Project Presentation

Deforestation in Bolivia: an image segmentation approach

Fanni Varhelyi, Sanha Tahir, Sukriti Mahajan

Dec 1, 2023

The project at a glance

Importance

Carbon emissions related global warming is changing our planet. Bolivia has seen intense deforestation in the last couple of years

Analysis

We analyzed satellite images for the last 7 years leveraging image segmentation to identify deforestation on a detailed level

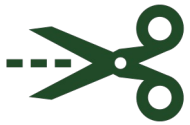
Results

We developed masks for newly acquired detailed satellite images from Planet and estimated the rate of deforestation

Deforestation & Climate change

Carbon emissions related global warming is changing our planet, and severe deforestation is accelerating the issue

Trees release carbon when cut down: in the 2010s, it's estimated that deforestation was responsible for yearly 5Gt-8Gt CO2 emissions



Carbon 'offset' programs or carbon removals have been gaining in popularity but have questionable quality

The capability to monitor and measure progress on a detailed level will be crucial going forward

Source: Economist, Guardian

Bolivia's per capita deforestation is four times worse than Brazil's, and there's less political willingness to change

- Global Forest Watch estimates that deforestation is increasing in Bolivia: 32% just from 2021 to 2022
- No political will to change: Bolivia blocked a pledge this year to end deforestation by 2030 at an Amazon summit
- Deforestation is cheaper than investing into existing land, driving expansion even for low returns
- Illegal deforestation is rarely penalized either, and the legally available land is increasing

Illustrative example: deforestation since 2016 is clearly visible on the two satellite images we acquired from Planet

Most deforested areas are turned into soya farms (Bolivia is the world's 10th largest soy producer), but beef, sugar cane, and corn are also responsible

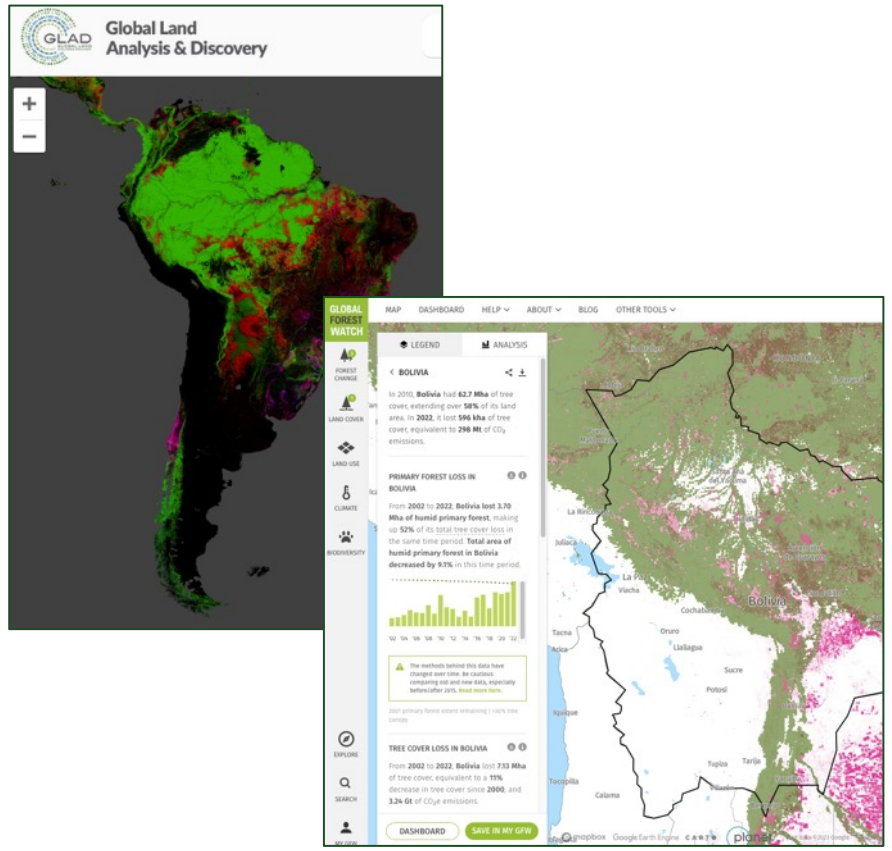
Source: Guardian, Trase Insights

Researchers lately leveraged image segmentation with various architectures to study deforestation

Summary of recent research shows interest in area and advances in image segmentation

- Researchers have been **investigating deforestation** for decades, traditionally with **image classifiers**
- Recently, focus has been on leveraging **image segmentation**: variations of **VGG, ResNet, or U-Net**
- **U-Net** is a **convolutional encoder-decoder** developed in 2015 for medical imaging
- **U-Net** was found to **outperform most other models** in forestry image segmentation (best accuracy ~94%)
- Researchers see encoder-based segmentation potentially **enabling real-time monitoring**

Published work typically focuses on map-based visualizations and statistics



Source: Global Forest Change (University of Maryland), Global Forest Watch, Bragagnolo et al. (2022), Alzu'bi and Alsmadi (2022), Bragagnolo et al. (2021), John & Zhang (2022)

Recap: image segmentation categorizes areas of images into polygon shapes, and returns pixel-wise masks

- Image segmentation can help both localize and detect objects in images
- Partitions an image into multiple regions or segments
- Groups together pixels with similar attributes
- Predicts the location, shape and class of each object
- Creates a pixel-wise mask for each object in the image

Semantic Segmentation

https://pytorch.org/vision/stable/models.html

Available semantic segmentation models

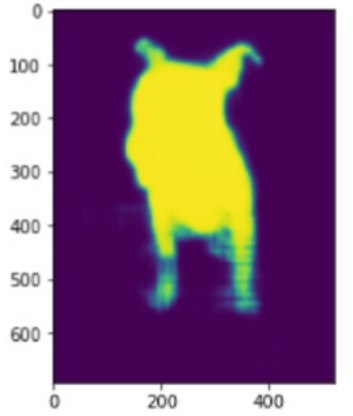
- DeepLabV3
- FCN
- LRASPP

Here is an example of how to use the pre-trained semantic segmentation models:

```
from torchvision.io.image import read_image
from torchvision.models.segmentation import fcn_resnet50, FCN_ResNet50_Weights
from torchvision.transforms.functional import to_pil_image

img = x

# Step 1: Initialize model with the best available weights
weights = FCN_ResNet50_Weights.DEFAULT
model = fcn_resnet50(weights=weights)
model.eval()

# Step 2: Initialize the inference transforms
preprocess = weights.transforms()

# Step 3: Apply inference preprocessing transforms
batch = preprocess(img).unsqueeze(0)

# Step 4: Use the model and visualize the prediction
prediction = model(batch)["out"]
normalized_masks = prediction.softmax(dim=1)
class_to_idx = {cls: idx for (idx, cls) in enumerate(weights.meta["categories"])}
mask = normalized_masks[0, class_to_idx["dog"]]
plt.imshow(x.permute(1, 2, 0)); plt.show()
print(mask.shape)
plt.imshow(mask.detach().numpy()); plt.show()
```

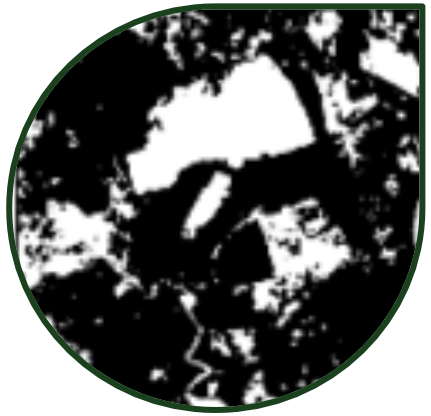
(Plain text of code is provided in lab-3.2)


```
torch.Size([692, 520])
```



Source: DSAN 6600 slides, Analytics Vidhya

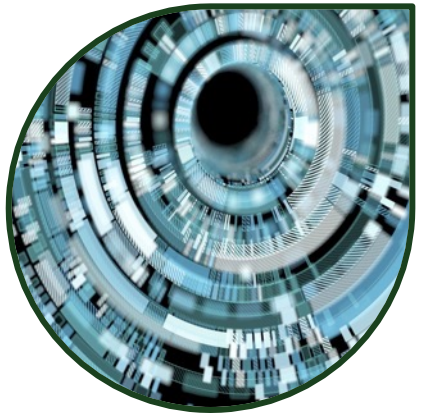
Our approach leveraged existing training data and a custom model to process newly acquired images



Training data

Existing rainforest satellite images with masks

Area: Amazon and Pacific, published by Bragagnolo et al. (2022)



Model

Modified U-Net and image segmentation models

Based on the literature review, re-built and tuned

Two models used as base case and main model

Test data

High quality satellite images of Bolivia

No masks available, covers the target area in detail (2016-2023, Planet)

Masks generated by the model

Results

Masks compared across time

Yearly and overall deforestation calculated

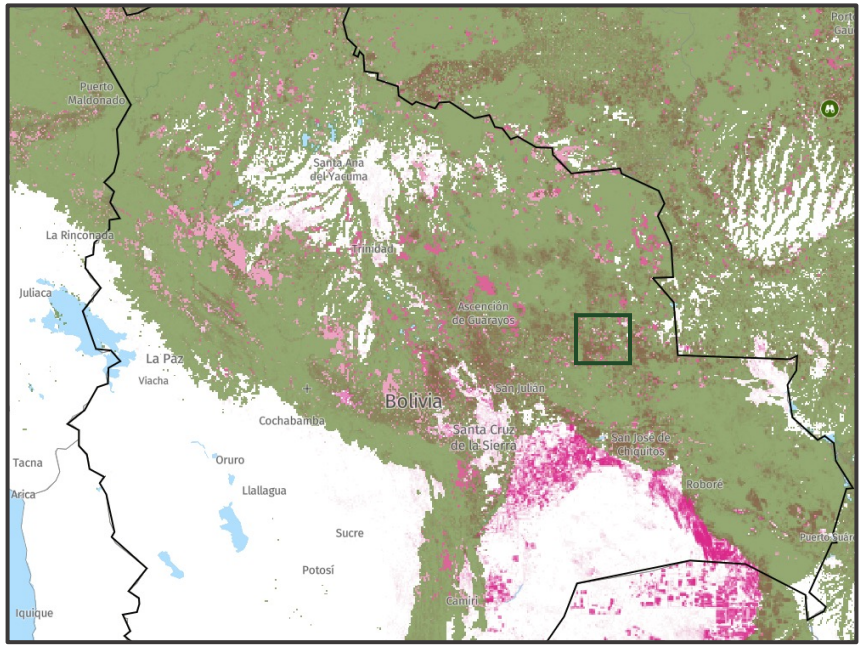
Results compared to existing estimates

Data collection was limited to an area within Bolivia: our goal was to cover an area with known deforestation

A specific area was selected within Bolivia for the analysis

Selected location:
15.69° - 16.48° S, 60.83° - 61.81° W

The polygon was placed based on existing deforestation estimates

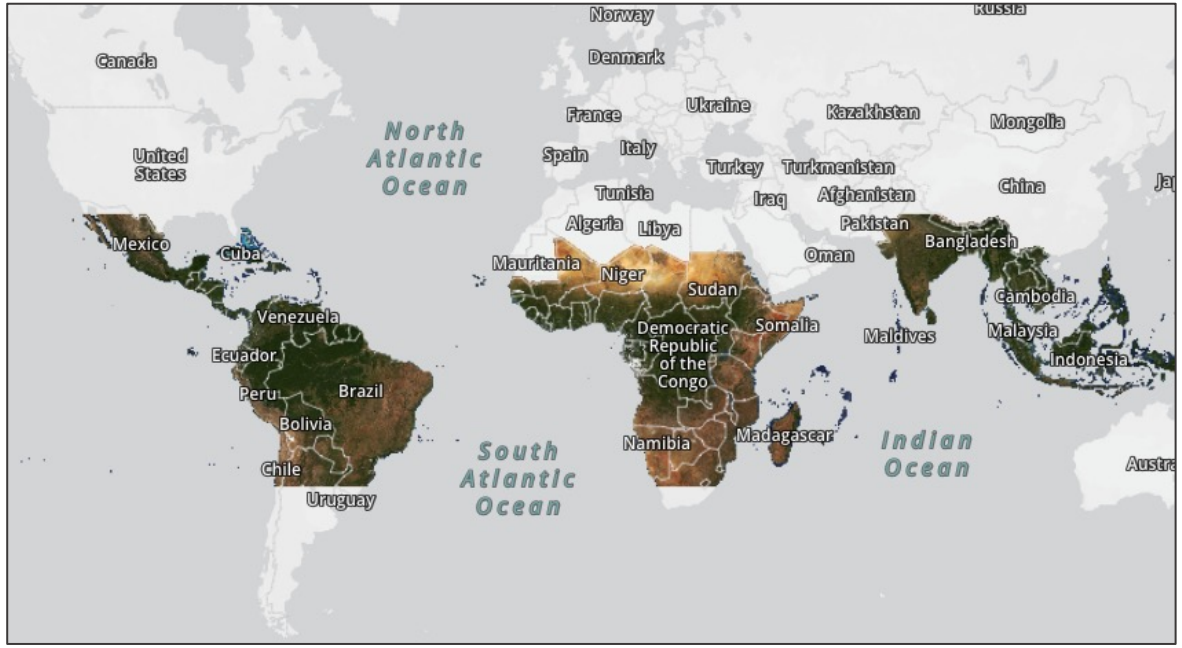


Pink highlights deforestation since 2002

Sources: Planet, Global Forest Watch

We collected data from Planet

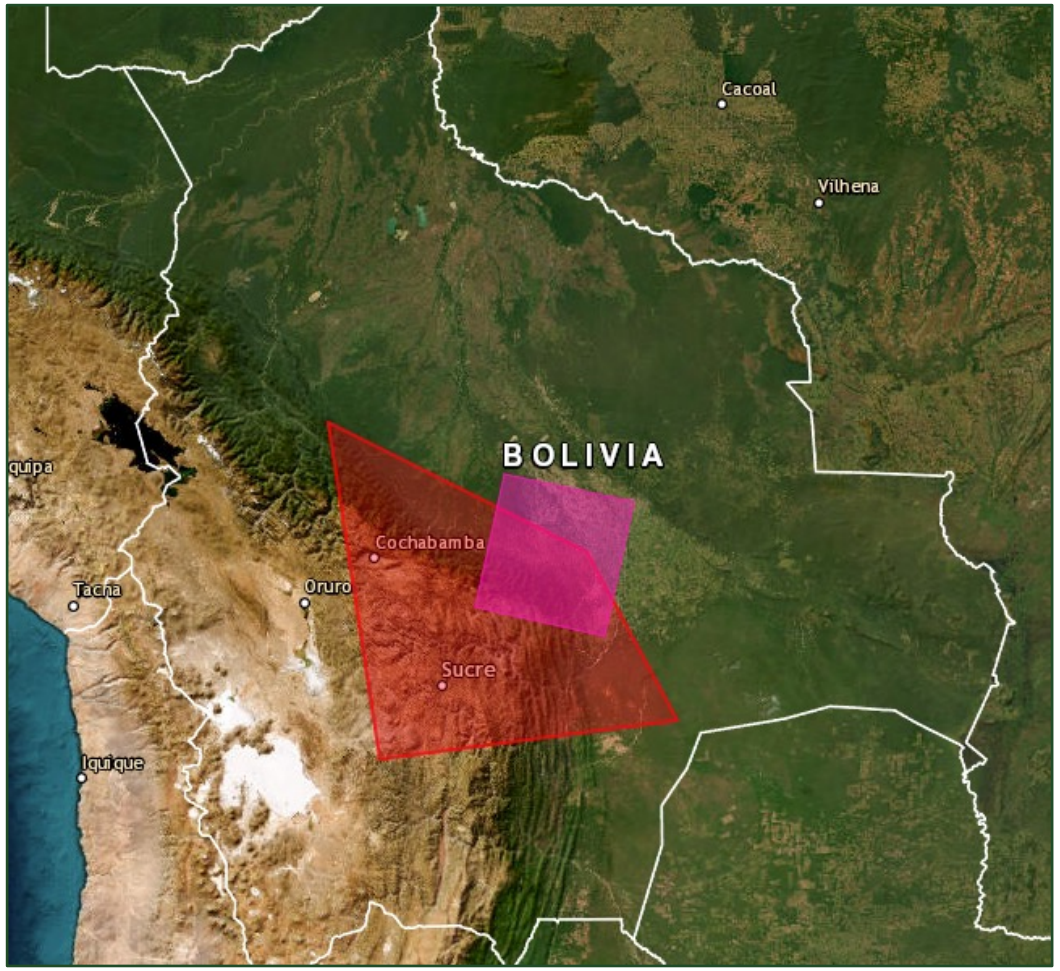
- Planet provides **detailed satellite imagery** and related analysis e.g., Forest Carbon Diligence
- Planet **operates** the PlanetScope (PS) and SkySat (SS) Earth-imaging **satellite constellations**
- **Tropical** imagery is **publicly accessible** via Norway's International Climate & Forests Initiative[1]
- **Each image is optimized** and comprised of multiple segments (e.g., no clouds)
- Specs: '**quads**' are **GeoTIFF** images, pixel resolution 4.77m, 4096 x 4096, 4 bands
- The **time period** is June/Dec 2015-2020, and monthly thereafter

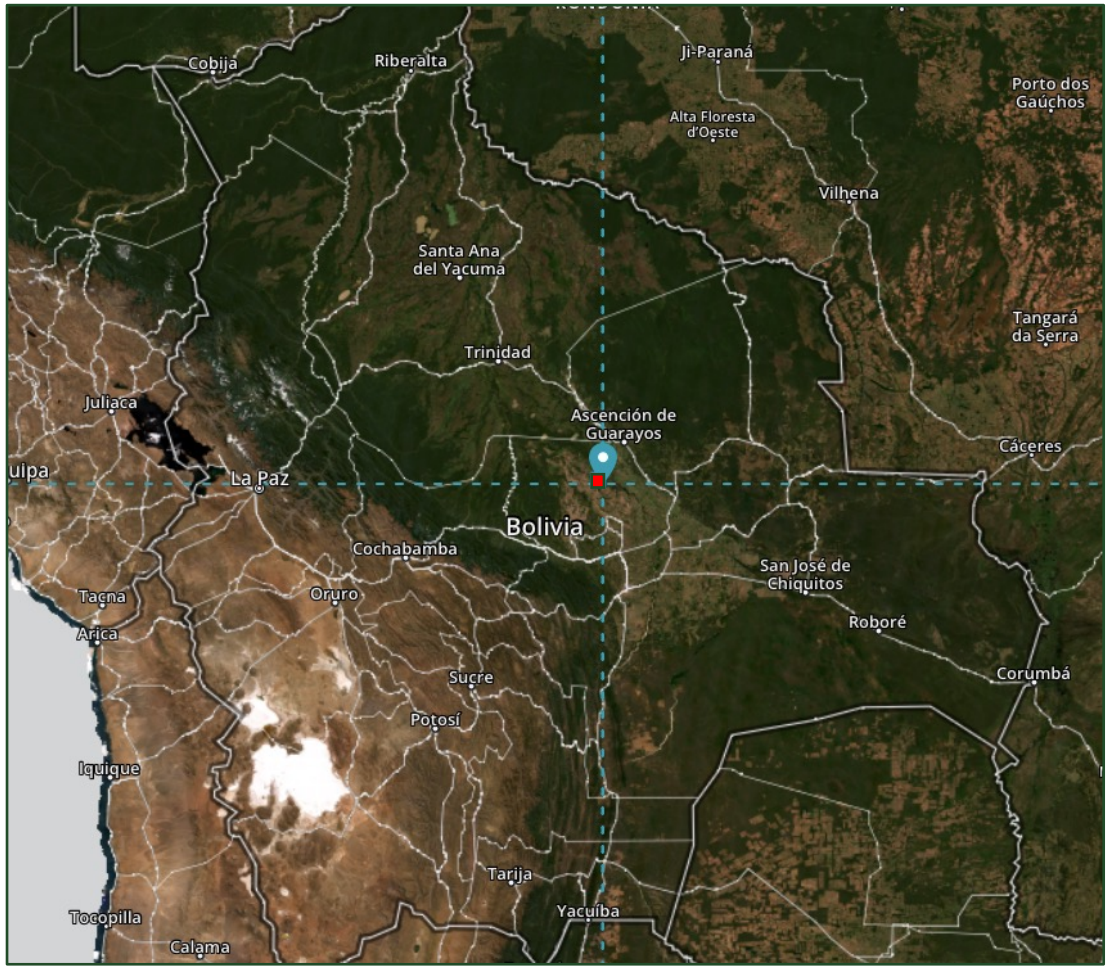


The visible area is publicly available for certain periods since 2015

1: The full database is available with a research license and limited download access per month
Source: Planet

Illustrative example: Planet provides images with greater granularity than most other public sources



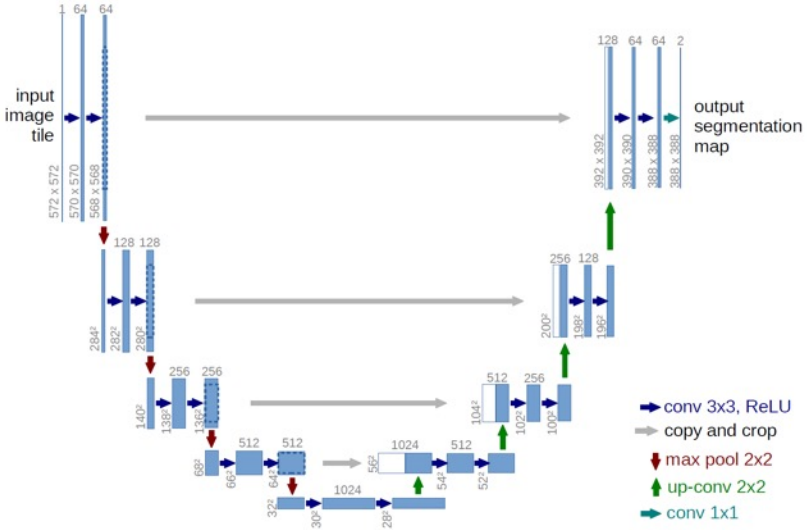


Sources: USGS Landsat EarthExplorer, Planet

A full-resolution quad is ~40 MB, and mapping the entire country for the given period would mean processing ~200 TB of data

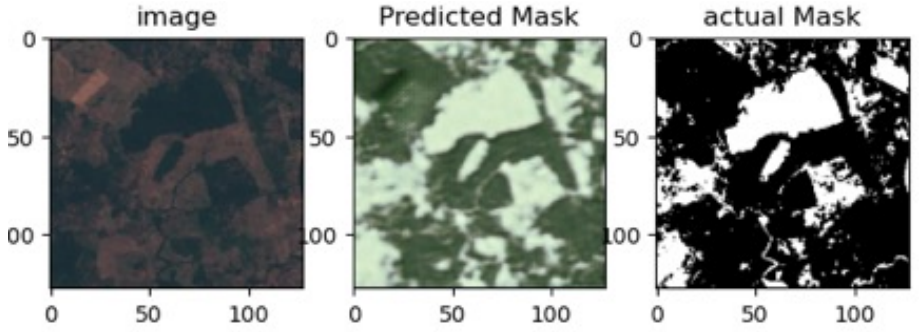
Model choices and architecture: we used a U-Net based model and trained it on two rainforest satellite datasets

The underlying UNet architecture is a Conv Encoder-Decoder often used for mask creation



- The contracting layer is constructed of 3 convolution layers, Maxpooling and ReLU are used for aggregation and non-linearity
- The expansive pathway uses up convolutions and re-combines spatial and feature information

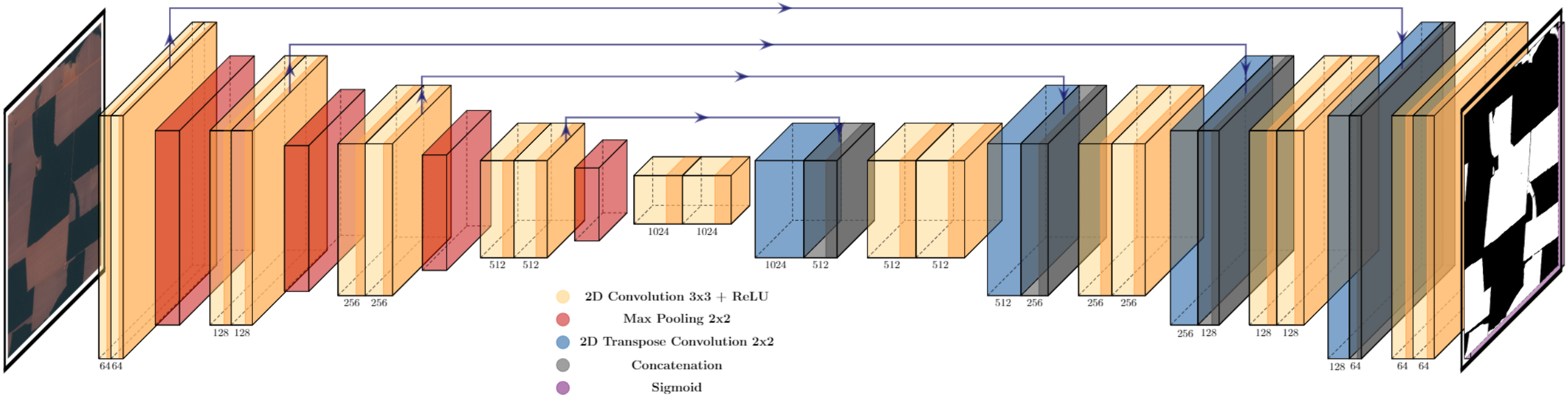
Early training results seemed promising but much tuning was needed



- The goal is to reduce spatial information and increase feature information, then re-construct a mask from the image
- We aimed for binary (forest / non-forest) classifier but future applications could be more nuanced

Source: Ronneberger et al. (2015), Bragagnolo et al. (2022)

In-depth look: U-Net



Source: John & Zhang (2022)

Modeling approach: architecture

- Type of layers (total: 23):
  - 5 Encoder blocks:
    - 2 Conv & 1 MaxPooling
    - Reduce spatial dims while increasing num of channels
  - 5 Decoding layers:
    - Conv2DTranspose, concatenation & 2 convolution
    - concatenation is to connect encoder to decoder
    - increasing spatial dimensions while decreasing the number of channels
- Other parameters:
  - Kernel initializer: he_normal
  - Activations (except final output): ReLu
  - Output activation ftn: Sigmoid
  - Parameters: 33,123,393
  - optimizer = 'Adam', lr = 0.0001
  - criterion = 'binary_crossentropy',
  - metrics = ['accuracy', 'mse']

```
# Now defining Unet
def GiveMeUnet(inputImage, numFilters = 16, droupouts = 0.1, doBatchNorm = True):
    # defining encoder Path
    c1 = Conv2dBlock(inputImage, numFilters * 1, kernelSize = 3, doBatchNorm = doBatchNorm)
    p1 = tf.keras.layers.MaxPooling2D((2,2))(c1)
    p1 = tf.keras.layers.Dropout(droupouts)(p1)

    c2 = Conv2dBlock(p1, numFilters * 2, kernelSize = 3, doBatchNorm = doBatchNorm)
    p2 = tf.keras.layers.MaxPooling2D((2,2))(c2)
    p2 = tf.keras.layers.Dropout(droupouts)(p2)

    c3 = Conv2dBlock(p2, numFilters * 4, kernelSize = 3, doBatchNorm = doBatchNorm)
    p3 = tf.keras.layers.MaxPooling2D((2,2))(c3)
    p3 = tf.keras.layers.Dropout(droupouts)(p3)

    c4 = Conv2dBlock(p3, numFilters * 8, kernelSize = 3, doBatchNorm = doBatchNorm)
    p4 = tf.keras.layers.MaxPooling2D((2,2))(c4)
    p4 = tf.keras.layers.Dropout(droupouts)(p4)

    c5 = Conv2dBlock(p4, numFilters * 16, kernelSize = 3, doBatchNorm = doBatchNorm)

    # defining decoder path
    u6 = tf.keras.layers.Conv2DTranspose(numFilters*8, (3, 3), strides = (2, 2), padding = 'same')(c5)
    u6 = tf.keras.layers.concatenate([u6, c4])
    u6 = tf.keras.layers.Dropout(droupouts)(u6)
    c6 = Conv2dBlock(u6, numFilters * 8, kernelSize = 3, doBatchNorm = doBatchNorm)

    u7 = tf.keras.layers.Conv2DTranspose(numFilters*4, (3, 3), strides = (2, 2), padding = 'same')(c6)

    u7 = tf.keras.layers.concatenate([u7, c3])
    u7 = tf.keras.layers.Dropout(droupouts)(u7)
    c7 = Conv2dBlock(u7, numFilters * 4, kernelSize = 3, doBatchNorm = doBatchNorm)

    u8 = tf.keras.layers.Conv2DTranspose(numFilters*2, (3, 3), strides = (2, 2), padding = 'same')(c7)
    u8 = tf.keras.layers.concatenate([u8, c2])
    u8 = tf.keras.layers.Dropout(droupouts)(u8)
    c8 = Conv2dBlock(u8, numFilters * 2, kernelSize = 3, doBatchNorm = doBatchNorm)

    u9 = tf.keras.layers.Conv2DTranspose(numFilters*1, (3, 3), strides = (2, 2), padding = 'same')(c8)
    u9 = tf.keras.layers.concatenate([u9, c1])
    u9 = tf.keras.layers.Dropout(droupouts)(u9)
    c9 = Conv2dBlock(u9, numFilters * 1, kernelSize = 3, doBatchNorm = doBatchNorm)

    output = tf.keras.layers.Conv2D(4, (1, 1), activation = 'sigmoid')(c9)
    model = tf.keras.Model(inputs = [inputImage], outputs = [output])
    return model
```

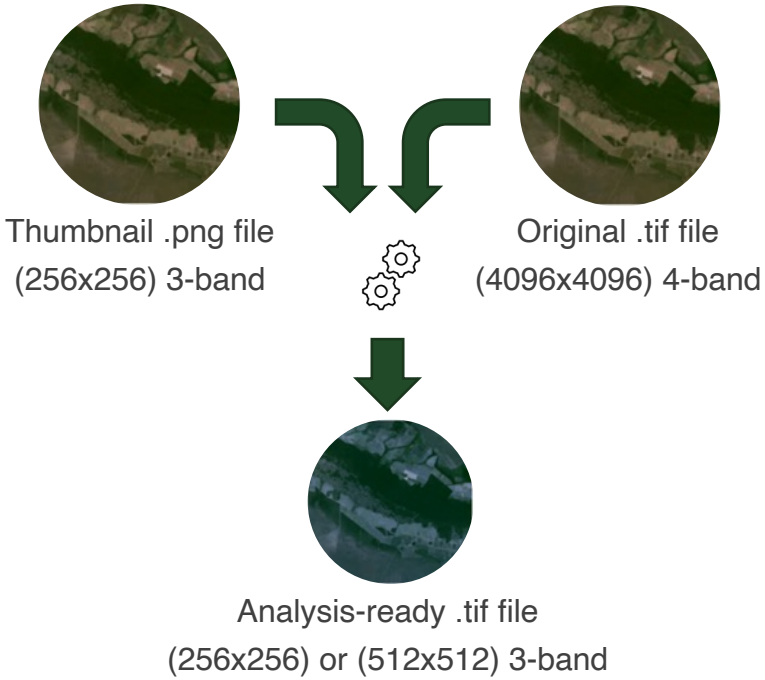
Modeling approach: helper functions, image preprocessing, image augmentation

We leveraged an existing training dataset of a similar tropical area with masks available

- 30 images to train, 15 to validate, all about the Amazonian rainforest area
- Specs: 512 x 512 pixels, 3 bands (RGB) for originals, 1 band for masks
- Normalized (/255) and change dtype to float 255
- Used image augmentation: rotation, added noise, and cropped images
- More advanced data augmentation, especially with color manipulation led to issues

Source: Planet, Bragagnolo et al. (2022)

For test data, the Planet images had to be processed to match training dataset



A new version of the model is being developed with 500 512x512 4-band training images

The task was not as easy as we assumed originally…

Original

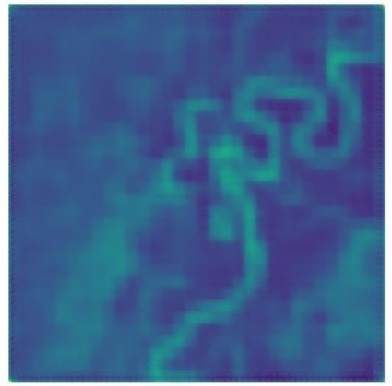
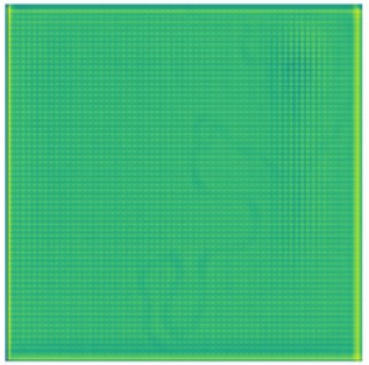
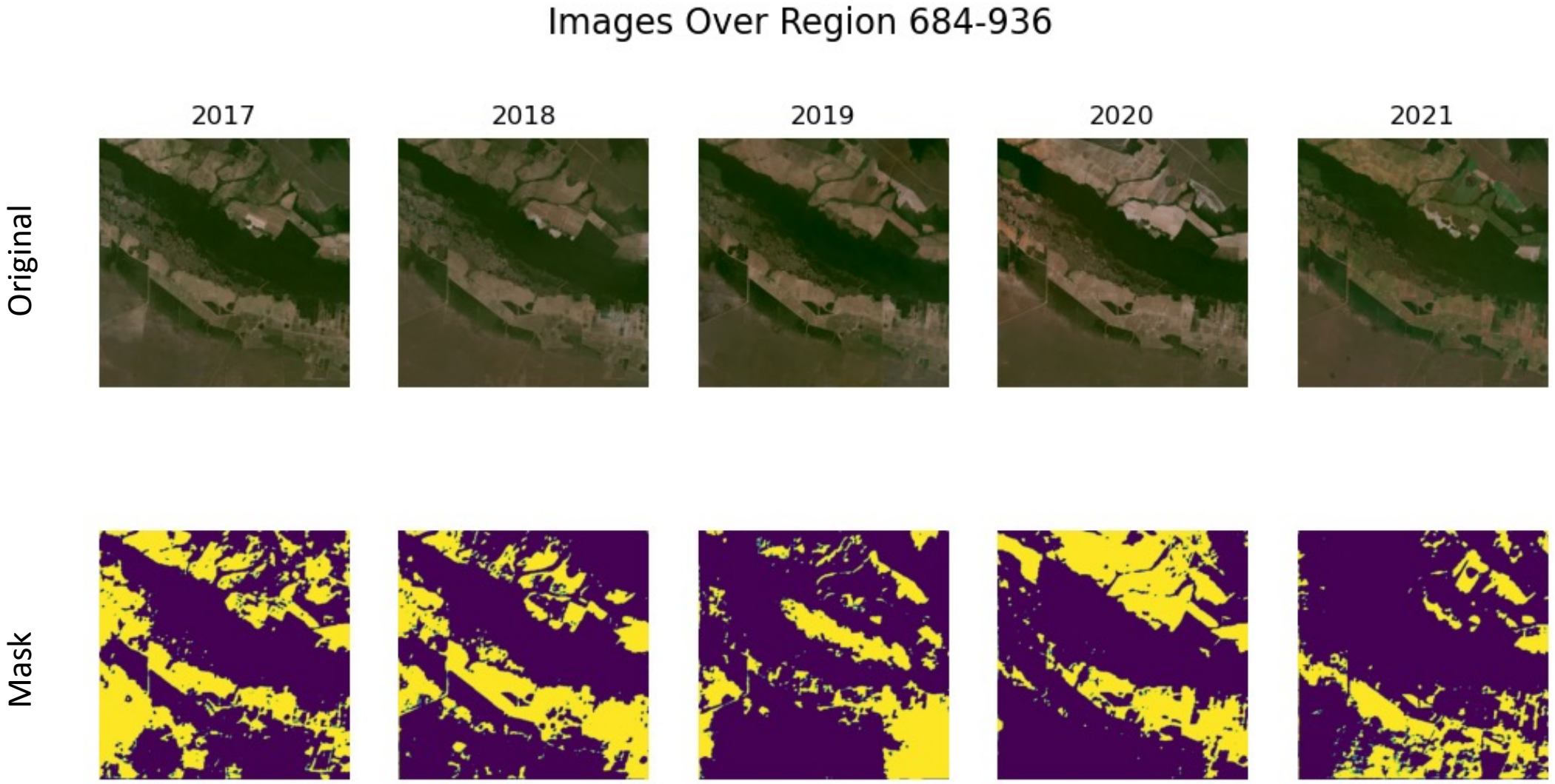


Image segmentation model (base)

U-Net based model



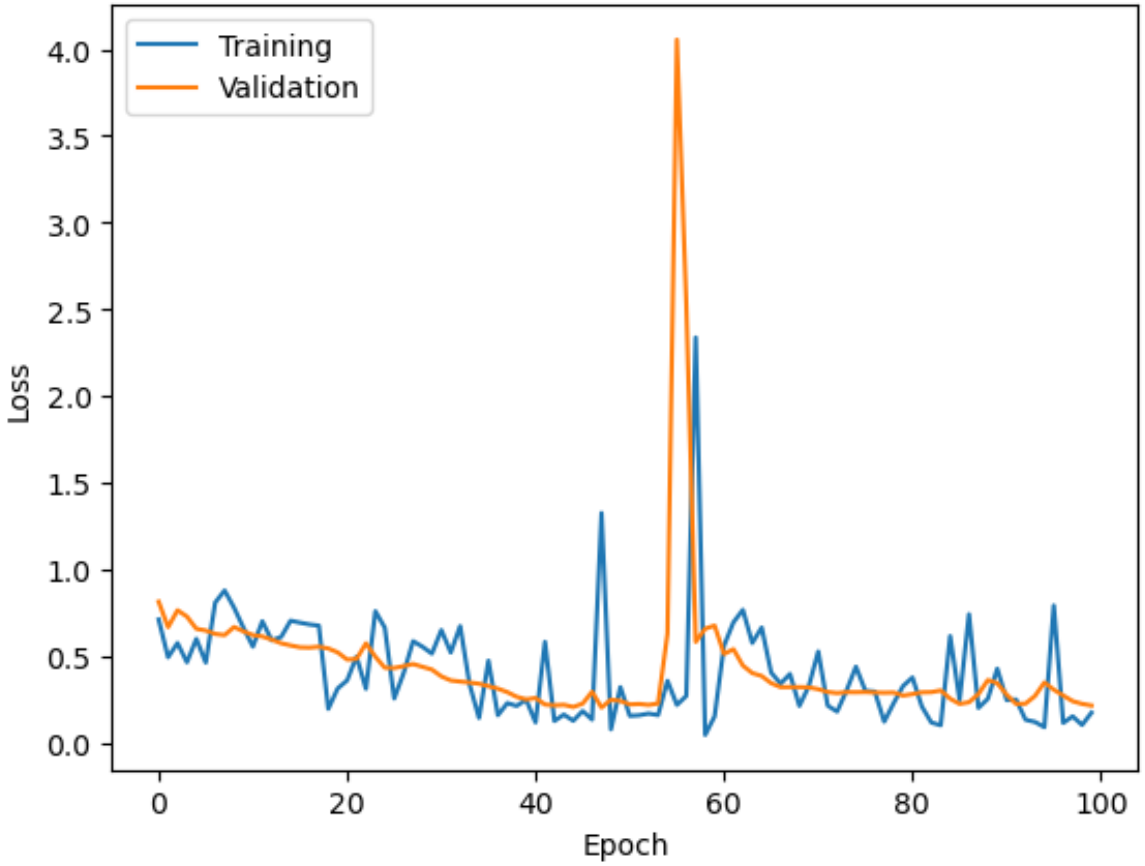
…but eventually we managed to improve the mask generation of forest / non-forest areas (CV2 viewer)



Model performed best around epoch 50 and 100

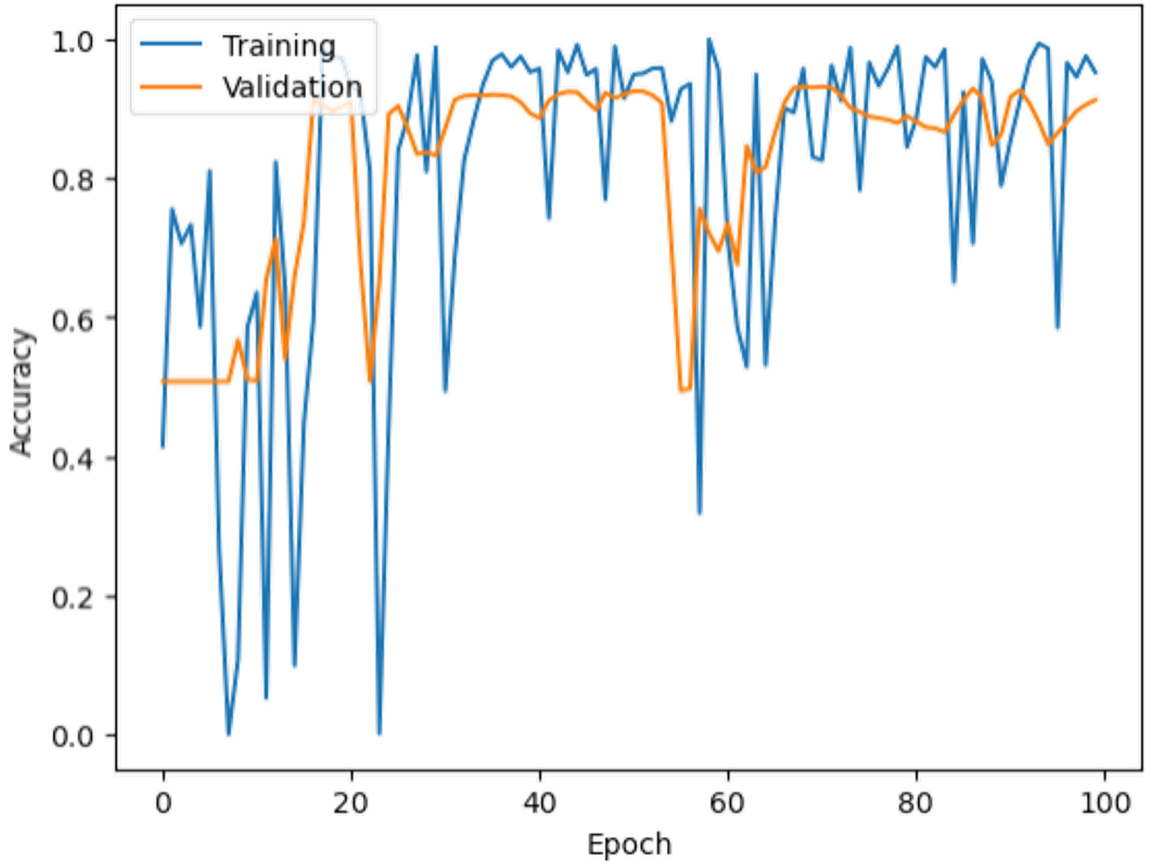
Training and validation loss

Epochs = 100



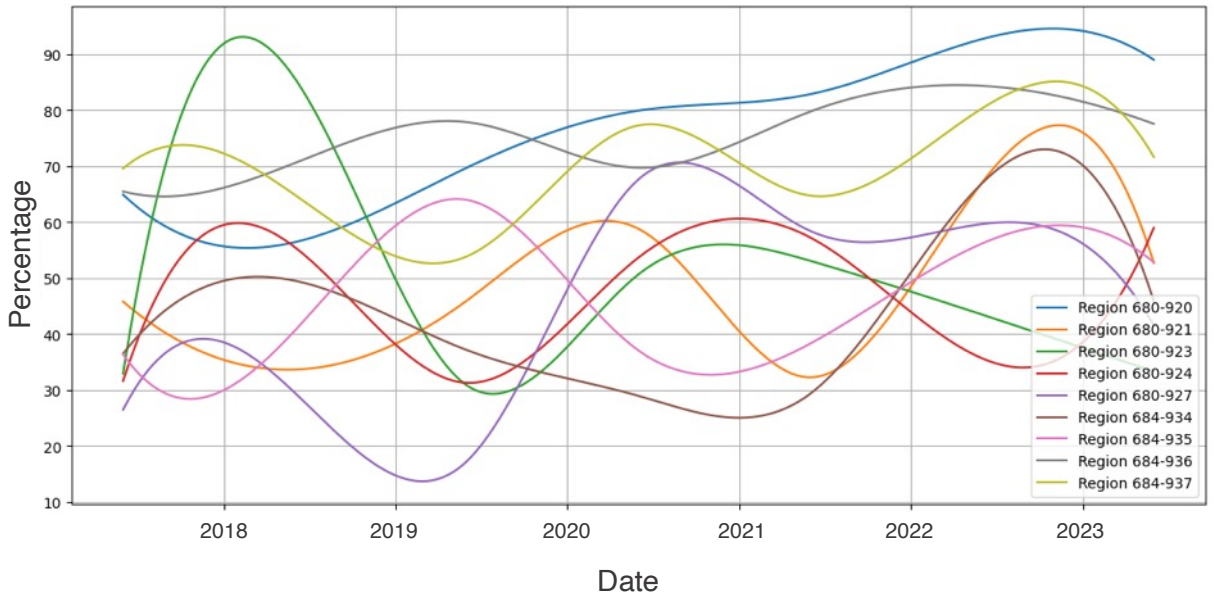
Training and validation accuracy

Epochs = 100

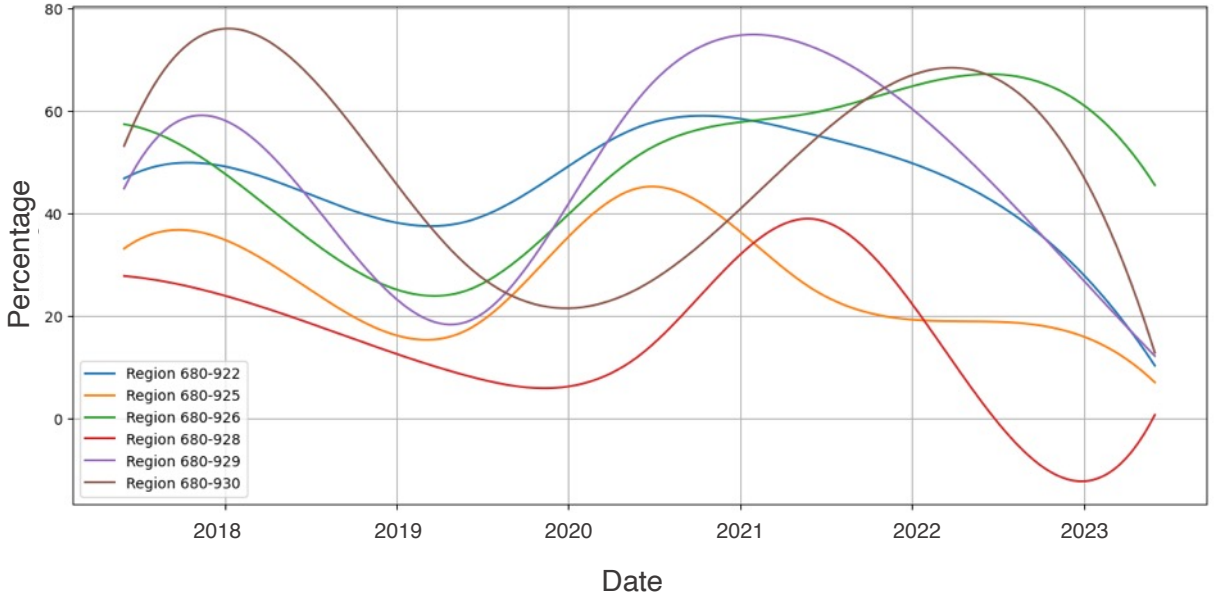


After acquiring the masked images from the test data, we calculated the percentage of forested area for comparison

Snapshot of early results: quads with increasing % of forested area over time



Snapshot of early results: quads with decreasing % of forested area over time



The most drastic difference found so far is on quad ‘680-930’, centered at 16.21° S, 63.37° W

Satellite image captured June-Dec 2016

Satellite image captured in August 2023

Source: Planet

Conclusion and finalizing the project

- Finalize training the new model capable of handling 4-band images & trained on 500 images
- Further hyperparameter tuning to adjust model to our data needs
- Finish quad-by-quad analysis and calculate more precisely the % of deforestation perceived
- Analyze the spatial aspect of deforestation

Potential future steps

Train the model on more data and test how diverse it can be while remaining accurate

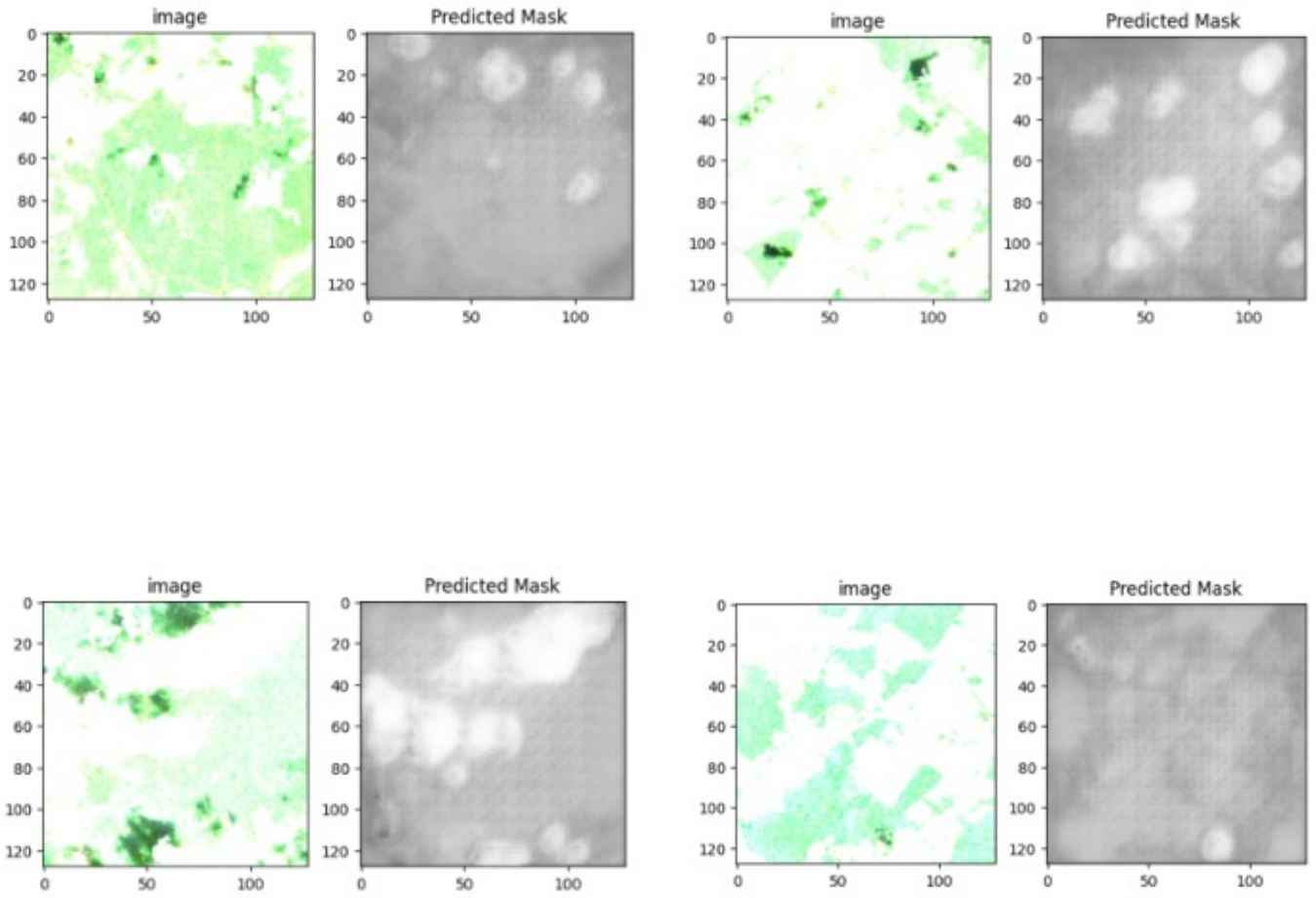
Multi-class masks could be developed to show not only deforestation, but estimate what the rest of the area is

Could be developed into a tool where users input a set of corresponding images and the model calculates deforestation

An even more advanced application would be users only inputting coordinates and the tool querying satellite images based on that

Appendix

Preliminary results of the updated U-Net model, trained on 500 images indicate further tuning is necessary



References

Sources:

- Deforestation in Bolivia has jumped by 32% in a year. What is going on? Guardian: https://www.theguardian.com/global-development/2023/oct/12/deforestation-in-bolivia-has-jumped-by-32-in-a-year-what-is-going-on
- Soy expansion drives deforestation in Bolivia. Trase insights: https://insights.trase.earth/insights/soy-expansion-drives-deforestation-in-bolivia/
- Trees alone will not save the world. The Economist: https://www.economist.com/special-report/2023/11/20/trees-alone-will-not-save-the-world
- Computer Vision Tutorial. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2019/04/introduction-image-segmentation-techniques-python/
- L. Bragagnolo, R.V. da Silva, J.M.V. Grzybowski,
  Amazon forest cover change mapping based on semantic segmentation by U-Nets, Ecological Informatics, Volume 62, 2021, 101279, ISSN 1574-9541, https://doi.org/10.1016/j.ecoinf.2021.101279.
- Ahmad Alzu'bi, Lujain Alsmadi, Monitoring deforestation in Jordan using deep semantic segmentation with satellite imagery, Ecological Informatics, Volume 70, 2022, 101745, ISSN 1574-9541, https://doi.org/10.1016/j.ecoinf.2022.101745.
- L. Bragagnolo, R.V. da Silva, J.M.V. Grzybowski, Towards the automatic monitoring of deforestation in Brazilian rainforest, Ecological Informatics, Volume 66, 2021, 101454, ISSN 1574-9541, https://doi.org/10.1016/j.ecoinf.2021.101454.
- David John, Ce Zhang, An attention-based U-Net for detecting deforestation within satellite sensor imagery, International Journal of Applied Earth Observation and Geoinformation, Volume 107, 2022, 102685, ISSN 1569-8432, https://doi.org/10.1016/j.jag.2022.102685.
- Ronneberger, O., Fischer, P., Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science(), vol 9351. Springer, Cham. https://doi.org/10.1007/978-3-319-24574-4_28

Data sources:

- Planet: https://developers.planet.com
- USGS Landsat: https://landsatlook.usgs.gov
- Amazon and Atlantic Forest image datasets for semantic segmentation. https://zenodo.org/records/4498086 and https://zenodo.org/records/3233081

All pictures and icons were sourced via Microsoft PowerPoint Stock Images (apart from the ones sourced from the indicated data sources).