SAR DATA CLASSIFICATION

Classification of SAR data by referring optical data using
Techniques of deep learning

Submitted by :
Jitendra mouli ,
Aerospace department ,
National institute of advance science ,
IISc,banglore,karnataka.

Objective :

To classify the some regions (land ,trees , water bodies) between tiptur and tumkur areas

What is SAR data ?

- Synthetic aperture radar(SAR) is the full form of radar that is used to Create two dimensional images or three dimensional images of objects such as landscapes

- SAR uses the motion of radar antenna over a target region to provide a finer spatial resolution

- As SAR device on board aircraft or spacecraft moves, the relative location of antenna and target changes with time Signal processing of the successive recorded radar echoes allows the combining of the recordings from these multiple antenna positions. This process forms the *synthetic antenna aperture* and allows the creation of higher-resolution images .

Importance of SAR data:

- Mainly, two types of satellite data SAR (radar) data and optical (MSI) data.

- Optical data is captures visible bands just like human eye

- The biggest advantages of SAR data(synthetic radar aperture data) over optical data is that it is not effected by the weather conditions and it can see through clouds and to some degree vegetation

- So, when we want to see even through the clouds and when you want to emphasis on roughness of the surfaces

Where the data comes from ?

- Data is provided by the ESA (European space agency) from the part of sentinel 1 mission satellite (SAR) data and sentinel 2 mission satellite(optical) data

- Sentinel 1 is imaging radar mission satellite providing continuous all weather day and night imagery at C band (4-8GHz)

- Sentinel 2 is multi spectral instrument mission which provide high resolution images of target region in visible spectrum

Preprocessing of SAR

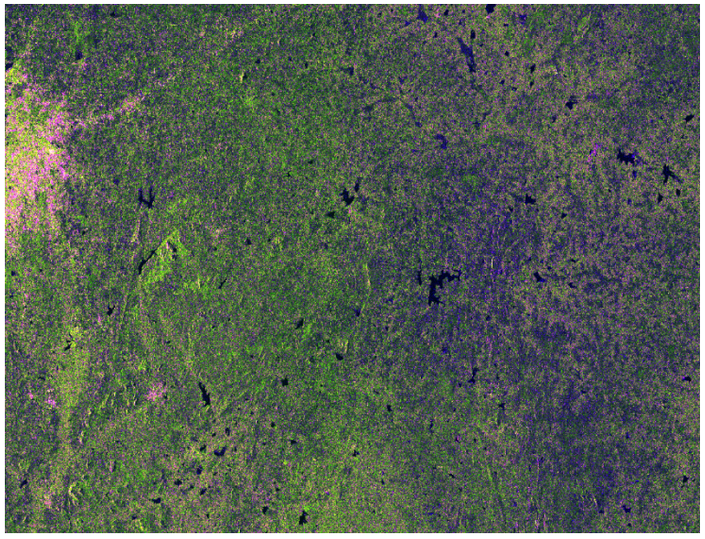
- Before going to build a machine learning model we should apply some preprocessing tasks to SAR data

- The product or data we get is raw data. So, it should be calibrated ,speckle filtered and terrain corrected

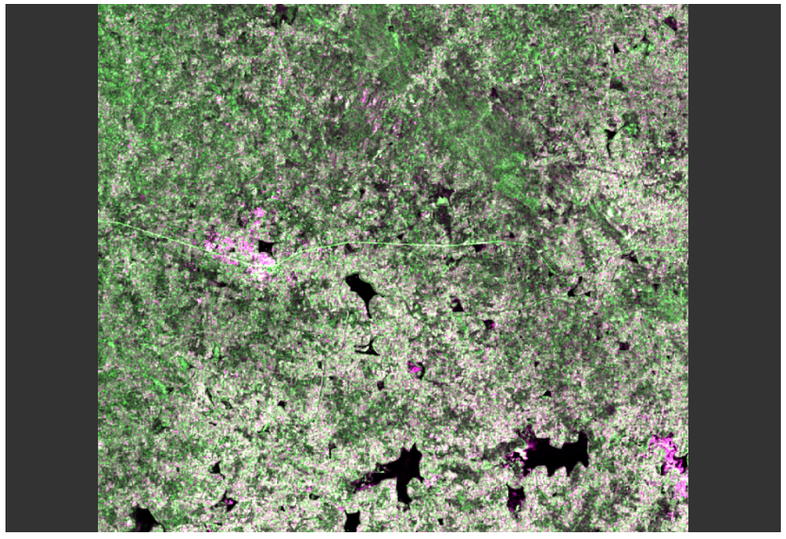
- Steps of preprocess the SAR data
  - Radiometric calibration
  - Speckle filtering
  - Range doppler terrain correction

- **Radiometric calibration** is to provide stability and accuracy. Though uncalibrated SAR imagery is sufficient for qualitative use, calibrated SAR imagery is essential for quantitative use of SAR data
- SAR images have inherent texturing called **speckles** (granular noise) which degrade the quality of the image and make interpretation of features more difficult. This process applies to reduce the speckles in the product
- Due to topographical variations of a scene and the tilt of the satellite sensor, distances can be distorted in the SAR images.**Terrain corrections** are intended to compensate for these distortions so that the geometric representation of the image will be as close as possible to the real world.

After Completion of above steps:



Original SAR image

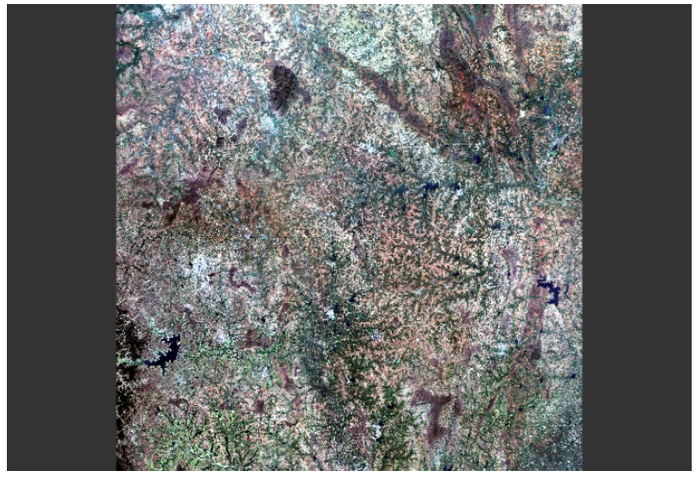


Pre-processed SAR image

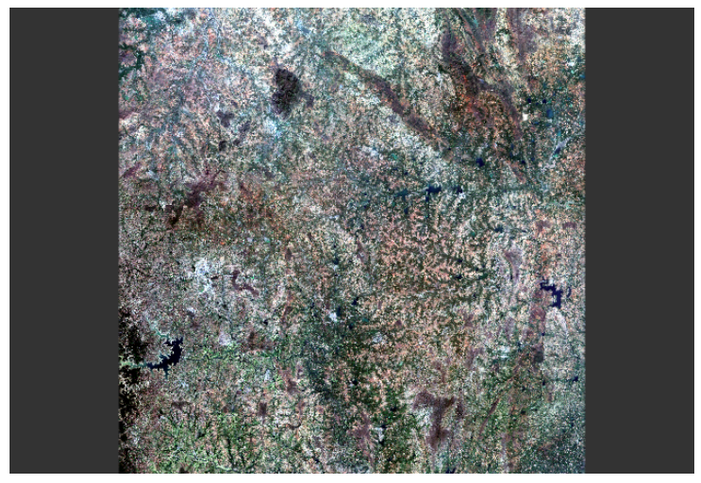
All the preprocessing steps are done using external tool called SNAP (sentinel application platform)

Preprocessing of the Optical Data:

- Since all 12 bands of the product are of different resolutions and sizes, it needs to be resampled into a single resolution and a single sized product.



Obtained Optical image product



Corresponding resampled product

All the preprocessing steps are done using external tool called SNAP (sentinel application platform)

- The SAR and optical data must contain the same geographical area in order for them to be used in the neural network as input and ground truth.
- The exported optical image must then be annotated into the respective classes so that it can be used as ground truth.

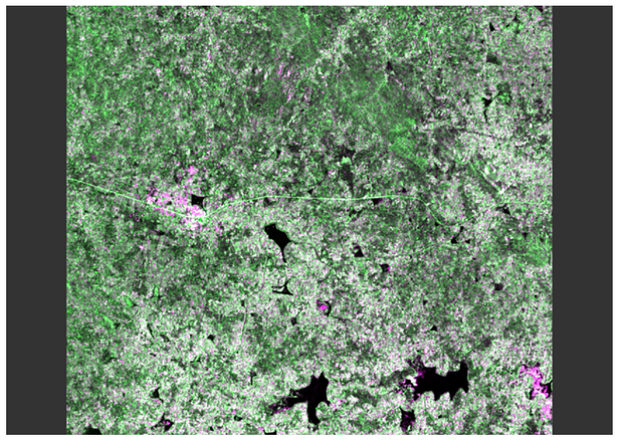
Creating Subsets:

- Subsets of both SAR and optical product must be created so that they have roughly the same geographical area.

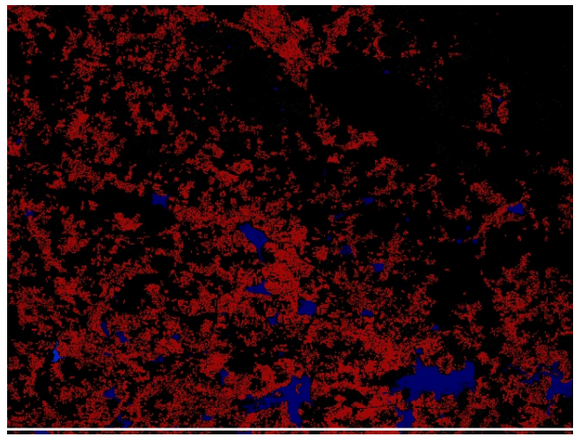
Geo referencing two layers can be easily done by an external tool called ARCGIS (ARCMap)

Annotation:

- The ground truth requires annotation to label the area into respective classes, in this case:Trees, Lakes, land(others).



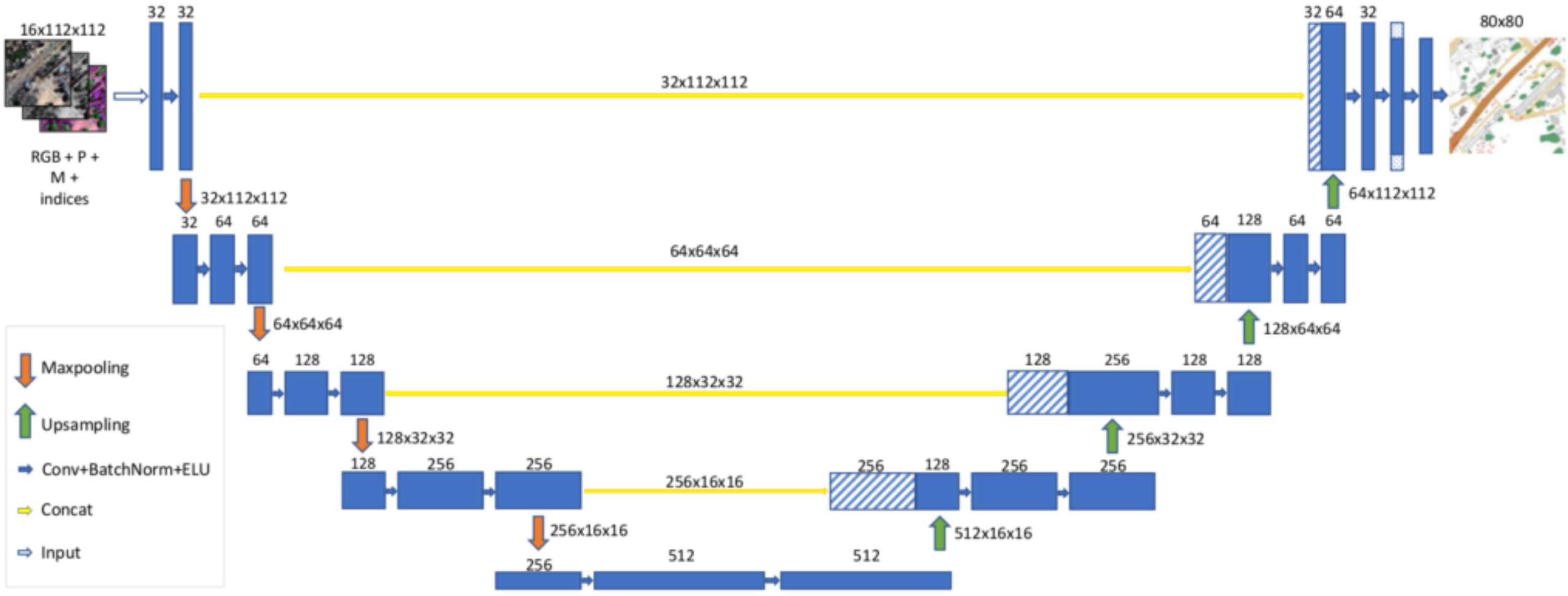
Collocated image



Annotated Image

All the annotation is done by use external tool called GIMP GNU image manipulation program

Model U net architecture:



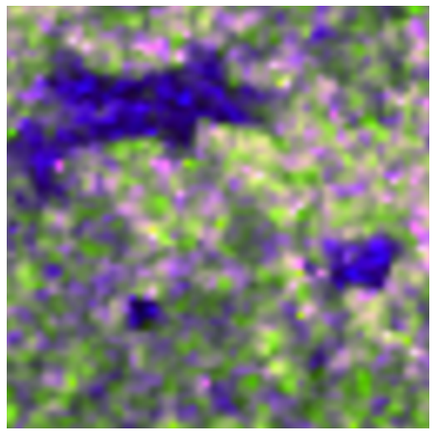
Explanation of architecture:

- The architecture looks like a ‘U’ which justifies its name.This architecture consists of three sections: the contraction, The bottleneck and the expansion section
- The contraction section is made of many contraction blocks ,each block takes an input and applies to convolutional layers followed by maxpooling layers.the number of kernels feature maps after each block doubles so that architecture can learns complex structures effectively.
- Similar to contraction layer, the expansion section also consists of several expansion blocks, each block passes through convolutional layers followed by upsampling layers.
- The number of expansion blocks is same as the number of contractiion blocks.
- Every time the input get appended by the feature maps of the contracting layer ,that will ensure the features that are learned and image will be used to reconstruct it.

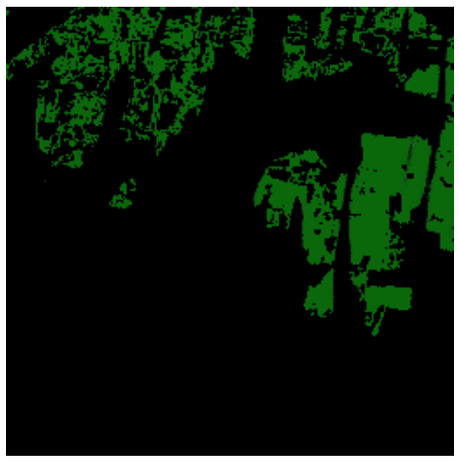
Code:

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(1,1),input_shape=input_shape,activation='relu'))
model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',padding='same'))
model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',padding='same'))
model.add(MaxPooling2D(pool_size=(2,2)))


#model.add(Dropout(0.8))
model.add(Conv2D(32, kernel_size=(1,1),activation='relu',data_format='channels_first'))
model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',padding='same'))
model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',padding='same'))
model.add(MaxPooling2D(pool_size=(2,2)))

#model.add(Dropout(0.8))
model.add(Conv2D(32, kernel_size=(1,1),activation='relu',data_format='channels_first'))
model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',padding='same'))
model.add(Conv2D(32, kernel_size=(3, 3),activation='relu',padding='same'))
model.add(UpSampling2D(size=(1,1),data_format='channels_first'))
model.add(UpSampling2D(size=(3, 3),data_format='channels_first'))

#model.add(Dropout(0.8))
model.add(UpSampling2D(size=(3, 3),data_format='channels_first'))
model.add(Conv2D(27, (3, 3), activation='relu',padding='same',data_format='channels_first'))
model.add(Dropout(0.8))

model.add(Flatten())
model.add(Dense(32, activation='relu'))
model.add(Dropout(0.8))
model.add(Dense(3, activation='softmax'))

#sgd = SGD(lr=0.0001, decay=1e-6, momentum=0.9, nesterov=True)
#model.compile(loss='categorical_crossentropy', optimizer='sgd', metrics=['accuracy'])

adam=Adam(learning_rate=0.001, beta_1=0.9, beta_2=0.999, amsgrad=False)
model.compile(loss='categorical_crossentropy',optimizer = Adam(lr = 1e-4), metrics=['accuracy'])

model.fit(X_train,Y_train,epochs=5)
#model.fit(X_train, Y_train, batch_size = 1000, nb_epoch = 5, verbose = 1, shuffle = True)

model.summary()
```

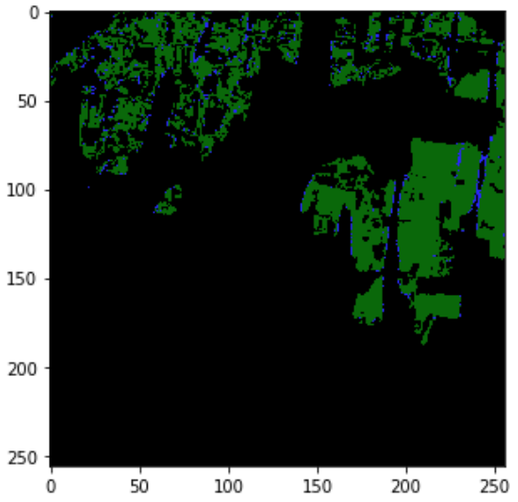
input image:



Target image:



Predicted image:



- Loss and accuracy:
  - Model 1(using original layer images by patching): loss- **0.6** accuracy- **82%**
  - Model 2(using splitting): loss-<**0.1** accuracy-**90%**

References:

- Introducing symmetry in segmentation article by heat sankesara-"https://towardsdatascience.com/u-net-b229b32b4a71"

- Learn how to train your U net on your dataset by sukriti paul-“https://medium.com/coinmonks/learn-how-to-train-u-net-on-your-dataset-8e3f89fbd623”

- Deep feature extraction and combination for synthetic aperture radar target classification by Moussa Armani -“https://www.semanticscholar.org/paper/Deep-feature-extraction-and-combination-for-radar-Amrani-Jiang/eea3b598a6ef21dd7fca426b365f24806f06bff1”

Thank you sir