Task 1 : Set up colab gpu runtime environment

```
!pip install segmentation-models-pytorch
!pip install -U git+https://github.com/albumentations-team/albumentations
!pip install --upgrade opencv-contrib-python
```

```
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->torchvision>=0.5.0->segmentati
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch->efficientnet-pytorch==
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/python3.10/dist-packages (from sympy->torch->efficientnet-pytorch==0.7.
Building wheels for collected packages: efficientnet-pytorch, pretrainedmodels
  Building wheel for efficientnet-pytorch (setup.py) ... done
  Created wheel for efficientnet-pytorch: filename=efficientnet_pytorch-0.7.1-py3-none-any.whl size=16427 sha256=9afe9b5df7c1672d7364
  Stored in directory: /root/.cache/pip/wheels/03/3f/e9/911b1bc46869644912bda90a56bcf7b960f20b5187feea3baf
  Building wheel for pretrainedmodels (setup.py) ... done
  Created wheel for pretrainedmodels: filename=pretrainedmodels-0.7.4-py3-none-any.whl size=60945 sha256=8e38d8f538a213930f1810ea22d4
  Stored in directory: /root/.cache/pip/wheels/35/cb/a5/8f534c60142835bfc889f9a482e4a67e0b817032d9c6883b64
Successfully built efficientnet-pytorch pretrainedmodels
Installing collected packages: safetensors, munch, huggingface-hub, timm, pretrainedmodels, efficientnet-pytorch, segmentation-models
Successfully installed efficientnet-pytorch-0.7.1 huggingface-hub-0.16.4 munch-4.0.0 pretrainedmodels-0.7.4 safetensors-0.3.1 segment
Collecting git+https://github.com/albumentations-team/albumentations
  Cloning https://github.com/albumentations-team/albumentations to /tmp/pip-req-build-b2ba0sqp
  Running command git clone --filter=blob:none --quiet https://github.com/albumentations-team/albumentations /tmp/pip-req-build-b2ba0
  Resolved https://github.com/albumentations-team/albumentations to commit e3b47b3a127f92541cfeb16abbb44a6f8bf79cc8
  Preparing metadata (setup.py) ... done
Requirement already satisfied: numpy>=1.11.1 in /usr/local/lib/python3.10/dist-packages (from albumentations==1.3.1) (1.22.4)
Requirement already satisfied: scipy>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from albumentations==1.3.1) (1.10.1)
Requirement already satisfied: scikit-image>=0.16.1 in /usr/local/lib/python3.10/dist-packages (from albumentations==1.3.1) (0.19.3)
Requirement already satisfied: PyYAML in /usr/local/lib/python3.10/dist-packages (from albumentations==1.3.1) (6.0)
Requirement already satisfied: qudida>=0.0.4 in /usr/local/lib/python3.10/dist-packages (from albumentations==1.3.1) (0.0.4)
Requirement already satisfied: opencv-python>=4.1.1 in /usr/local/lib/python3.10/dist-packages (from albumentations==1.3.1) (4.7.0.72
Requirement already satisfied: scikit-learn>=0.19.1 in /usr/local/lib/python3.10/dist-packages (from qudida>=0.0.4->albumentations==1
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.10/dist-packages (from qudida>=0.0.4->albumentations==1.3.
Requirement already satisfied: opencv-python-headless>=4.0.1 in /usr/local/lib/python3.10/dist-packages (from qudida>=0.0.4->albument
Requirement already satisfied: networkx>=2.2 in /usr/local/lib/python3.10/dist-packages (from scikit-image>=0.16.1->albumentations==1
Requirement already satisfied: pillow!=7.1.0,!=7.1.1,!=8.3.0,>=6.1.0 in /usr/local/lib/python3.10/dist-packages (from scikit-image>=0
Requirement already satisfied: imageio>=2.4.1 in /usr/local/lib/python3.10/dist-packages (from scikit-image>=0.16.1->albumentations==
Requirement already satisfied: tifffile>=2019.7.26 in /usr/local/lib/python3.10/dist-packages (from scikit-image>=0.16.1->albumentati
Requirement already satisfied: PyWavelets>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-image>=0.16.1->albumentation
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from scikit-image>=0.16.1->albumentations=
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn>=0.19.1->qudida>=0.0.4->al
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn>=0.19.1->qudida>=0.
Building wheels for collected packages: albumentations
  Building wheel for albumentations (setup.py) ... done
  Created wheel for albumentations: filename=albumentations-1.3.1-py3-none-any.whl size=125700 sha256=0071ba216d3cd0cec76fbe45d171575
  Stored in directory: /tmp/pip-ephem-wheel-cache-oz6n786p/wheels/51/4d/ab/5aafa8b980086fbc362946de7da4aa3df33aacb3da0da29b93
Successfully built albumentations
Installing collected packages: albumentations
  Attempting uninstall: albumentations
    Found existing installation: albumentations 1.2.1
    Uninstalling albumentations-1.2.1:
      Successfully uninstalled albumentations-1.2.1
Successfully installed albumentations-1.3.1
Requirement already satisfied: opencv-contrib-python in /usr/local/lib/python3.10/dist-packages (4.7.0.72)
Collecting opencv-contrib-python
  Downloading opencv_contrib_python-4.8.0.74-cp37-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (67.8 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 67.8/67.8 MB 11.0 MB/s eta 0:00:00
Requirement already satisfied: numpy>=1.21.2 in /usr/local/lib/python3.10/dist-packages (from opencv-contrib-python) (1.22.4)
Installing collected packages: opencv-contrib-python
  Attempting uninstall: opencv-contrib-python
    Found existing installation: opencv-contrib-python 4.7.0.72
    Uninstalling opencv-contrib-python-4.7.0.72:
      Successfully uninstalled opencv-contrib-python-4.7.0.72
Successfully installed opencv-contrib-python-4.8.0.74
```

About Dataset

Dataset

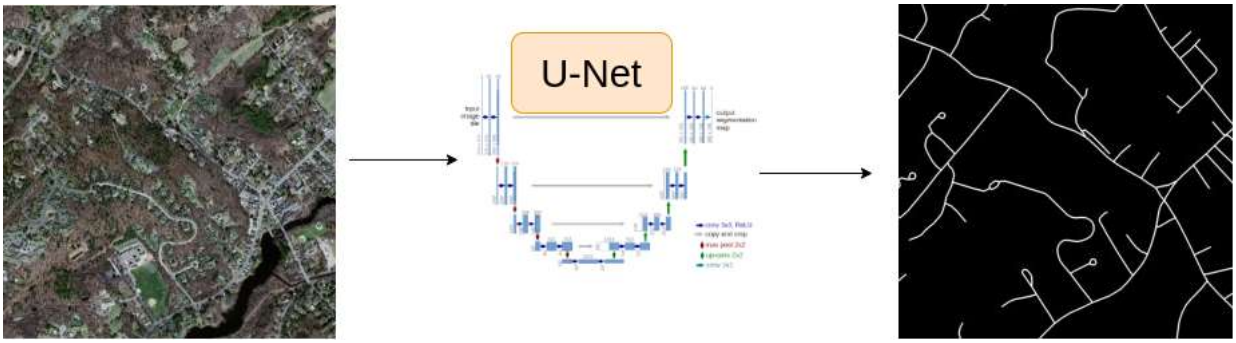
Here the dataset which we are going to use in this guided project is the subset(200 images and its masks) of the original dataset (Massachusetts Roads Dataset) consists of 1171 aerial images of the state of Massachusetts. Each image is 1500×1500 pixels in size, covering an area of 2.25 square kilometers

Full Dataset

After compeletion of this project you can try the same pipeline on full dataset

https://www.cs.toronto.edu/~vmnih/data/

```
@phdthesis{MnihThesis,
author = {Volodymyr Mnih},
title = {Machine Learning for Aerial Image Labeling},
school = {University of Toronto},
year = {2013}
}
```



Download Subset Dataset

```
!git clone https://github.com/parth1620/Road_seg_dataset.git
```

```
Cloning into 'Road_seg_dataset'...
remote: Enumerating objects: 411, done.
remote: Total 411 (delta 0), reused 0 (delta 0), pack-reused 411
Receiving objects: 100% (411/411), 851.74 MiB | 18.26 MiB/s, done.
Resolving deltas: 100% (2/2), done.
Updating files: 100% (401/401), done.
```

Some Common Imports

```
import sys
sys.path.append('/content/Road_seg_dataset')
```

```
import torch
import cv2

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from tqdm import tqdm

import helper
```

Task : 2 Setup Configurations

```
CSV_FILE = '/content/Road_seg_dataset/train.csv'
DATA_DIR = '/content/Road_seg_dataset/'

DEVICE = 'cuda'

EPOCHS = 15
LR = 0.003
BATCH_SIZE = 8
IMG_SIZE = 512
```

```
ENCODER = 'timm-efficientnet-b0'
WEIGHTS = 'imagenet'


df = pd.read_csv(CSV_FILE)
df.head()
```

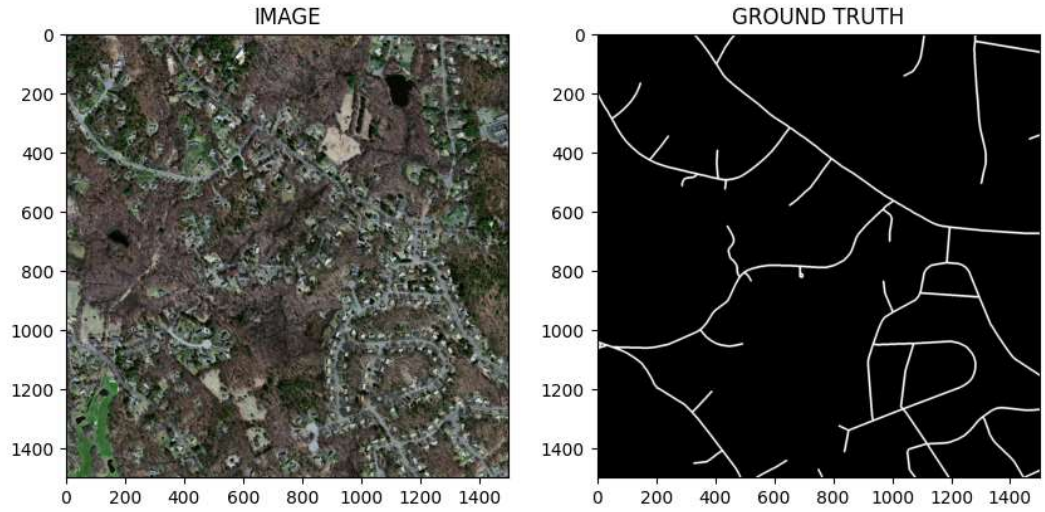
	images	masks
0	images/17428750_15.png	masks/17428750_15.png
1	images/23279080_15.png	masks/23279080_15.png
2	images/24179185_15.png	masks/24179185_15.png
3	images/24179035_15.png	masks/24179035_15.png
4	images/11128810_15.png	masks/11128810_15.png

```
idx = 20

row = df.iloc[idx]

image_path = DATA_DIR + row.images
mask_path = DATA_DIR + row.masks

print(image_path + '\n' + mask_path)

image = cv2.imread(image_path)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

mask = cv2.imread(mask_path, cv2.IMREAD_GRAYSCALE) / 255
```

```
/content/Road_seg_dataset/images/20428975_15.png
/content/Road_seg_dataset/masks/20428975_15.png
```

```
f, (ax1, ax2) = plt.subplots(1, 2, figsize=(10,5))

ax1.set_title('IMAGE')
ax1.imshow(image)

ax2.set_title('GROUND TRUTH')
ax2.imshow(mask,cmap = 'gray')
```

```
<matplotlib.image.AxesImage at 0x7dabee5ae560>
```




```
train_df, valid_df = train_test_split(df, test_size=0.20, random_state=42)


print(len(train_df))
print(len(valid_df))
```

```
159
40
```

Task 3 : Augmentation Functions

albumentation documentation : https://albumentations.ai/docs/

```
import albumentations as A


def get_train_augs():
    return A.Compose([
        A.Resize(IMG_SIZE, IMG_SIZE),
        A.HorizontalFlip(p=0.5),
        A.VerticalFlip(p=0.5)
    ])

def get_valid_augs():
    return A.Compose([
        A.Resize(IMG_SIZE, IMG_SIZE)
    ])
```

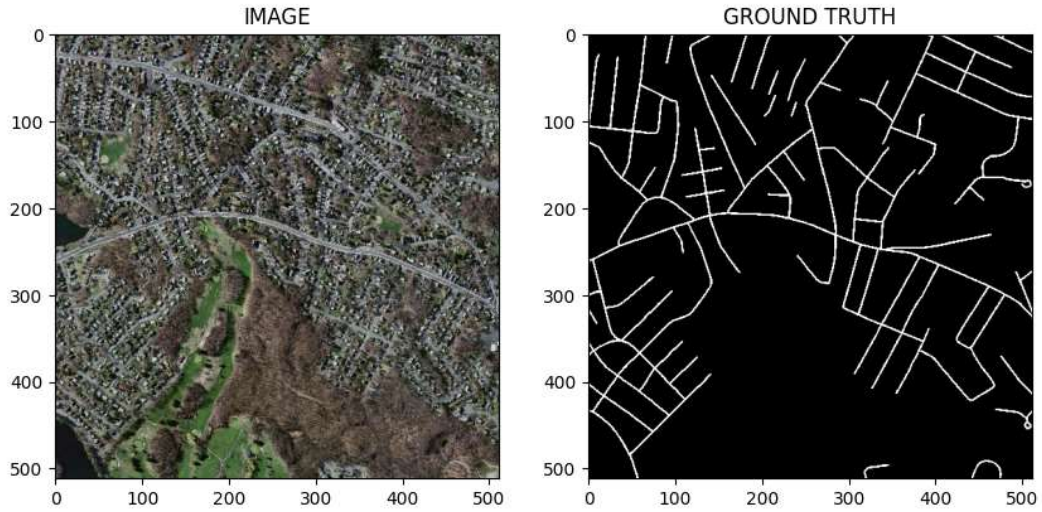
Task 4 : Create Custom Dataset

```
from torch.utils.data import Dataset


class SegmentationDataset(Dataset):

    def __init__(self, df, augmentations):
        self.df = df
        self.augmentations = augmentations

    def __len__(self):
        return len(self.df)

    def __getitem__(self, idx):
        row = self.df.iloc[idx]
        image_path = DATA_DIR + row.images
        mask_path = DATA_DIR + row.masks

        image = cv2.imread(image_path)
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

        mask = cv2.imread(mask_path, cv2.IMREAD_GRAYSCALE)# (h,w)
        mask = np.expand_dims(mask, axis = -1) # (h,w,c)

        if self.augmentations:
            data = self.augmentations(image=image, mask=mask)
            image = data['image']
            mask = data['mask']

        image = np.transpose(image, (2,0,1)).astype(np.float32) #(c,h,w)
        mask = np.transpose(mask, (2,0,1)).astype(np.float32) #(c,h,w)

        image = torch.Tensor(image) / 255.0
        mask = torch.round(torch.Tensor(mask) / 255.0)

        return image, mask


trainset = SegmentationDataset(train_df, get_train_augs())
validset = SegmentationDataset(valid_df, get_valid_augs())


print(f'Size of trainset : {len(trainset)}')
print(f'Size of validset : {len(validset)}')
```

```
Size of trainset : 159
Size of validset : 40
```

```
idx = 20
```

```
image, mask = trainset[idx]

helper.show_image(image, mask)
```



Task 5 : Load dataset into batches

```
from torch.utils.data import DataLoader


trainloader = DataLoader(trainset, batch_size=BATCH_SIZE)
validloader = DataLoader(validset, batch_size=BATCH_SIZE)


print(f'Total number of batches in trainloader : {len(trainloader)}')
print(f'Total number of batches in validloader : {len(validloader)}')
```

```
Total number of batches in trainloader : 20
Total number of batches in validloader : 5
```

```
for images, masks in trainloader:
    print(f'One batch image shape : {images.shape}')
    print(f'One batch mask shape : {masks.shape}')
    break
```

```
One batch image shape : torch.Size([8, 3, 512, 512])
One batch mask shape : torch.Size([8, 1, 512, 512])
```

Task 6 : Create Segmentation Model

segmentation_models_pytorch documentation : https://smp.readthedocs.io/en/latest/

```
import segmentation_models_pytorch as smp
from segmentation_models_pytorch.losses import DiceLoss

from torch import nn


class SegmentationModel(nn.Module):

    def __init__(self):
        super(SegmentationModel,self).__init__()

        self.backbone = smp.Unet(
            encoder_name = ENCODER,
            encoder_weights = WEIGHTS,
            in_channels = 3,
            classes = 1,
            activation = None
```

```
        )

    def forward(self, images, masks = None):

        logits = self.backbone(images)

        if masks != None:
            return logits, DiceLoss(mode = 'binary')(logits, masks) + nn.BCEWithLogitsLoss()(logits, masks)

        return logits


model = SegmentationModel()
model.to(DEVICE)
```

```
                (se): SqueezeExcite(
                  (conv_reduce): Conv2d(480, 20, kernel_size=(1, 1), stride=(1, 1))
                  (act1): Swish()
                  (conv_expand): Conv2d(20, 480, kernel_size=(1, 1), stride=(1, 1))
                  (gate): Sigmoid()
                )
                (conv_pwl): Conv2d(480, 80, kernel_size=(1, 1), stride=(1, 1), bias=False)
                (bn3): BatchNormAct2d(
                  80, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True
                  (drop): Identity()
                  (act): Identity()
                )
                (drop_path): DropPath(drop_prob=0.088)
              )
            )
            (4): Sequential(
              (0): InvertedResidual(
                (conv_pw): Conv2d(80, 480, kernel_size=(1, 1), stride=(1, 1), bias=False)
                (bn1): BatchNormAct2d(
                  480, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True
                  (drop): Identity()
                  (act): Swish()
                )
                (conv_dw): Conv2d(480, 480, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2), groups=480, bias=False)
                (bn2): BatchNormAct2d(
                  480, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True
                  (drop): Identity()
                  (act): Swish()
                )
                (se): SqueezeExcite(
                  (conv_reduce): Conv2d(480, 20, kernel_size=(1, 1), stride=(1, 1))
                  (act1): Swish()
                  (conv_expand): Conv2d(20, 480, kernel_size=(1, 1), stride=(1, 1))
                  (gate): Sigmoid()
                )
                (conv_pwl): Conv2d(480, 112, kernel_size=(1, 1), stride=(1, 1), bias=False)
                (bn3): BatchNormAct2d(
                  112, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True
                  (drop): Identity()
                  (act): Identity()
                )
                (drop_path): DropPath(drop_prob=0.100)
              )
              (1): InvertedResidual(
                (conv_pw): Conv2d(112, 672, kernel_size=(1, 1), stride=(1, 1), bias=False)
                (bn1): BatchNormAct2d(
                  672, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True
                  (drop): Identity()
                  (act): Swish()
                )
                (conv_dw): Conv2d(672, 672, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2), groups=672, bias=False)
                (bn2): BatchNormAct2d(
                  672, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True
                  (drop): Identity()
                  (act): Swish()
                )
                (se): SqueezeExcite(
                  (conv_reduce): Conv2d(672, 28, kernel_size=(1, 1), stride=(1, 1))
```

Task 7 : Create Train and Validation Function

```
def train_fn(dataloader, model, optimizer):

    model.train() # Turn ON dropout, batchnorm, etc
    total_loss = 0.0
```

```
    for images, masks in tqdm(dataloader):

        images = images.to(DEVICE)
        masks = masks.to(DEVICE)

        optimizer.zero_grad()
        logits, loss = model(images, masks)
        loss.backward()
        optimizer.step()

        total_loss += loss.item()

    return total_loss / len(dataloader)


def eval_fn(dataloader, model):

    model.eval() # Turn OFF dropout, batchnorm, etc
    total_loss = 0.0

    with torch.no_grad():

        for images, masks in tqdm(dataloader):

            images = images.to(DEVICE)
            masks = masks.to(DEVICE)
            logits, loss = model(images, masks)
            total_loss += loss.item()

        return total_loss / len(dataloader)
```

Task 8 : Train Model

```
optimizer = torch.optim.Adam(model.parameters(), lr = LR)


best_loss = np.Inf

for i in range(EPOCHS):
    train_loss = train_fn(trainloader, model, optimizer)
    valid_loss = eval_fn(trainloader, model)

    if valid_loss < best_loss:
        torch.save(model.state_dict(), "best-model.pt")
        print("SAVED-MODEL")
        best_loss = valid_loss

    print(f"Epoch : {i+1} Train Loss : {train_loss} Valid Loss : {valid_loss}")
```

```
 100%|██████████| 20/20 [00:21<00:00,  1.10s/it]
 100%|██████████| 20/20 [00:17<00:00,  1.14it/s]
 SAVED-MODEL
 Epoch : 1 Train Loss : 0.7342194855213166 Valid Loss : 0.8394036501646042
 100%|██████████| 20/20 [00:29<00:00,  1.49s/it]
 100%|██████████| 20/20 [00:20<00:00,  1.03s/it]
 SAVED-MODEL
 Epoch : 2 Train Loss : 0.6987123727798462 Valid Loss : 0.7301326841115952
 100%|██████████| 20/20 [00:26<00:00,  1.30s/it]
 100%|██████████| 20/20 [00:19<00:00,  1.02it/s]
 SAVED-MODEL
 Epoch : 3 Train Loss : 0.6775169938802719 Valid Loss : 0.681255754828453
 100%|██████████| 20/20 [00:26<00:00,  1.34s/it]
 100%|██████████| 20/20 [00:20<00:00,  1.02s/it]
 SAVED-MODEL
 Epoch : 4 Train Loss : 0.6695525914430618 Valid Loss : 0.6755333960056304
 100%|██████████| 20/20 [00:27<00:00,  1.37s/it]
 100%|██████████| 20/20 [00:20<00:00,  1.03s/it]
 SAVED-MODEL
 Epoch : 5 Train Loss : 0.6432499021291733 Valid Loss : 0.6331136375665665
 100%|██████████| 20/20 [00:24<00:00,  1.22s/it]
 100%|██████████| 20/20 [00:16<00:00,  1.21it/s]
 SAVED-MODEL
 Epoch : 6 Train Loss : 0.6271708250045777 Valid Loss : 0.6290787070989609
 100%|██████████| 20/20 [00:21<00:00,  1.09s/it]
 100%|██████████| 20/20 [00:21<00:00,  1.06s/it]
 SAVED-MODEL
```

```
Epoch : 7 Train Loss : 0.6181249469518661 Valid Loss : 0.6106242090463638
100%|██████████| 20/20 [00:21<00:00,  1.10s/it]
100%|██████████| 20/20 [00:15<00:00,  1.32it/s]
SAVED-MODEL
Epoch : 8 Train Loss : 0.61368168592453 Valid Loss : 0.602013349533081
100%|██████████| 20/20 [00:21<00:00,  1.10s/it]
100%|██████████| 20/20 [00:15<00:00,  1.30it/s]
SAVED-MODEL
Epoch : 9 Train Loss : 0.5998836487531662 Valid Loss : 0.5988190352916718
100%|██████████| 20/20 [00:22<00:00,  1.11s/it]
100%|██████████| 20/20 [00:15<00:00,  1.31it/s]
Epoch : 10 Train Loss : 0.6078713506460189 Valid Loss : 0.6136816561222076
100%|██████████| 20/20 [00:21<00:00,  1.09s/it]
100%|██████████| 20/20 [00:15<00:00,  1.31it/s]
Epoch : 11 Train Loss : 0.6154474556446076 Valid Loss : 0.6333729833364486
100%|██████████| 20/20 [00:21<00:00,  1.09s/it]
100%|██████████| 20/20 [00:15<00:00,  1.27it/s]
SAVED-MODEL
Epoch : 12 Train Loss : 0.5953654542565345 Valid Loss : 0.5938671320676804
100%|██████████| 20/20 [00:22<00:00,  1.10s/it]
100%|██████████| 20/20 [00:15<00:00,  1.28it/s]
SAVED-MODEL
Epoch : 13 Train Loss : 0.5897223949432373 Valid Loss : 0.5908070012927056
100%|██████████| 20/20 [00:21<00:00,  1.08s/it]
100%|██████████| 20/20 [00:15<00:00,  1.27it/s]
SAVED-MODEL
Epoch : 14 Train Loss : 0.5906181469559669 Valid Loss : 0.5809299558401108
100%|██████████| 20/20 [00:21<00:00,  1.08s/it]
100%|██████████| 20/20 [00:15<00:00,  1.26it/s]SAVED-MODEL
Epoch : 15 Train Loss : 0.5786691695451737 Valid Loss : 0.5753279447555542
```

Task 9 : Inference

```
idx = 20

model.load_state_dict(torch.load('/content/best-model.pt'))
image, mask = validset[idx]

logits_mask = model(image.to(DEVICE).unsqueeze(0)) # (c,h,w) -> (b,c,h,w)
pred_mask = torch.sigmoid(logits_mask)
pred_mask = (pred_mask > 0.5)*1.0


helper.show_image(image, mask, pred_mask.detach().cpu().squeeze(0))
```

✓ 1s completed at 1:26 AM