**Note there is already 1 example training model that was already done. In this case it was vgg\_unet, so if you don’t want to train and just see how it works:**

1. Unzip and follow step 1-4 from **Training and Validation of Deep Encoder-Decoder Networks Stage 2 on Rebar Dataset**

2. The prediction model is in **checkpoints/** folder

3. Go into **prediction** folder which will give you the predict from model for images in **example\_dataset/images\_prepped\_test**

4. Run the code line given below in the command prompt to get metric like precision, iou score, etc

**python3 -m keras\_segmentation evaluate\_model --checkpoints\_path="checkpoints/checkpoints" --images\_path="example\_dataset/images\_prepped\_test" --segs\_path="example\_dataset/annotations\_prepped\_test"**

*note: there might be some warnings so you might have to scroll around to find the print statement in terminal.*

5. To train, go to next section will show you stage 2 & 1.

**Training and Validation of Deep Encoder-Decoder Networks** **Stage 2 on Rebar Dataset**

1. Download the code ZIP file.

2. Unzip the folder

3. Install gpu for Tensorflow (recommended version: 2.4.1) <https://www.tensorflow.org/install/gpu>

4. Resolve the following dependencies:

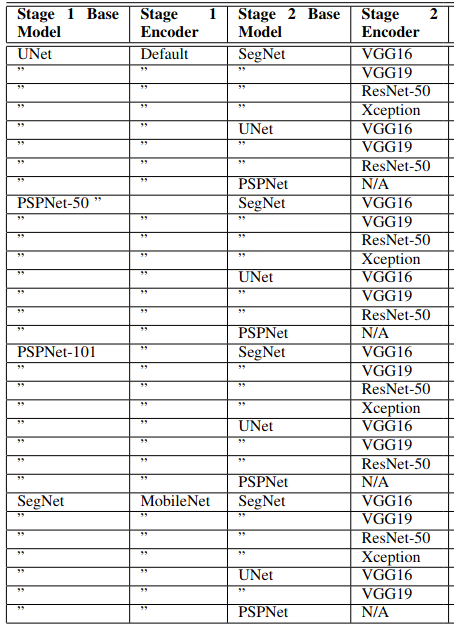
* Keras (recommended version: 2.4.3)
  + Use the command “**pip3 install keras**“
* OpenCV for Python
* [imgaug](https://www.tensorflow.org/install/gpu)
* tqdm

5. Run the code for training the SegNet or other frameworks using the code line given below:

**python3 -m keras\_segmentation train --checkpoints\_path="checkpoints" --train\_images="example\_dataset/images\_prepped\_train" --train\_annotations="example\_dataset/annotations\_prepped\_train" --val\_images="example\_dataset/images\_prepped\_test" --val\_annotations="example\_dataset/annotations\_prepped\_test" --n\_classes=2 --input\_height=764 --input\_width=764 --model\_name="segnet" --epoch=100 --batch\_size=8**

where **segnet** is, switch out different type of encoder + base model combinations like ‘xception\_segnet’, ‘vgg\_segnet’, etc. Refer to paper for all different types for stage 2. **checkpoints** is where the model will go. If you just have checkpoints, then all model will be in **Segmentation + segnet/** folder. If you want your model to go to a different folder like mine where I have it in **checkpoints/,** then first create a folder name **<folder name>**, then in place of **checkpoints**, put in **<folder name>/checkpoints.**

For input height and width, please check the base model requirement by opening **keras\_segmentation/models/<base model name>.py** and input the right width and height, you will get errors. Below is the base model and encoder names:



6. Run the code line given below in the command prompt to validate the trained Deep Encoder Decoder system:

**python3 -m keras\_segmentation predict --checkpoints\_path="checkpoints" --input\_path="example\_dataset/images\_prepped\_test" --output\_path="prediction"**

For **checkpoints**, if in **step 5** you choose to put your model in **<folder name>/checkpoints**, then you will have to put **<folder name>/checkpoints** in place of **checkpoints**

7. Run the code line given below in the command prompt to get metric like precision, iou score, etc

**python3 -m keras\_segmentation evaluate\_model --checkpoints\_path="checkpoints" --images\_path="example\_dataset/images\_prepped\_test" --segs\_path="example\_dataset/annotations\_prepped\_test"**

*note: there might be some warnings so you might have to scroll around to find the print statement in terminal.*

For **checkpoints**, if in **step 5** you choose to put your model in **<folder name>/checkpoints**, then you will have to put **<folder name>/checkpoints** in place of **checkpoints**

**Training and Validation of Deep Encoder-Decoder Networks** **Stage 1 on Rebar Dataset**

1. Follow step 1-4 in **Training and Validation of Deep Encoder-Decoder Networks** **Stage 2 on Rebar Dataset**

2. Run the code for training the SegNet or other frameworks using the code line given below:

**python3 -m keras\_segmentation train --checkpoints\_path="checkpoints" --train\_images="example\_dataset/Img" --train\_annotations="example\_dataset/Img\_Seg" --val\_images="example\_dataset/Img\_Test" --val\_annotations="example\_dataset/Img\_Seg\_Test" --n\_classes=2 --input\_height=764 --input\_width=764 --model\_name="mobilenet\_segnet" --epoch=100 –batch\_size=8**

where **mobilenet\_segnet** is, switch out different type of encoder + base model combinations like ‘unet’, etc. Refer to paper for all different types for stage 1

**Data:** The filenames of the annotation images should be same as the filenames of the RGB images.

The size of the annotation image for the corresponding RGB image should be same.

The annotation should follow the RGB value range based on the class number. If the segmentation is on binary class the annotation rgb values should be within (0-1)

range.