

Halil İbrahim Taşkömür

Mail : taskomurhalilibrahim@gmail.com

linkedin.com/in/hitaskomur

Phone number : +90 544 410 67 71

github.com/hitaskomur

Introduction :

- Purpose of Project-----	2
- Dataset-----	2
- Deep Learning Model-----	3
- Improvements-----	6
- Conclusions-----	8

Purpose of Project:

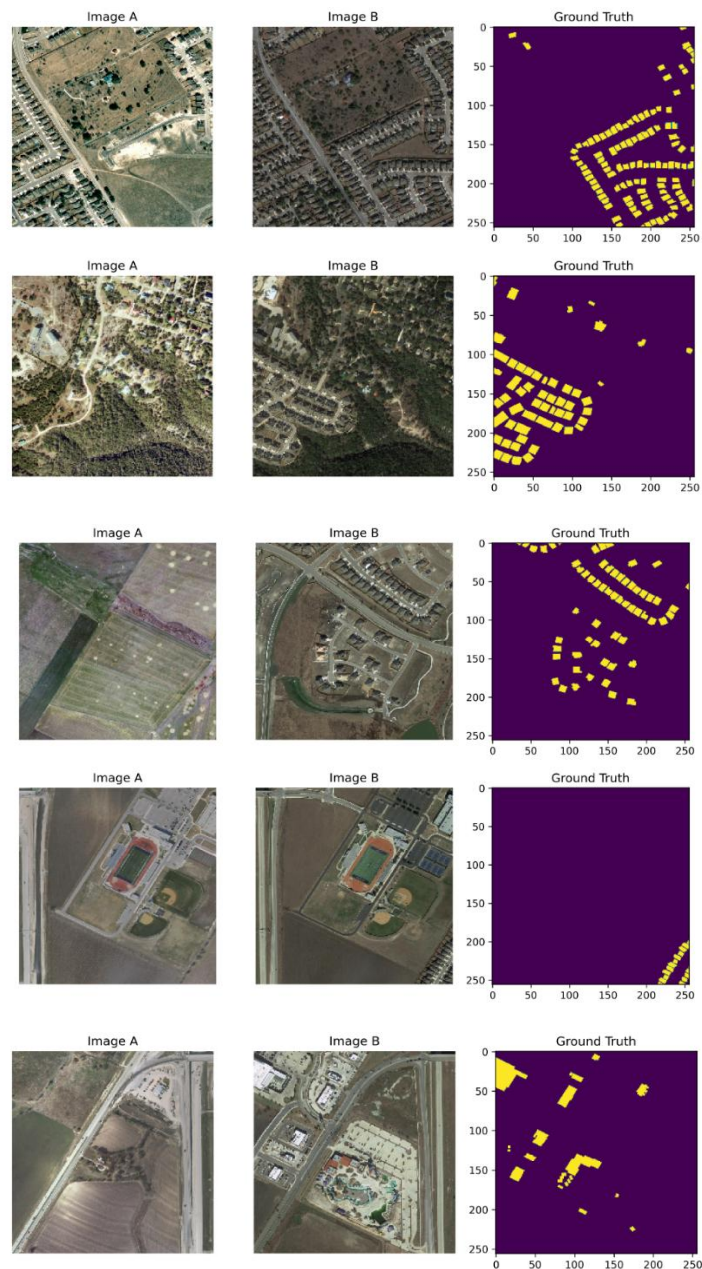
-Detecting differences between satellite images taken at two different times.

Dataset:

-[Used LEVIR_CD Dataset](#).

- LEVIR-CD consists of 637 very high-resolution (VHR, 0.5m/pixel) Google Earth (GE) image patch pairs with a size of 1024×1024 pixels. These bitemporal images with time span of 5 to 14 years have significant land-use changes, especially the construction growth. LEVIR-CD covers various types of buildings, such as villa residences, tall apartments, small garages and large warehouses.

Dataset Samples



Deep Learning Model:

Before providing the dataset with input images, we performed several preprocessing operations. We resized each image to 256. We processed the first and last images in the dataset as RGB (3-channel). After merging the two RGB images, we fed them to the model as input. The model now has a 6-channel input ($\text{RGB}(3) + \text{RGB}(3) = 6$).

In other deep learning models, such as Siamese Unet, two images are fed into the model without being merged, and each image is processed separately. This requires significant resources. **Our model achieved success with fewer resources.**

A Fully Connected Neural Network algorithm was used. It comprised input, encoder, decoder, and output layers.

The train dataset, included in the dataset, was used for model training, the validation dataset for validation, and the test dataset for testing.

The activation functions within the model were "relu," "sigmoid (in the output layer)," the "same" parameter for padding, and the "bilinear" parameter for interpolation in the upsampling layers.

To make model analysis more meaningful, in addition to commonly used metrics (accuracy), the IoU (Intersection of Unions: a performance measure used to evaluate the accuracy of segmentation and object detection algorithms) metric was used.

Although different techniques and parameters were tested, this model was selected as the most cost-effective model (U-Net, Siamese-U-Net, augmentation data, etc.).

Model Architecture

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import InputLayer, Conv2D, MaxPooling2D, UpSampling2D
from tensorflow.keras.optimizers import Adam

def build_model(input_shape=(IMG_SIZE, IMG_SIZE, 6)):
    model = Sequential()
    model.add(InputLayer(input_shape=input_shape))

    # Encoder
    model.add(Conv2D(32, (3,3), activation='relu', padding='same'))
    model.add(MaxPooling2D((2,2)))

    model.add(Conv2D(64, (3,3), activation='relu', padding='same'))
    model.add(MaxPooling2D((2,2)))

    # Decoder - Conv2DTranspose yeriine UpSampling + Conv2D
    model.add(UpSampling2D(size=(2,2), interpolation='bilinear'))
    model.add(Conv2D(64, (3,3), activation='relu', padding='same'))

    model.add(UpSampling2D(size=(2,2), interpolation='bilinear'))
    model.add(Conv2D(32, (3,3), activation='relu', padding='same'))

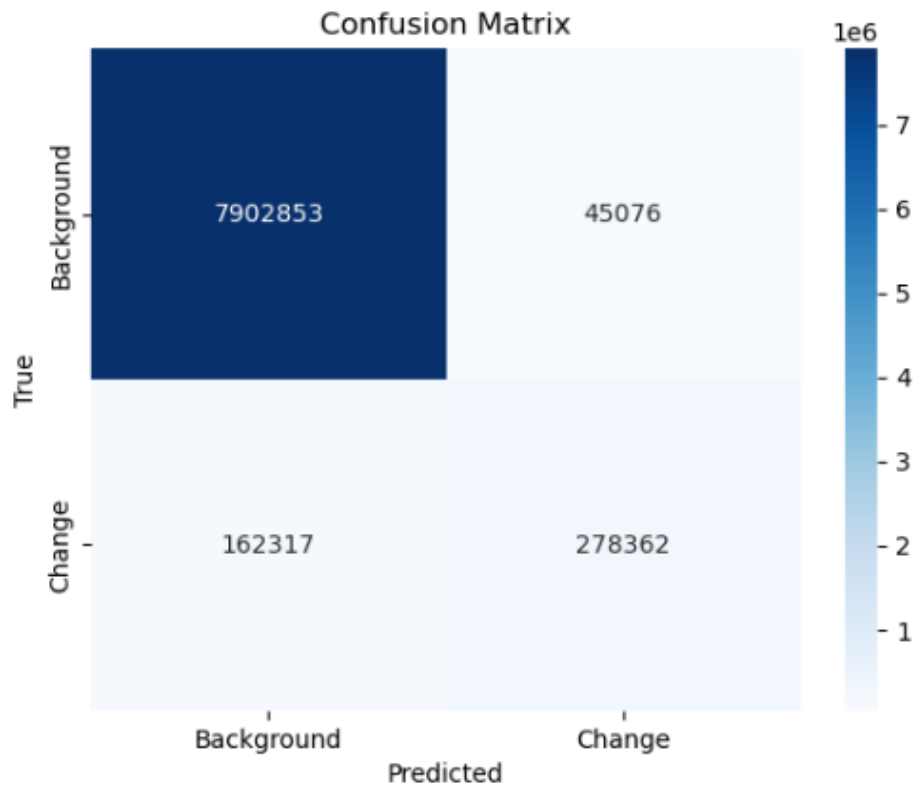
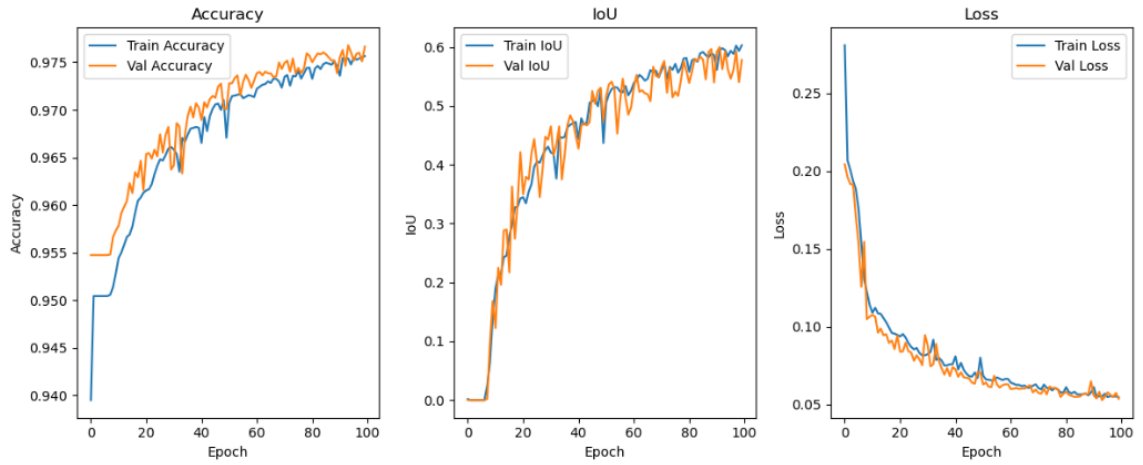
    # Last Layer (Predict Mask)
    model.add(Conv2D(1, (1,1), activation='sigmoid', padding='same'))

    model.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accuracy', iou_metric])
    return model
base_model = build_model()
base_model.summary()
```

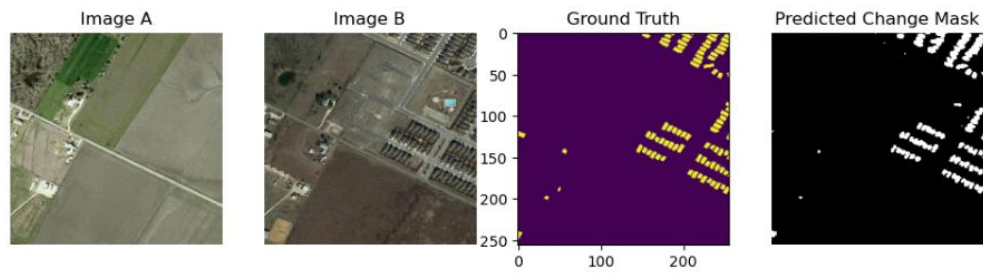
Model: "sequential"

Layer (type)	Output Shape	Param #
=====		
conv2d (Conv2D)	(None, 256, 256, 32)	1760
max_pooling2d (MaxPooling2D)	(None, 128, 128, 32)	0
conv2d_1 (Conv2D)	(None, 128, 128, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 64)	0
up_sampling2d (UpSampling2D)	(None, 128, 128, 64)	0
conv2d_2 (Conv2D)	(None, 128, 128, 64)	36928
up_sampling2d_1 (UpSampling2D)	(None, 256, 256, 64)	0
conv2d_3 (Conv2D)	(None, 256, 256, 32)	18464
conv2d_4 (Conv2D)	(None, 256, 256, 1)	33
=====		
Total params: 75,681		
Trainable params: 75,681		
Non-trainable params: 0		

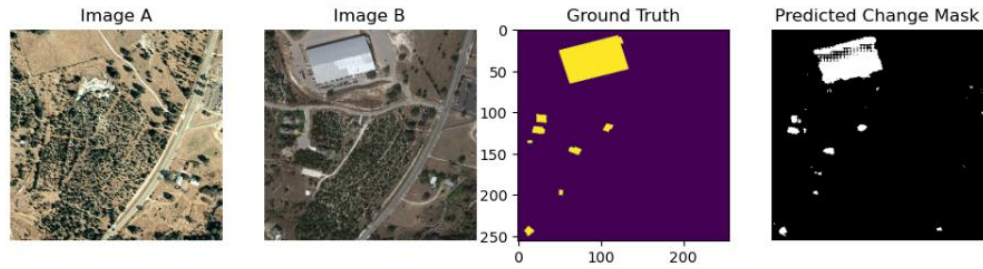
Model Parameters Results



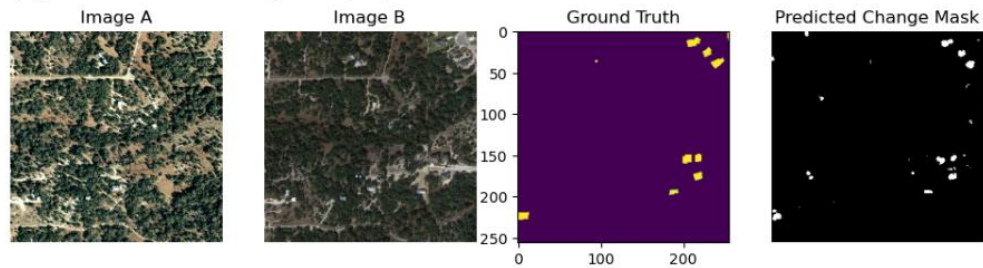
Visualizing Test Samples



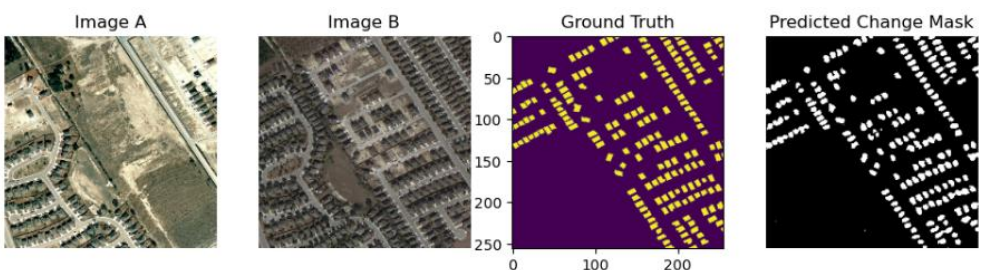
1/1 [=====] - 0s 32ms/step



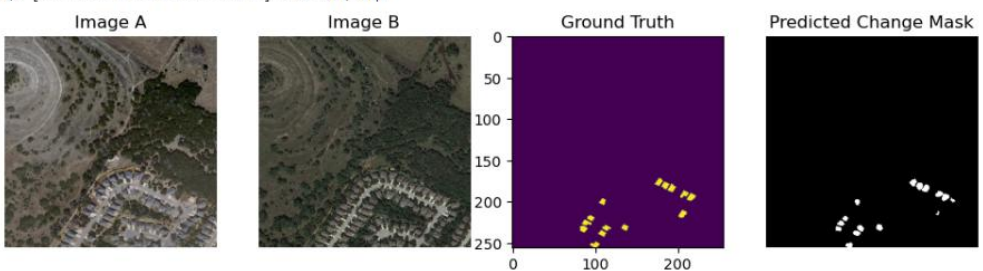
1/1 [=====] - 0s 30ms/step



1/1 [=====] - 0s 26ms/step



1/1 [=====] - 0s 26ms/step



Test Evaluation Scores

-Test Loss: 0.0573, -Test Accuracy: 0.9733, -Test IoU: 0.6190

-Precision: 0.8413579301898346, -Recall: 0.6983813614898826,

-F1 Score: 0.7632314119168575

Improvements:

- Created UI web page with streamlit

Change Detection with Satellite Images

Timezone 1 Image (T1)

Drag and drop file here
Limit 200MB per file • PNG, JPG, JPEG

Browse files

test_20.png 2.4MB

Timezone 2 Image (T2)

Drag and drop file here
Limit 200MB per file • PNG, JPG, JPEG

Browse files

test_20.png 2.3MB

Loaded Images




image T1 image T2



Change Detection (Mask)



- Created Api by fastapi.

default

POST /predict Predict

Parameters

No parameters

Request body **required**

multipart/form-data

image1 **required**
string(binary)

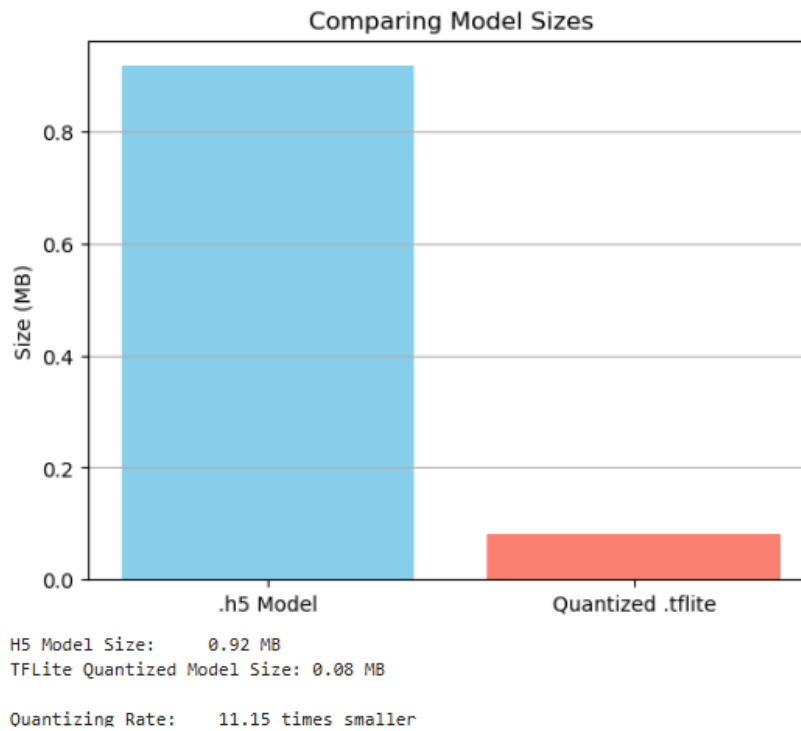
image2 **required**
string(binary)

Execute Clear

- Quantization (Size reduction has been made so that it can be used on hardware with low resources (mobile, embedded systems, etc.)

◆ Keras Model:
Precision: 0.7926137284783243
Recall: 0.7562726232094793
F1 Score: 0.7740168450383128

◆ Quantized TFLite Model:
Precision: 0.7991544736929022
Recall: 0.7485691303406243
F1 Score: 0.773035143769968



Conclusions:

- More detailed fine-tuning can be done for specific areas or structures in the model.
- Model results can be re-evaluated with data augmentation.
- Improvements can be made based on the area to be used.