: Model input vs ground truth vs prediction: selected examples in Paris and Ankara.

Due to the large amount of time spent data cleaning, looking for online guides for handling satellite data, understanding packages and file types (e.g. learning to use XArray, GeoJSON, geopandas, rioxarray for manipulating and analyzing geographic data; reading and writing various geospatial file formats; performing spatial joins, spatial querying, plotting maps; etc.), and debugging my data cleaning pipeline, I was unfortunately unable to do these fun extensions I really wanted to:

- Run a hyperparameter sweep using wandb.io, which was another tool I hoped to learn to use throughout this project
- Explore satellite data beyond the European Space Agency's Sentinel-2 satellites
- Trying to understand the mapping between LULC and GDP projections (e.g. measuring the R^2 correlation coefficient)
- Testing a wider variety of segmentation models, including ones inspired by vision transformers
- Using datasets on urban flood mapping and crop parcel mapping
- Interpolating between maps in a Gaussian Process-like manner
- Understanding important urban features in a Variational Encoder-like manner (e.g. gradual land use categories). Using VAE-like structures to create "fake cities" (e.g. give me a city that's a mix of Cairo and Bangkok).

These can be the focus of my future efforts.

5.1 Other Credit and Attribution

I owe a large amount of credit to these guides to segmentation and satellite data cleaning[1][2][3], whose code I essentially replicated on my own before expanding upon it to form this project. These served as useful tutorial guides for this project, and I would have struggled with data cleaning otherwise.

This Medium article helped me understand how the UNet architecture worked.[4]

Bibliography

[1] K. Levin and S. Parsons, "7 things to know about the ipcc's special report on climate change and land," *World Resources Institute*, Aug. 2019. [Online]. Available: https://www.wri.org/insights/7-things-know-about-ipccs-special-report-climate-change-and-land.

[2] K. Byrd, J. Ratliff, N. Bliss, *et al.*, "Quantifying climate change mitigation potential in the united states great plains wetlands for three greenhouse gas emission scenarios," *Mitigation and Adaptation Strategies for Global Change*, vol. 20, no. 3, pp. 439–465, 2015. DOI: 10.1007/s11027-013-9500-0.

[3] S. Panlasigui, A. J. S. Davis, M. J. Mangiante, and J. A. Darling, "Assessing threats of non-native species to native freshwater biodiversity: Conservation priorities for the united states," *Biological Conservation*, vol. 224, pp. 199–208, 2018, ISSN: 0006-3207. DOI: https://doi.org/10.1016/j.biocon.2018.05.019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S000632071731697X.

[4] E. C. Riordan and P. W. Rundel, "Land use compounds habitat losses under projected climate change in a threatened california ecosystem," *PLOS ONE*, vol. 9, no. 1, e86487, 2014. DOI: 10.1371/journal.pone.0086487. [Online]. Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0086487.

[1] Automating the creation of LULC datasets for Semantic Segmentation
[2] Land Use and Land Cover Classification using Pytorch
[3] Automating Land Use and Land Cover Mapping
[4] Medium: What is UNet?

[5] A. Rajib and V. Merwade, "Hydrologic response to future land use change in the upper mississippi river basin by the end of 21st century," *Hydrological Processes*, vol. 31, pp. 3645–3661, 2017. DOI: 10.1002/hyp.11282. [Online]. Available: https://doi.org/10.1002/hyp.11282.

[6] D. Charles. "Satellites reveal the secrets of water-guzzling farms in california," NPR. (Oct. 2021), [Online]. Available: https://www.npr.org/2021/10/18/1047224973/satellites-reveal-the-secrets-of-water-guzzling-farms-in-california.

[7] T. Mollick, M. G. Azam, and S. Karim, "Geospatial-based machine learning techniques for land use and land cover mapping using a high-resolution unmanned aerial vehicle image," *Remote Sensing Applications: Society and Environment*, vol. 29, p. 100 859, 2023, ISSN: 2352-9385. DOI: https://doi.org/10.1016/j.rsase.2022.100859. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352938522001677.

[8] J. Wang, M. Bretz, M. A. A. Dewan, and M. A. Delavar, "Machine learning in modelling land-use and land cover-change (lulcc): Current status, challenges and prospects," *Science of The Total Environment*, vol. 822, p. 153 559, 2022, ISSN: 0048-9697. DOI: https://doi.org/10.1016/j.scitotenv.2022.153559. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0048969722006519.

[9] W. Yang, H. Song, L. Du, S. Dai, and Y. Xu, "A change detection method for remote sensing images based on coupled dictionary and deep learning," *Computational Intelligence and Neuroscience*, vol. 2022, p. 3 404 858, 2022. DOI: 10.1155/2022/3404858.

[10] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *arXiv preprint arXiv:1505.04597*, 2015.