Satellite Remote Sensing Image -RSI-CB256

Satellite image Classification Dataset-RSI-CB256

Dep. computer Engineering (AASTMT)
Arab Academy for science and technology.
Alexandria, Egypt

Introduction

The analysis of remote sensing (RS) images and its numerous applications have made significant strides in recent years. There is a growing need for the automatic interpretation of RS photos as they are now more readily available than previously. The benchmark datasets are crucial building blocks for creating and evaluating clever interpretation algorithms in this situation. This article discusses the issue of how to quickly create a good benchmark dataset for RS image interpretation after examining the existing benchmark datasets in the research community of RS image interpretation. To be more precise, using bibliometric analyses, we first assess the difficulties encountered when creating intelligent algorithms for RS picture interpretation. The general advice on producing benchmark datasets effectively is then presented.

In addition to the offered guidelines, we provide an example of how to develop an RS image dataset, namely Million-AID, a new large-scale benchmark dataset containing a million examples for RS image scene classification. Finally, certain obstacles and opportunities in RS picture annotation are highlighted to aid research in benchmark dataset construction. We expect that our study will give the RS community a broader view on creating large-scale and useful image collections for future research, particularly data-driven ones.

Related Work

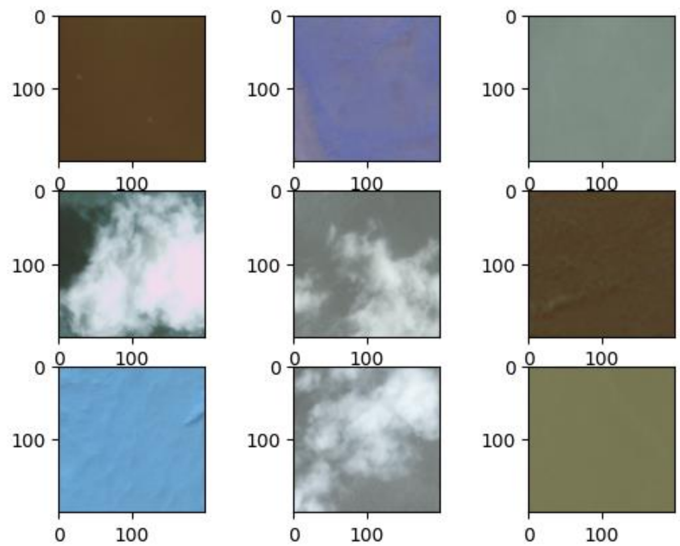
Datasets with Annotations for RS Image Interpretation The interpretation of RS pictures has become increasingly relevant in a wide range of applications, attracting significant academic attention. As a result, numerous datasets have been created to aid in the development of RS image interpretation algorithms. We provide a systematic evaluation of existing RS image datasets concerning the current mainstream of RS image interpretation tasks, such as scene categorization, object detection, semantic segmentation, and change detection, using material published over the last decade.

Satellite image Classification Dataset-RSI-CB256 classes

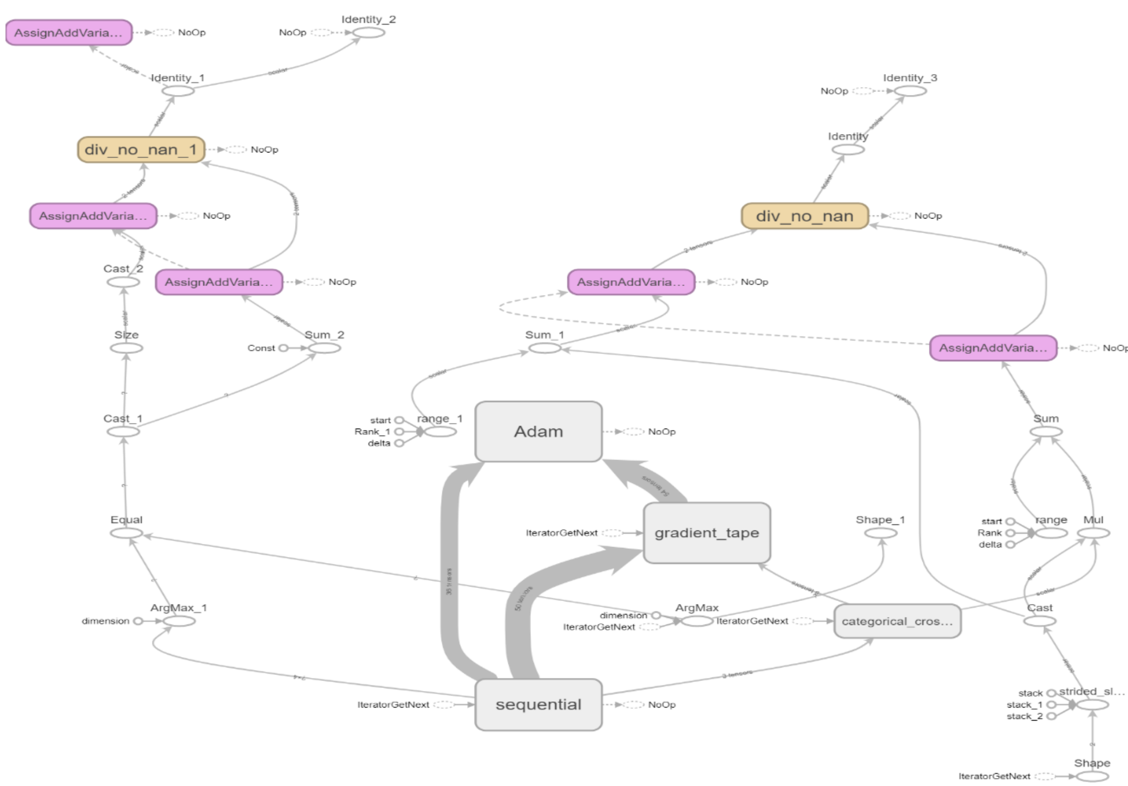
Classes	Size
Cloudy	1500 image
Desert	1131 image
Green-area	1500 image
Water	1500 image

Proposed Model & Experimental Work – Hand-Crafted Model

Dataset Sample



Model Graph



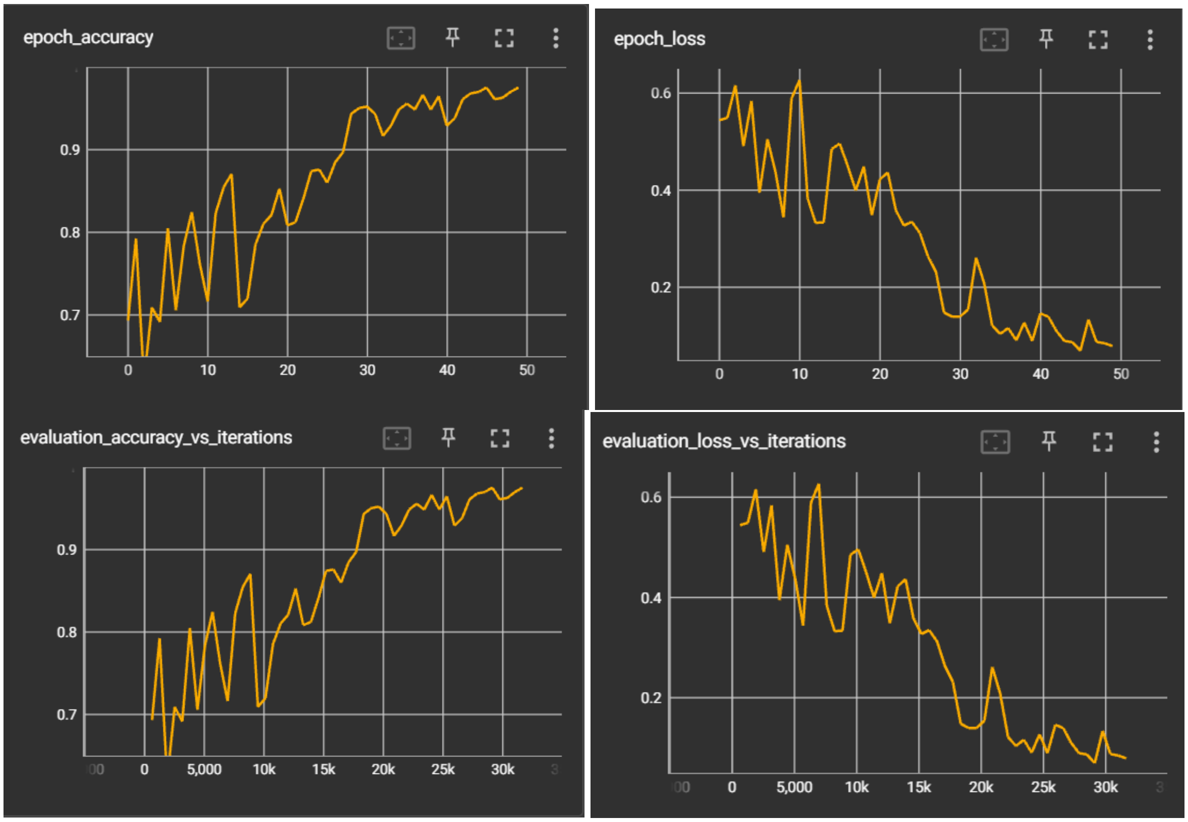
Keras Model structure

```
model = Sequential()

model.add(Conv2D(96, (3, 3), padding = 'same', input_shape=(200, 200, 3)))
model.add(Activation('relu'))
model.add(Conv2D(96, (3, 3), padding = 'same'))
model.add(Activation('relu'))
model.add(Conv2D(96, (3, 3), padding = 'same', strides = (2,2)))
model.add(Dropout(0.5))

model.add(Conv2D(192, (3, 3), padding = 'same'))
model.add(Activation('relu'))
model.add(Conv2D(192, (3, 3), padding = 'same'))
model.add(Activation('relu'))
model.add(Conv2D(192, (3, 3), padding = 'same', strides = (2,2)))
model.add(Dropout(0.5))

model.add(Conv2D(192, (3, 3), padding = 'same'))
model.add(Activation('relu'))
model.add(Conv2D(192, (1, 1), padding = 'valid'))
model.add(Activation('relu'))
model.add(Conv2D(4, (1, 1), padding = 'valid'))

model.add(GlobalAveragePooling2D())
model.add(Activation('sigmoid'))
```

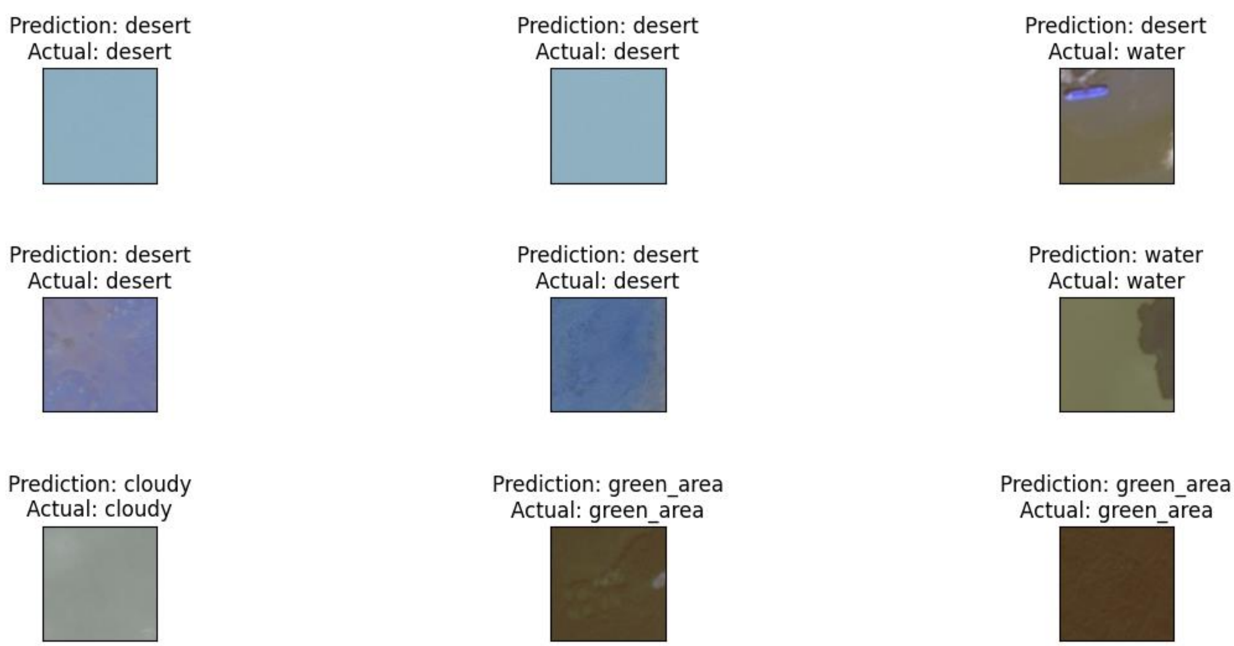
Model summary

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 200, 200, 96)      2688

 activation (Activation)     (None, 200, 200, 96)      0

 conv2d_1 (Conv2D)           (None, 200, 200, 96)      83040

 activation_1 (Activation)   (None, 200, 200, 96)      0

 conv2d_2 (Conv2D)           (None, 100, 100, 96)      83040

 dropout (Dropout)           (None, 100, 100, 96)      0

 conv2d_3 (Conv2D)           (None, 100, 100, 192)     166080

 activation_2 (Activation)   (None, 100, 100, 192)     0

 conv2d_4 (Conv2D)           (None, 100, 100, 192)     331968

 activation_3 (Activation)   (None, 100, 100, 192)     0

 conv2d_5 (Conv2D)           (None, 50, 50, 192)       331968

 dropout_1 (Dropout)         (None, 50, 50, 192)       0

 conv2d_6 (Conv2D)           (None, 50, 50, 192)       331968

 activation_4 (Activation)   (None, 50, 50, 192)       0

 conv2d_7 (Conv2D)           (None, 50, 50, 192)       37056

 activation_5 (Activation)   (None, 50, 50, 192)       0

 conv2d_8 (Conv2D)           (None, 50, 50, 4)         772

 global_average_pooling2d (G  (None, 4)                0
 lobalAveragePooling2D)

 activation_6 (Activation)   (None, 4)                 0

=================================================================
Total params: 1,368,580
Trainable params: 1,368,580
Non-trainable params: 0
_________________________________________________________________
```

Model Evaluation Metrics

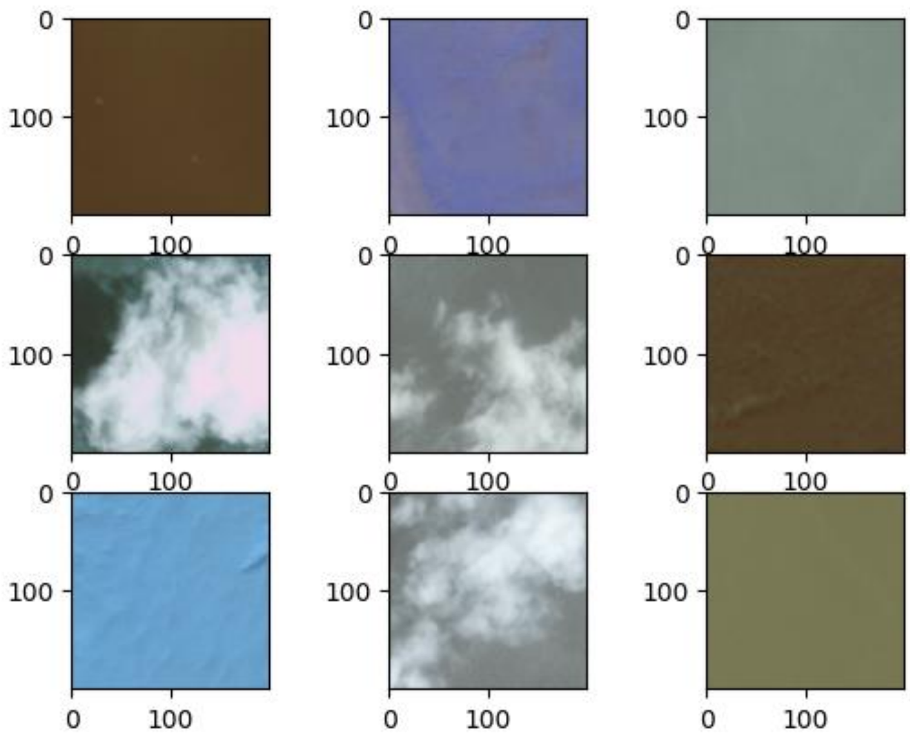


Model Results



Proposed Model & Experimental Work – TF Model

Dataset Sample



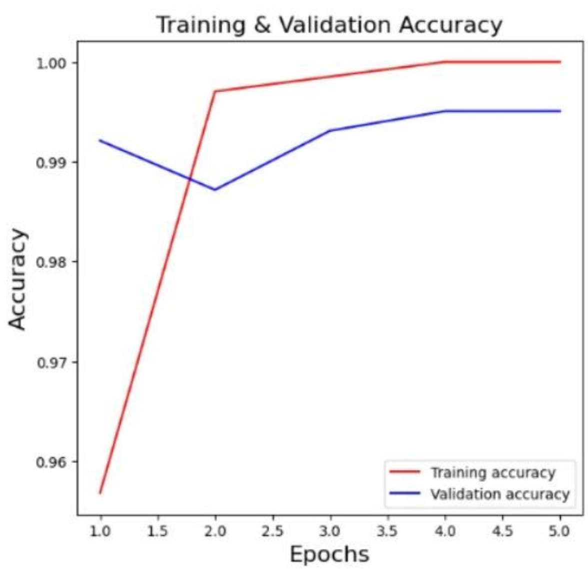
Model structure

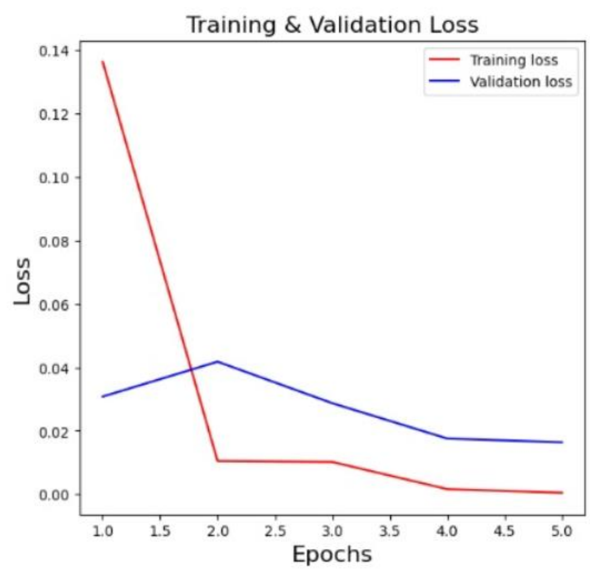
```
flatten_layer = layers.Flatten()
dense_layer_1 = layers.Dense(50, activation='relu')
dense_layer_2 = layers.Dense(20, activation='relu')
prediction_layer = layers.Dense(4, activation='softmax')


model = models.Sequential([
    base_model,
    flatten_layer,
    dense_layer_1,
    dense_layer_2,
    prediction_layer])
```

Model Summary

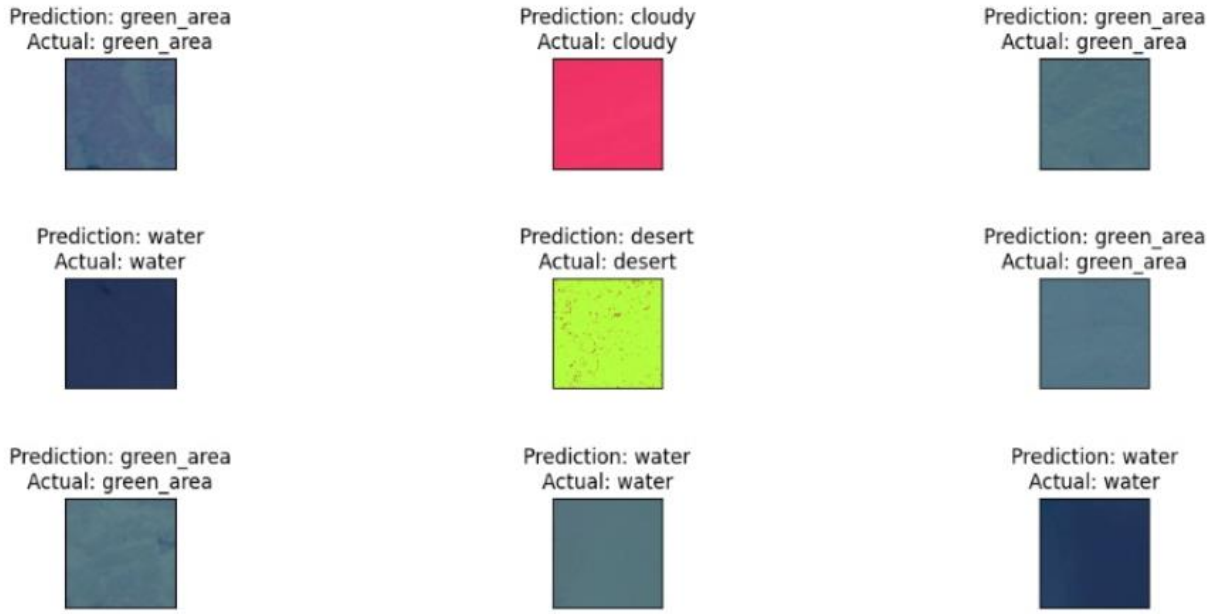
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 6, 6, 512)         14714688

 flatten (Flatten)           (None, 18432)             0

 dense (Dense)               (None, 50)                921650

 dense_1 (Dense)             (None, 20)                1020

 dense_2 (Dense)             (None, 4)                 84

=================================================================
Total params: 15,637,442
Trainable params: 922,754
Non-trainable params: 14,714,688
_________________________________________________________________
```

Model Evaluation Metrics





Model Results



Conclusion

In conclusion both models are quite accurate but TF model needs much less computation power. Training the hand-crafted model required extensive GPU performance, with multiple training trials crashing mid-session. This is most probably due to the intrinsic properties of the dataset images, as well as possible outlier corrupt images in the original files. However, once trained, the model provides satisfactory results that could easily be extrapolated upon for future use.

References

Remote Sensing: Models and Methods for Image Processing
By Robert A. Schowengerdt

https://books.google.com.eg/books?hl=en&lr=&id=KQXNaDH0X-IC&oi=fnd&pg=PP1&dq=Satellite+Remote+Sensing+Image+classification+-RSI-CB256&ots=sonWEOCaQC&sig=tr4NVPJ_UcVsj1ovF9qmjHQE5ts&redir_esc=y#v=onepage&q&f=false

Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification
By Emmanuel Maggiori; Yuliya Tarabalka; Guillaume Charpiat; Pierre Alliez

https://ieeexplore.ieee.org/abstract/document/7592858/authors#authors