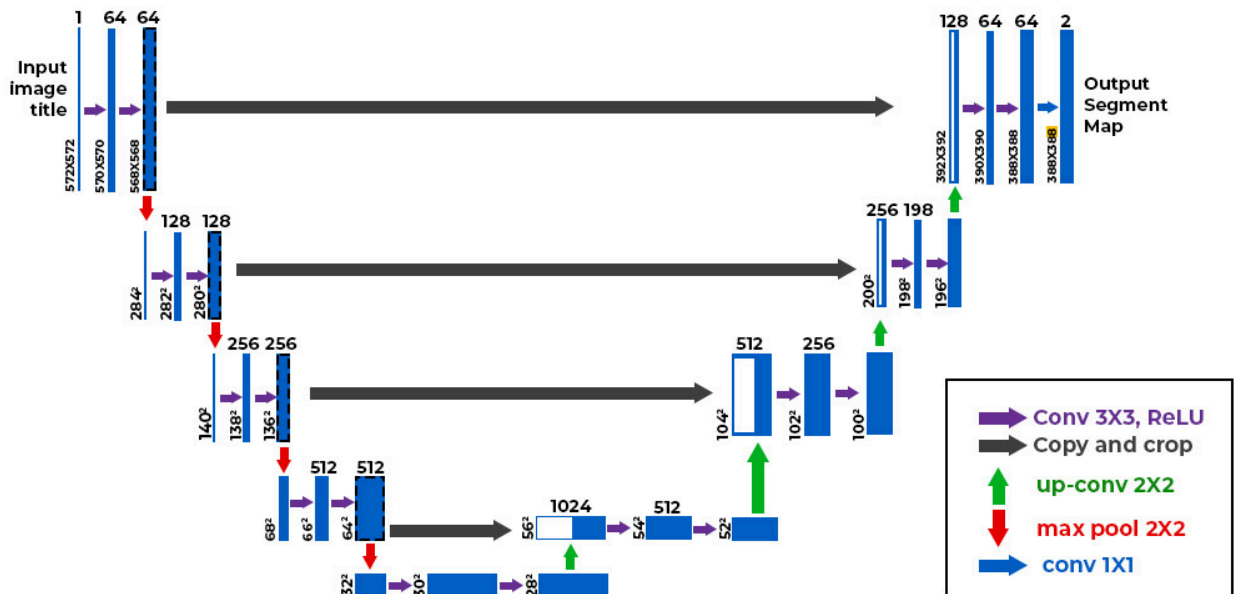


UNET code using Python-Tensorflow-Keras

- BY DIVESH JADHWANI



```
def multi_unet_model(n_classes=5, image_height=256,  
image_width=256, image_channels=1):
```

```
inputs = Input((image_height, image_width, image_channels))
```

MAX POOLING

source_input = inputs

**c1 = Conv2D(16, (3,3), activation="relu",
kernel_initializer="he_normal", padding="same")(source_input)**

c1 = Dropout(0.2)(c1)

**c1 = Conv2D(16, (3,3), activation="relu",
kernel_initializer="he_normal", padding="same")(c1)**

p1 = MaxPooling2D((2,2))(c1)

**c2 = Conv2D(32, (3,3), activation="relu",
kernel_initializer="he_normal", padding="same")(p1)**

c2 = Dropout(0.2)(c2)

**c2 = Conv2D(32, (3,3), activation="relu",
kernel_initializer="he_normal", padding="same")(c2)**

p2 = MaxPooling2D((2,2))(c2)

**c3 = Conv2D(64, (3,3), activation="relu",
kernel_initializer="he_normal", padding="same")(p2)**

c3 = Dropout(0.2)(c3)

**c3 = Conv2D(64, (3,3), activation="relu",
kernel_initializer="he_normal", padding="same")(c3)**

p3 = MaxPooling2D((2,2))(c3)

**c4 = Conv2D(128, (3,3), activation="relu",
kernel_initializer="he_normal", padding="same")(p3)**

c4 = Dropout(0.2)(c4)

**c4 = Conv2D(128, (3,3), activation="relu",
kernel_initializer="he_normal", padding="same")(c4)**

p4 = MaxPooling2D((2,2))(c4)

UP SAMPLING

```
c5 = Conv2D(256, (3,3), activation="relu",  
kernel_initializer="he_normal", padding="same")(p4)  
c5 = Dropout(0.2)(c5)  
c5 = Conv2D(256, (3,3), activation="relu",  
kernel_initializer="he_normal", padding="same")(c5)
```

```
u6 = Conv2DTranspose(128, (2,2), strides=(2,2),  
padding="same")(c5)  
u6 = concatenate([u6, c4])  
c6 = Conv2D(128, (3,3), activation="relu",  
kernel_initializer="he_normal", padding="same")(u6)  
c6 = Dropout(0.2)(c6)  
c6 = Conv2D(128, (3,3), activation="relu",  
kernel_initializer="he_normal", padding="same")(c6)
```

```
u7 = Conv2DTranspose(64, (2,2), strides=(2,2),  
padding="same")(c6)  
u7 = concatenate([u7, c3])  
c7 = Conv2D(64, (3,3), activation="relu",  
kernel_initializer="he_normal", padding="same")(u7)  
c7 = Dropout(0.2)(c7)  
c7 = Conv2D(64, (3,3), activation="relu",  
kernel_initializer="he_normal", padding="same")(c7)
```

```
u8 = Conv2DTranspose(32, (2,2), strides=(2,2),  
padding="same")(c7)  
u8 = concatenate([u8, c2])
```

```
c8 = Conv2D(32, (3,3), activation="relu",  
kernel_initializer="he_normal", padding="same")(u8)  
c8 = Dropout(0.2)(c8)  
c8 = Conv2D(32, (3,3), activation="relu",  
kernel_initializer="he_normal", padding="same")(c8)
```

```
u9 = Conv2DTranspose(16, (2,2), strides=(2,2),  
padding="same")(c8)  
u9 = concatenate([u9, c1], axis=3)  
c9 = Conv2D(16, (3,3), activation="relu",  
kernel_initializer="he_normal", padding="same")(u9)  
c9 = Dropout(0.2)(c9)  
c9 = Conv2D(16, (3,3), activation="relu",  
kernel_initializer="he_normal", padding="same")(c9)
```

```
outputs = Conv2D(n_classes, (1,1), activation="softmax")(c9)
```

```
model = Model(inputs=[inputs], outputs=[outputs])
```

```
return model
```