IGVC_Full_Dataset

March 20, 2020

[0]:
```
# Mount Drive
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[0]:
```
# Restart runtime after running once

# Library for pretrained segmentation models fo PyTorch
!pip install segmentation-models-pytorch==0.1.0

# Catalyst library
!pip install -U catalyst
```

```
Collecting segmentation-models-pytorch==0.1.0
  Downloading https://files.pythonhosted.org/packages/70/88/763a25dfe076a9
f30f33466b1bd0f2d31b915b88d4cb4481fe4043cf26b4/segmentation_models_pytorch-0.1.0
-py3-none-any.whl (42kB)
     |                  | 51kB 1.9MB/s
Collecting pretrainedmodels==0.7.4
  Downloading https://files.pythonhosted.org/packages/84/0e/be6a0e58447ac1
6c938799d49bfb5fb7a80ac35e137547fc6cee2c08c4cf/pretrainedmodels-0.7.4.tar.gz
(58kB)
     |                  | 61kB 3.8MB/s
Collecting efficientnet-pytorch>=0.5.1
  Downloading https://files.pythonhosted.org/packages/b8/cb/0309a6e3d404862ae4bc
017f89645cf150ac94c14c88ef81d215c8e52925/efficientnet_pytorch-0.6.3.tar.gz
Requirement already satisfied: torchvision>=0.3.0 in
/usr/local/lib/python3.6/dist-packages (from segmentation-models-pytorch==0.1.0)
(0.5.0)
Requirement already satisfied: torch in /usr/local/lib/python3.6/dist-packages
(from pretrainedmodels==0.7.4->segmentation-models-pytorch==0.1.0) (1.4.0)
Collecting munch
  Downloading https://files.pythonhosted.org/packages/cc/ab/85d8da5c9a45e072301b
eb37ad7f833cd344e04c817d97e0cc75681d248f/munch-2.5.0-py2.py3-none-any.whl
Requirement already satisfied: tqdm in /usr/local/lib/python3.6/dist-packages
(from pretrainedmodels==0.7.4->segmentation-models-pytorch==0.1.0) (4.38.0)
```

```
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages
(from torchvision>=0.3.0->segmentation-models-pytorch==0.1.0) (1.18.2)
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages
(from torchvision>=0.3.0->segmentation-models-pytorch==0.1.0) (1.12.0)
Requirement already satisfied: pillow>=4.1.1 in /usr/local/lib/python3.6/dist-
packages (from torchvision>=0.3.0->segmentation-models-pytorch==0.1.0) (7.0.0)
Building wheels for collected packages: pretrainedmodels, efficientnet-pytorch
  Building wheel for pretrainedmodels (setup.py) … done
  Created wheel for pretrainedmodels: filename=pretrainedmodels-0.7.4-cp36-none-
any.whl size=60962
sha256=8796ea1a7273a0e5773a1176fc77412d6e94d706501116a0b6b3b6a652bd2656
  Stored in directory: /root/.cache/pip/wheels/69/df/63/62583c096289713f22db605a
a2334de5b591d59861a02c2ecd
  Building wheel for efficientnet-pytorch (setup.py) … done
  Created wheel for efficientnet-pytorch:
filename=efficientnet_pytorch-0.6.3-cp36-none-any.whl size=12422
sha256=a01f8eea99b45da3bde07c107bd2091465f2c19ede5f06747514acbb691aa8b0
  Stored in directory: /root/.cache/pip/wheels/42/1e/a9/2a578ba9ad04e776e80bf0f7
0d8a7f4c29ec0718b92d8f6ccd
Successfully built pretrainedmodels efficientnet-pytorch
Installing collected packages: munch, pretrainedmodels, efficientnet-pytorch,
segmentation-models-pytorch
Successfully installed efficientnet-pytorch-0.6.3 munch-2.5.0
pretrainedmodels-0.7.4 segmentation-models-pytorch-0.1.0
Collecting catalyst
  Downloading https://files.pythonhosted.org/packages/c9/02/8328fb3b01b7c1
28a4f8abfe2df9508bcae64866a8c41c44ea9f7981f44f/catalyst-20.3.2-py2.py3-none-
any.whl (388kB)
     |                  | 389kB 3.4MB/s
Collecting crc32c>=1.7
  Downloading https://files.pythonhosted.org/packages/ab/82/f60248c01a8a23ae07bd
4c43d78d69b20ffe324311db3b0785e391aa09d2/crc32c-2.0-cp36-cp36m-manylinux1_x86_64
.whl
Requirement already satisfied, skipping upgrade: tensorboard>=1.14.0 in
/tensorflow-1.15.0/python3.6 (from catalyst) (1.15.0)
Requirement already satisfied, skipping upgrade: torchvision>=0.2.1 in
/usr/local/lib/python3.6/dist-packages (from catalyst) (0.5.0)
Requirement already satisfied, skipping upgrade: PyYAML in
/usr/local/lib/python3.6/dist-packages (from catalyst) (3.13)
Collecting Pillow<7
  Downloading https://files.pythonhosted.org/packages/8a/fd/bbbc569f98f478
13c50a116b539d97b3b17a86ac7a309f83b2022d26caf2/Pillow-6.2.2-cp36-cp36m-manylinux
1_x86_64.whl (2.1MB)
     |                  | 2.1MB 38.6MB/s
Requirement already satisfied, skipping upgrade: scikit-image>=0.14.2 in
/usr/local/lib/python3.6/dist-packages (from catalyst) (0.16.2)
Requirement already satisfied, skipping upgrade: opencv-python in
/usr/local/lib/python3.6/dist-packages (from catalyst) (4.1.2.30)
```

```
Collecting deprecation
  Downloading https://files.pythonhosted.org/packages/b9/2a/d5084a8781398cea745c
01237b95d9762c382697c63760a95cc6a814ad3a/deprecation-2.0.7-py2.py3-none-any.whl
Requirement already satisfied, skipping upgrade: packaging in
/usr/local/lib/python3.6/dist-packages (from catalyst) (20.3)
Requirement already satisfied, skipping upgrade: tqdm>=4.33.0 in
/usr/local/lib/python3.6/dist-packages (from catalyst) (4.38.0)
Requirement already satisfied, skipping upgrade: torch>=1.0.0 in
/usr/local/lib/python3.6/dist-packages (from catalyst) (1.4.0)
Requirement already satisfied, skipping upgrade: plotly>=4.1.0 in
/usr/local/lib/python3.6/dist-packages (from catalyst) (4.4.1)
Requirement already satisfied, skipping upgrade: numpy>=1.16.4 in
/usr/local/lib/python3.6/dist-packages (from catalyst) (1.18.2)
Requirement already satisfied, skipping upgrade: pandas>=0.22 in
/usr/local/lib/python3.6/dist-packages (from catalyst) (0.25.3)
Requirement already satisfied, skipping upgrade: imageio in
/usr/local/lib/python3.6/dist-packages (from catalyst) (2.4.1)
Requirement already satisfied, skipping upgrade: seaborn in
/usr/local/lib/python3.6/dist-packages (from catalyst) (0.10.0)
Collecting tensorboardX
  Downloading https://files.pythonhosted.org/packages/35/f1/5843425495765c
8c2dd0784a851a93ef204d314fc87bcc2bbb9f662a3ad1/tensorboardX-2.0-py2.py3-none-
any.whl (195kB)
     |                  | 204kB 36.0MB/s
Collecting GitPython>=2.1.11
  Downloading https://files.pythonhosted.org/packages/d3/2f/6a366d56c9b135
5b0880be9ea66b166cb3536392638d8d91413ec66305ad/GitPython-3.1.0-py3-none-any.whl
(450kB)
     |                  | 460kB 41.2MB/s
Requirement already satisfied, skipping upgrade: scikit-learn>=0.20 in
/usr/local/lib/python3.6/dist-packages (from catalyst) (0.22.2.post1)
Requirement already satisfied, skipping upgrade: matplotlib in
/usr/local/lib/python3.6/dist-packages (from catalyst) (3.2.0)
Requirement already satisfied, skipping upgrade: ipython in
/usr/local/lib/python3.6/dist-packages (from catalyst) (5.5.0)
Requirement already satisfied, skipping upgrade: markdown>=2.6.8 in
/usr/local/lib/python3.6/dist-packages (from tensorboard>=1.14.0->catalyst)
(3.2.1)
Requirement already satisfied, skipping upgrade: absl-py>=0.4 in
/usr/local/lib/python3.6/dist-packages (from tensorboard>=1.14.0->catalyst)
(0.9.0)
Requirement already satisfied, skipping upgrade: wheel>=0.26; python_version >=
"3" in /usr/local/lib/python3.6/dist-packages (from
tensorboard>=1.14.0->catalyst) (0.34.2)
Requirement already satisfied, skipping upgrade: setuptools>=41.0.0 in
/usr/local/lib/python3.6/dist-packages (from tensorboard>=1.14.0->catalyst)
(46.0.0)
Requirement already satisfied, skipping upgrade: protobuf>=3.6.0 in
```

```
/usr/local/lib/python3.6/dist-packages (from tensorboard>=1.14.0->catalyst)
(3.10.0)
Requirement already satisfied, skipping upgrade: six>=1.10.0 in
/usr/local/lib/python3.6/dist-packages (from tensorboard>=1.14.0->catalyst)
(1.12.0)
Requirement already satisfied, skipping upgrade: grpcio>=1.6.3 in
/usr/local/lib/python3.6/dist-packages (from tensorboard>=1.14.0->catalyst)
(1.24.3)
Requirement already satisfied, skipping upgrade: werkzeug>=0.11.15 in
/usr/local/lib/python3.6/dist-packages (from tensorboard>=1.14.0->catalyst)
(1.0.0)
Requirement already satisfied, skipping upgrade: scipy>=0.19.0 in
/usr/local/lib/python3.6/dist-packages (from scikit-image>=0.14.2->catalyst)
(1.4.1)
Requirement already satisfied, skipping upgrade: networkx>=2.0 in
/usr/local/lib/python3.6/dist-packages (from scikit-image>=0.14.2->catalyst)
(2.4)
Requirement already satisfied, skipping upgrade: PyWavelets>=0.4.0 in
/usr/local/lib/python3.6/dist-packages (from scikit-image>=0.14.2->catalyst)
(1.1.1)
Requirement already satisfied, skipping upgrade: pyparsing>=2.0.2 in
/usr/local/lib/python3.6/dist-packages (from packaging->catalyst) (2.4.6)
Requirement already satisfied, skipping upgrade: retrying>=1.3.3 in
/usr/local/lib/python3.6/dist-packages (from plotly>=4.1.0->catalyst) (1.3.3)
Requirement already satisfied, skipping upgrade: pytz>=2017.2 in
/usr/local/lib/python3.6/dist-packages (from pandas>=0.22->catalyst) (2018.9)
Requirement already satisfied, skipping upgrade: python-dateutil>=2.6.1 in
/usr/local/lib/python3.6/dist-packages (from pandas>=0.22->catalyst) (2.8.1)
Collecting gitdb<5,>=4.0.1
  Downloading https://files.pythonhosted.org/packages/1e/f5/8f84b3bf9d94bd
f2454a302f2fa375832b53660ea532586b8a55ff16ae9a/gitdb-4.0.2-py3-none-any.whl
(63kB)
     |                  | 71kB 9.8MB/s
Requirement already satisfied, skipping upgrade: joblib>=0.11 in
/usr/local/lib/python3.6/dist-packages (from scikit-learn>=0.20->catalyst)
(0.14.1)
Requirement already satisfied, skipping upgrade: cycler>=0.10 in
/usr/local/lib/python3.6/dist-packages (from matplotlib->catalyst) (0.10.0)
Requirement already satisfied, skipping upgrade: kiwisolver>=1.0.1 in
/usr/local/lib/python3.6/dist-packages (from matplotlib->catalyst) (1.1.0)
Requirement already satisfied, skipping upgrade: pickleshare in
/usr/local/lib/python3.6/dist-packages (from ipython->catalyst) (0.7.5)
Requirement already satisfied, skipping upgrade: pygments in
/usr/local/lib/python3.6/dist-packages (from ipython->catalyst) (2.1.3)
Requirement already satisfied, skipping upgrade: pexpect; sys_platform !=
"win32" in /usr/local/lib/python3.6/dist-packages (from ipython->catalyst)
(4.8.0)
Requirement already satisfied, skipping upgrade: prompt-toolkit<2.0.0,>=1.0.4 in
```

```
/usr/local/lib/python3.6/dist-packages (from ipython->catalyst) (1.0.18)
Requirement already satisfied, skipping upgrade: simplegeneric>0.8 in
/usr/local/lib/python3.6/dist-packages (from ipython->catalyst) (0.8.1)
Requirement already satisfied, skipping upgrade: traitlets>=4.2 in
/usr/local/lib/python3.6/dist-packages (from ipython->catalyst) (4.3.3)
Requirement already satisfied, skipping upgrade: decorator in
/usr/local/lib/python3.6/dist-packages (from ipython->catalyst) (4.4.2)
Collecting smmap<4,>=3.0.1
  Downloading https://files.pythonhosted.org/packages/35/d2/27777ab463cd44842c78
305fa8097dfba0d94768abbb7e1c4d88f1fa1a0b/smmap-3.0.1-py2.py3-none-any.whl
Requirement already satisfied, skipping upgrade: ptyprocess>=0.5 in
/usr/local/lib/python3.6/dist-packages (from pexpect; sys_platform !=
"win32"->ipython->catalyst) (0.6.0)
Requirement already satisfied, skipping upgrade: wcwidth in
/usr/local/lib/python3.6/dist-packages (from prompt-
toolkit<2.0.0,>=1.0.4->ipython->catalyst) (0.1.8)
Requirement already satisfied, skipping upgrade: ipython-genutils in
/usr/local/lib/python3.6/dist-packages (from traitlets>=4.2->ipython->catalyst)
(0.2.0)
ERROR: albumentations 0.1.12 has requirement imgaug<0.2.7,>=0.2.5, but

you'll have imgaug 0.2.9 which is incompatible.
Installing collected packages: crc32c, Pillow, deprecation, tensorboardX, smmap,
gitdb, GitPython, catalyst
  Found existing installation: Pillow 7.0.0
    Uninstalling Pillow-7.0.0:
      Successfully uninstalled Pillow-7.0.0
Successfully installed GitPython-3.1.0 Pillow-6.2.2 catalyst-20.3.2 crc32c-2.0
deprecation-2.0.7 gitdb-4.0.2 smmap-3.0.1 tensorboardX-2.0
```

[0]:
```python
# Dependencies

# Handles data
import json
import numpy as np
import matplotlib.pyplot as plt
import cv2
import glob
from operator import itemgetter

# Torch utilities
from typing import List
from pathlib import Path
from torch.utils.data import Dataset
import torch

# Data Loader utilities
import collections
```

```
from sklearn.model_selection import train_test_split

from torch.utils.data import DataLoader

# Model building and training
import segmentation_models_pytorch as smp
from torch import nn

from catalyst.contrib.nn import DiceLoss, IoULoss
from torch import optim
from catalyst import utils

from catalyst.contrib.nn import RAdam, Lookahead
from catalyst.dl import SupervisedRunner

from catalyst.dl.callbacks import DiceCallback, IouCallback, \
  CriterionCallback, AccuracyCallback, MulticlassDiceMetricCallback
```

[0]:
```
# Sets a seed for better reproducibility
SEED = 42
utils.set_global_seed(SEED)
utils.prepare_cudnn(deterministic=True)
```

```
<IPython.core.display.HTML object>
```

[0]:
```
# Defines and establishes a dataset class
class SegmentationDataset(Dataset):
    def __init__(
        self,
        image_arr_path,
        mask_arr_path,
    ) -> None:
        self.images = np.load(image_arr_path)
        self.masks = np.load(mask_arr_path)

    def __len__(self) -> int:
        return len(self.images)

    def __getitem__(self, idx: int) -> dict:
        image = self.images[idx]
        image = np.swapaxes(image, 2, 0)
        image = np.swapaxes(image, 2, 1)
        image = torch.from_numpy(image).float()
        result = {"image": image}

        if self.masks is not None:
```

```
            mask = self.masks[idx]
            mask = np.swapaxes(mask, 2, 0)
            mask = np.swapaxes(mask, 2, 1)
            mask = torch.from_numpy(mask).float()
            result["mask"] = mask
        return result
```

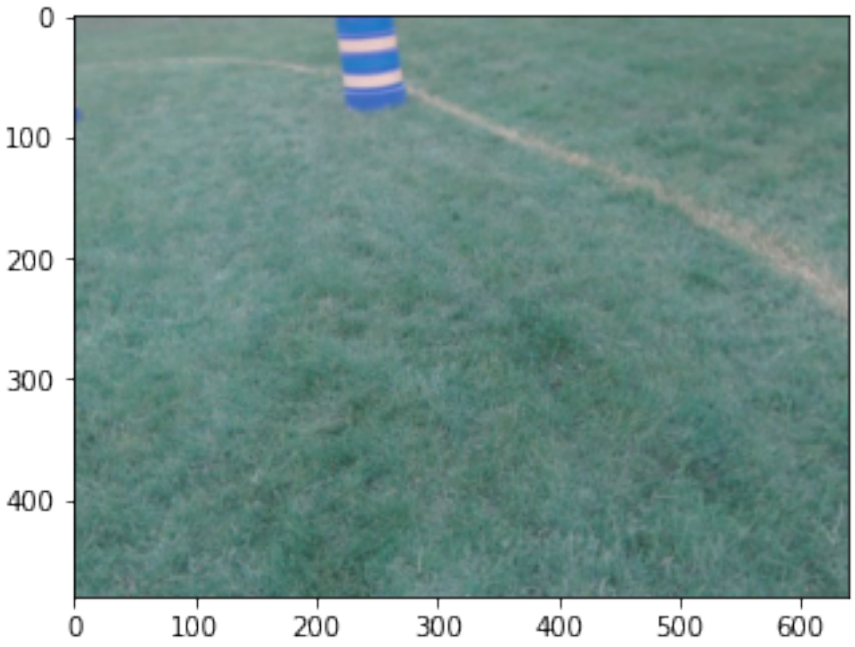
[0]:
```
# Loading once to enable visualizion prior to model training
dset = SegmentationDataset(image_arr_path="/content/drive/My Drive/RoboJackets/
 ↪Split_Data/train_images.npy", mask_arr_path="/content/drive/My Drive/
 ↪RoboJackets/Split_Data/train_masks.npy")
```

[0]:
```
# Show sizes of the image and mask
out = dset[0]
out["image"].shape, out["mask"].shape, len(dset)
```

[0]:
```
(torch.Size([3, 480, 640]), torch.Size([1, 480, 640]), 592)
```

[0]:
```
# Show an image
show_image = np.asarray(dset[40]['image'])
show_image = np.swapaxes(show_image, 2, 0)
show_image = np.swapaxes(show_image, 1, 0)
show_image = show_image.astype(np.uint8)
np.shape(show_image)
plt.imshow(show_image)
```

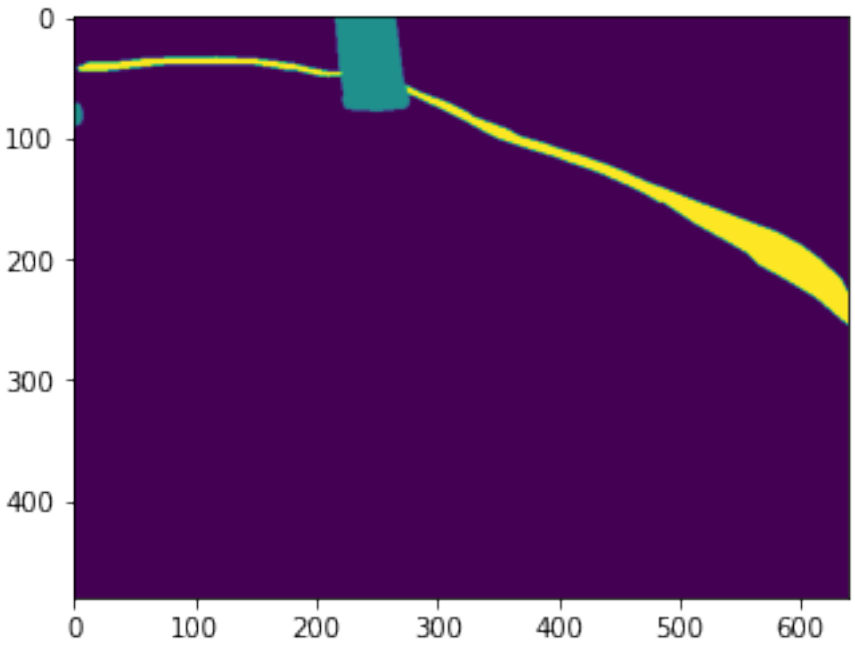
[0]:
```
<matplotlib.image.AxesImage at 0x7ff0343bdfd0>
```



[0]:
```
# Show associated mask

show_mask = np.squeeze(dset[40]['mask'])
plt.imshow(show_mask)
```

[0]:
```
<matplotlib.image.AxesImage at 0x7ff0736afef0>
```



[0]:
```
# Set up transfer learning system

ENCODER = 'efficientnet-b3'
ENCODER_WEIGHTS = 'imagenet'
DEVICE = 'cuda'

# ACTIVATION = 'softmax'
ACTIVATION = None

model = smp.Unet(
    encoder_name=ENCODER,
    encoder_weights=ENCODER_WEIGHTS,
    classes=3,
    activation=ACTIVATION,
)
```

```
# preprocessing_fn = smp.encoders.get_preprocessing_fn(ENCODER, ENCODER_WEIGHTS)
```

[0]:
```
# Define loaders

def get_loaders(
    images: List[Path],
    masks: List[Path],
    image_arr_path: str,
    mask_arr_path: str,
    random_state: int,
    valid_size: float = 0.1,
    batch_size: int = 12,
    num_workers: int = 4,
    # train_transforms_fn = None,
    # valid_transforms_fn = None,
    ) -> dict:

    indices = np.arange(len(images))

    train_indices, valid_indices = train_test_split(
      indices, test_size=valid_size, random_state=random_state, shuffle=True
    )

    np_images = np.array(images)
    np_masks = np.array(masks)

    #print(np_images.shape, np_masks.shape)
    #print(train_indices)

    train_dataset = SegmentationDataset(image_arr_path, mask_arr_path)
    train_dataset.images = np_images[train_indices]
    train_dataset.masks = np_masks[train_indices]
    #print(len(train_dataset))
    #print(train_dataset.images.shape)
    #print(train_dataset.masks.shape)

    valid_dataset = SegmentationDataset(image_arr_path, mask_arr_path)
    valid_dataset.images = np_images[valid_indices]
    valid_dataset.masks = np_masks[valid_indices]
    #print(len(valid_dataset))
    #print(valid_dataset.images.shape)
    #print(valid_dataset.masks.shape)


    train_loader = DataLoader(
      train_dataset,
      batch_size=batch_size,
```

```
      shuffle=False,
      num_workers=num_workers,
      drop_last=False,
    )

    valid_loader = DataLoader(
      valid_dataset,
      batch_size=batch_size,
      shuffle=False,
      num_workers=num_workers,
      drop_last=False,
    )

    loaders = collections.OrderedDict()
    loaders["train"] = train_loader
    loaders["valid"] = valid_loader

    return loaders
```

[0]:
```
# Get loaders
loaders = get_loaders(
    images=np.load("/content/drive/My Drive/RoboJackets/Split_Data/train_images.
 ↪npy"),
    masks=np.load("/content/drive/My Drive/RoboJackets/Split_Data/train_masks.
 ↪npy"),
    image_arr_path="/content/drive/My Drive/RoboJackets/Split_Data/train_images.
 ↪npy",
    mask_arr_path="/content/drive/My Drive/RoboJackets/Split_Data/train_masks.
 ↪npy",
    random_state=420,
    valid_size=0.1,
    batch_size=3,
    num_workers=2,
)
```

```
(592, 480, 640, 3) (592, 480, 640, 1)
[412  85 558 199 435  91 212  43  47 245 400 480 500 347 337 202 142  64
  45 455 149  70 173 384 219 227 176 120 208 572  80 555 263 420 378  11
  40  36  69 254 130 410 554 121 548  84 411 195  49 427  12 503 339 128
 297  82  61 445 491 273 467 260 537 164 391  48 562  65  41 541 213 580
 233 413 416  90   0 285 458 497 311 426 526 472 255 259 116 134 204 579
 186 281 225 383 459  25 545  95 470  32 529 156  44 333 162 181 372 544
 494 550 531 387 161 557 179 473 271 356  13 180 570 417 286 177 381 140
 106 457 553 418 581 250 589 137 210 117 315 546 486 346 460 144 159 287
 340 290 489 216 275  19 135 355 207 229 215 591 523 436  83 112 167 328
 133 113 319 214  56  86 364 584 336 561 122 375  53 289 477 353 332 101
 222 320  34 183 343  79 404 123  18 453 362  87 228 155  67 291 165 351
```

```
 153 349  42  23 124 429 270 143  71 590 476  17 567  57 508 171 348 540
 514 444 421 578 395 365 475 110 262 312 466  78 484 350 288 274  29 236
 576 114 295 170 166 205 193 585 490 283 401 511 380 419 536 582 298   7
 518 240 573  21 322 430 423  24 256 432  20  15 509 111  97 438 376  26
 515 139  31 342 542 146 402 530 408 428 163 501 234 556 249  35 463 498
 104 329  66 434 409   2 118 507  55 502 310   6 253 373 483 564 524 251
 293 451 330   4 493 454 257  62 358 533 172 547  94 538 393 587 235 568
 248  33 175 325  39 187 126  77 354  38 247 276 334  75 148   5 539 150
 437 261 127 469 450 226 299 318 341  68 496 488 528 506 324  89  14 394
 267 447  74 583 399 513 335  46 382 302 359  22 244 105 258 345 178 499
  99 505 252 327 482 115 174 189  16 296 326 282 269 439  54 492  27 521
 396 405 232 566 465 192 462 217 243 371 456 520  51 313 389 203 100 471
 125 102 363 218  52 138 534 168 415 323 268 440 169  81 481 504 532 368
 495 145 294 188 385 209 158 360  37  30 279 338 160 221 303 109 571  93
 388  58 238 301  10 321 292 551 516  88 224   9 306 563 560 407 510 151
 397 277 200 485   8 422 468 414 304 190 565 425 331 552 374 317 424 361
   3 237 527 449 398  63  92  98 370 369 196 103 223  73 344 569 379 136
 431 367 535  28 461 308 246 517  59 377 386 392 107 543  76 559 197 549
 448 182 152 230 266 154 185 575 390  72]
(532, 480, 640, 3)
(532, 480, 640, 1)
(60, 480, 640, 3)
(60, 480, 640, 1)
```

[0]:

```
# Helpful code taken from ---
#
#    Copyright 2019 Division of Medical Image Computing, German Cancer Research Center (DKFZ), Heidelberg, Germany
#
#    Licensed under the Apache License, Version 2.0 (the "License");
#    you may not use this file except in compliance with the License.
#    You may obtain a copy of the License at
#
#        http://www.apache.org/licenses/LICENSE-2.0
#
#    Unless required by applicable law or agreed to in writing, software
#    distributed under the License is distributed on an "AS IS" BASIS,
#    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#    See the License for the specific language governing permissions and
#    limitations under the License.
import torch
from torch import nn
import numpy as np

def sum_tensor(inp, axes, keepdim=False):
    axes = np.unique(axes).astype(int)
    if keepdim:
```

```
        for ax in axes:
            inp = inp.sum(int(ax), keepdim=True)
    else:
        for ax in sorted(axes, reverse=True):
            inp = inp.sum(int(ax))
    return inp

def softmax_helper(x):
    rpt = [1 for _ in range(len(x.size()))]
    rpt[1] = x.size(1)
    x_max = x.max(1, keepdim=True)[0].repeat(*rpt)
    e_x = torch.exp(x - x_max)
    return e_x / e_x.sum(1, keepdim=True).repeat(*rpt)

class CrossentropyND(nn.CrossEntropyLoss):
    """
    Network has to have NO NONLINEARITY!
    """
    def forward(self, inp, target):
        target = target.long()
        num_classes = inp.size()[1]

        i0 = 1
        i1 = 2

        while i1 < len(inp.shape): # this is ugly but torch only allows to␣
↪transpose two axes at once
            inp = inp.transpose(i0, i1)
            i0 += 1
            i1 += 1

        inp = inp.contiguous()
        inp = inp.view(-1, num_classes)

        target = target.view(-1,)

        return super(CrossentropyND, self).forward(inp, target)

def get_tp_fp_fn(net_output, gt, axes=None, mask=None, square=False):
    """
    net_output must be (b, c, x, y(, z)))
    gt must be a label map (shape (b, 1, x, y(, z)) OR shape (b, x, y(, z))) or␣
↪one hot encoding (b, c, x, y(, z))
    if mask is provided it must have shape (b, 1, x, y(, z)))
    :param net_output:
    :param gt:
    :param axes:
```

```
    :param mask: mask must be 1 for valid pixels and 0 for invalid pixels
    :param square: if True then fp, tp and fn will be squared before summation
    :return:
    """
    if axes is None:
        axes = tuple(range(2, len(net_output.size())))

    shp_x = net_output.shape
    shp_y = gt.shape

    with torch.no_grad():
        if len(shp_x) != len(shp_y):
            gt = gt.view((shp_y[0], 1, *shp_y[1:]))

        if all([i == j for i, j in zip(net_output.shape, gt.shape)]):
            # if this is the case then gt is probably already a one hot encoding
            y_onehot = gt
        else:
            gt = gt.long()
            y_onehot = torch.zeros(shp_x)
            if net_output.device.type == "cuda":
                y_onehot = y_onehot.cuda(net_output.device.index)
            y_onehot.scatter_(1, gt, 1)

    tp = net_output * y_onehot
    fp = net_output * (1 - y_onehot)
    fn = (1 - net_output) * y_onehot

    if mask is not None:
        tp = torch.stack(tuple(x_i * mask[:, 0] for x_i in torch.unbind(tp,␣
↪dim=1)), dim=1)
        fp = torch.stack(tuple(x_i * mask[:, 0] for x_i in torch.unbind(fp,␣
↪dim=1)), dim=1)
        fn = torch.stack(tuple(x_i * mask[:, 0] for x_i in torch.unbind(fn,␣
↪dim=1)), dim=1)

    if square:
        tp = tp ** 2
        fp = fp ** 2
        fn = fn ** 2

    tp = sum_tensor(tp, axes, keepdim=False)
    fp = sum_tensor(fp, axes, keepdim=False)
    fn = sum_tensor(fn, axes, keepdim=False)

    return tp, fp, fn
```

```
class SoftDiceLoss(nn.Module):
    def __init__(self, apply_nonlin=None, batch_dice=False, do_bg=True,
                 smooth=1., square=False):
        """
        """
        super(SoftDiceLoss, self).__init__()

        self.square = square
        self.do_bg = do_bg
        self.batch_dice = batch_dice
        self.apply_nonlin = apply_nonlin
        self.smooth = smooth

    def forward(self, x, y, loss_mask=None):
        shp_x = x.shape

        if self.batch_dice:
            axes = [0] + list(range(2, len(shp_x)))
        else:
            axes = list(range(2, len(shp_x)))

        if self.apply_nonlin is not None:
            x = self.apply_nonlin(x)

        tp, fp, fn = get_tp_fp_fn(x, y, axes, loss_mask, self.square)

        dc = (2 * tp + self.smooth) / (2 * tp + fp + fn + self.smooth)

        if not self.do_bg:
            if self.batch_dice:
                dc = dc[1:]
            else:
                dc = dc[:, 1:]
        dc = dc.mean()

        return -dc


class DC_and_CE_loss(nn.Module):
    def __init__(self, soft_dice_kwargs, ce_kwargs, aggregate="sum"):
        super(DC_and_CE_loss, self).__init__()
        self.aggregate = aggregate
        self.ce = CrossentropyND(**ce_kwargs)
        self.dc = SoftDiceLoss(apply_nonlin=softmax_helper, **soft_dice_kwargs)

    def forward(self, net_output, target):
```

```
        dc_loss = self.dc(net_output, target)
        ce_loss = self.ce(net_output, target)
        if self.aggregate == "sum":
            result = ce_loss + dc_loss
        else:
            raise NotImplementedError("nah son") # reserved for other stuff␣
 ↪(later)
        return result
```

[0]:
```
# Define loss criterion
criterion = {
    "CE": CrossentropyND(),
}

from torch.optim import AdamW

# Set up optimization

learning_rate = 0.001 #0.001
encoder_learning_rate = 0.0005
encoder_weight_decay = 0.00003 #0.00003
optimizer_weight_decay = 0.0003 #0.0003
optim_factor = 0.25 #0.25
optim_patience = 2 #2

optimizer = AdamW(model.parameters(), lr=0.001, betas=(0.9, 0.999), eps=1e-08,␣
 ↪weight_decay=0.01, amsgrad=False)

scheduler = optim.lr_scheduler.ReduceLROnPlateau(optimizer,␣
 ↪factor=optim_factor, patience=optim_patience)

num_epochs = 10
device = utils.get_device()

runner = SupervisedRunner(device=device, input_key="image",␣
 ↪input_target_key="mask")
```

[0]:
```
callbacks = [
        CriterionCallback(
            input_key="mask",
            prefix="loss",
            criterion_key="CE"
        ),
        MulticlassDiceMetricCallback(input_key="mask")
        ]
```

[0]:
```
runner.train(
    model=model,
    criterion=criterion,
    optimizer=optimizer,
    scheduler=scheduler,
    loaders=loaders,
    callbacks=callbacks,
    logdir='content/full_model2', #this logdir must be changed with every new␣
 ↪run
    num_epochs=num_epochs,

    main_metric="loss",
    minimize_metric=True,
    fp16=None,

    monitoring_params=None,

    verbose=True,
)
```

```
1/10 * Epoch (train): 100% 178/178 [01:41<00:00,  1.75it/s, _timer/_fps=0.821,
_timer/batch_time=3.654, _timer/data_time=0.312, _timer/model_time=3.341,
loss=0.053]
1/10 * Epoch (valid): 100% 20/20 [00:03<00:00,  6.29it/s, _timer/_fps=20.898,
_timer/batch_time=0.144, _timer/data_time=0.012, _timer/model_time=0.131,
loss=0.112]
[2020-03-20 03:33:53,038]
1/10 * Epoch 1 (train): _timer/_fps=6.3847 | _timer/batch_time=0.5372 |
_timer/data_time=0.3472 | _timer/model_time=0.1900 | dice_0=0.9635 |
dice_1=0.0130 | dice_2=0.5910 | dice_mean=0.5225 | loss=0.2064
1/10 * Epoch 1 (valid): _timer/_fps=19.6645 | _timer/batch_time=0.1569 |
_timer/data_time=0.0235 | _timer/model_time=0.1333 | dice_0=0.9769 |
dice_1=0.5987 | dice_2=0.6905 | dice_mean=0.7554 | loss=0.1282
2/10 * Epoch (train): 100% 178/178 [01:22<00:00,  2.15it/s, _timer/_fps=7.819,
_timer/batch_time=0.384, _timer/data_time=0.286, _timer/model_time=0.098,
loss=0.044]
2/10 * Epoch (valid): 100% 20/20 [00:03<00:00,  5.98it/s, _timer/_fps=20.212,
_timer/batch_time=0.148, _timer/data_time=0.013, _timer/model_time=0.136,
loss=0.073]
[2020-03-20 03:35:21,168]
2/10 * Epoch 2 (train): _timer/_fps=6.4760 | _timer/batch_time=0.4641 |
_timer/data_time=0.3083 | _timer/model_time=0.1557 | dice_0=0.9895 |
dice_1=0.5373 | dice_2=0.8154 | dice_mean=0.7808 | loss=0.0591
2/10 * Epoch 2 (valid): _timer/_fps=18.6243 | _timer/batch_time=0.1648 |
_timer/data_time=0.0237 | _timer/model_time=0.1412 | dice_0=0.9869 |
dice_1=0.6184 | dice_2=0.7951 | dice_mean=0.8001 | loss=0.0757
3/10 * Epoch (train): 100% 178/178 [01:22<00:00,  2.15it/s, _timer/_fps=7.283,
```

```
_timer/batch_time=0.412, _timer/data_time=0.305, _timer/model_time=0.107,
loss=0.037]
3/10 * Epoch (valid): 100% 20/20 [00:03<00:00,  6.26it/s, _timer/_fps=21.101,
_timer/batch_time=0.142, _timer/data_time=0.011, _timer/model_time=0.131,
loss=0.039]
[2020-03-20 03:36:49,210]
3/10 * Epoch 3 (train): _timer/_fps=6.4597 | _timer/batch_time=0.4649 |
_timer/data_time=0.3103 | _timer/model_time=0.1546 | dice_0=0.9910 |
dice_1=0.6875 | dice_2=0.8353 | dice_mean=0.8379 | loss=0.0467
3/10 * Epoch 3 (valid): _timer/_fps=19.6169 | _timer/batch_time=0.1574 |
_timer/data_time=0.0193 | _timer/model_time=0.1380 | dice_0=0.9890 |
dice_1=0.8331 | dice_2=0.8027 | dice_mean=0.8749 | loss=0.0630
4/10 * Epoch (train): 100% 178/178 [01:22<00:00,  2.15it/s, _timer/_fps=7.647,
_timer/batch_time=0.392, _timer/data_time=0.298, _timer/model_time=0.094,
loss=0.034]
4/10 * Epoch (valid): 100% 20/20 [00:03<00:00,  6.08it/s, _timer/_fps=21.047,
_timer/batch_time=0.143, _timer/data_time=0.013, _timer/model_time=0.130,
loss=0.031]
[2020-03-20 03:38:17,556]
4/10 * Epoch 4 (train): _timer/_fps=6.4678 | _timer/batch_time=0.4644 |
_timer/data_time=0.3086 | _timer/model_time=0.1558 | dice_0=0.9927 |
dice_1=0.7995 | dice_2=0.8587 | dice_mean=0.8837 | loss=0.0378
4/10 * Epoch 4 (valid): _timer/_fps=19.1945 | _timer/batch_time=0.1616 |
_timer/data_time=0.0231 | _timer/model_time=0.1385 | dice_0=0.9900 |
dice_1=0.9001 | dice_2=0.8102 | dice_mean=0.9001 | loss=0.0585
5/10 * Epoch (train): 100% 178/178 [01:22<00:00,  2.16it/s, _timer/_fps=7.183,
_timer/batch_time=0.418, _timer/data_time=0.318, _timer/model_time=0.100,
loss=0.029]
5/10 * Epoch (valid): 100% 20/20 [00:03<00:00,  6.25it/s, _timer/_fps=20.178,
_timer/batch_time=0.149, _timer/data_time=0.012, _timer/model_time=0.137,
loss=0.024]
[2020-03-20 03:39:45,143]
5/10 * Epoch 5 (train): _timer/_fps=6.4919 | _timer/batch_time=0.4626 |
_timer/data_time=0.3076 | _timer/model_time=0.1550 | dice_0=0.9939 |
dice_1=0.8967 | dice_2=0.8694 | dice_mean=0.9200 | loss=0.0325
5/10 * Epoch 5 (valid): _timer/_fps=19.5679 | _timer/batch_time=0.1575 |
_timer/data_time=0.0209 | _timer/model_time=0.1366 | dice_0=0.9879 |
dice_1=0.8508 | dice_2=0.7516 | dice_mean=0.8635 | loss=0.0770
6/10 * Epoch (train): 100% 178/178 [01:22<00:00,  2.14it/s, _timer/_fps=7.483,
_timer/batch_time=0.401, _timer/data_time=0.308, _timer/model_time=0.093,
loss=0.023]
6/10 * Epoch (valid): 100% 20/20 [00:03<00:00,  6.27it/s, _timer/_fps=20.726,
_timer/batch_time=0.145, _timer/data_time=0.011, _timer/model_time=0.134,
loss=0.023]
[2020-03-20 03:41:12,348]
6/10 * Epoch 6 (train): _timer/_fps=6.4583 | _timer/batch_time=0.4650 |
_timer/data_time=0.3094 | _timer/model_time=0.1556 | dice_0=0.9944 |
dice_1=0.9358 | dice_2=0.8749 | dice_mean=0.9350 | loss=0.0297
```

```
6/10 * Epoch 6 (valid): _timer/_fps=19.6176 | _timer/batch_time=0.1571 |
_timer/data_time=0.0196 | _timer/model_time=0.1375 | dice_0=0.9893 |
dice_1=0.9125 | dice_2=0.7859 | dice_mean=0.8959 | loss=0.0633
7/10 * Epoch (train): 100% 178/178 [01:22<00:00,  2.15it/s, _timer/_fps=7.636,
_timer/batch_time=0.393, _timer/data_time=0.289, _timer/model_time=0.104,
loss=0.028]
7/10 * Epoch (valid): 100% 20/20 [00:03<00:00,  6.30it/s, _timer/_fps=20.504,
_timer/batch_time=0.146, _timer/data_time=0.012, _timer/model_time=0.135,
loss=0.034]
[2020-03-20 03:42:39,445]
7/10 * Epoch 7 (train): _timer/_fps=6.4820 | _timer/batch_time=0.4635 |
_timer/data_time=0.3078 | _timer/model_time=0.1558 | dice_0=0.9946 |
dice_1=0.9381 | dice_2=0.8802 | dice_mean=0.9376 | loss=0.0281
7/10 * Epoch 7 (valid): _timer/_fps=19.7017 | _timer/batch_time=0.1563 |
_timer/data_time=0.0187 | _timer/model_time=0.1376 | dice_0=0.9833 |
dice_1=0.5378 | dice_2=0.7334 | dice_mean=0.7515 | loss=0.1073
8/10 * Epoch (train): 100% 178/178 [01:22<00:00,  2.15it/s, _timer/_fps=7.823,
_timer/batch_time=0.383, _timer/data_time=0.292, _timer/model_time=0.091,
loss=0.022]
8/10 * Epoch (valid): 100% 20/20 [00:03<00:00,  6.25it/s, _timer/_fps=21.096,
_timer/batch_time=0.142, _timer/data_time=0.012, _timer/model_time=0.130,
loss=0.022]
[2020-03-20 03:44:06,497]
8/10 * Epoch 8 (train): _timer/_fps=6.4862 | _timer/batch_time=0.4632 |
_timer/data_time=0.3083 | _timer/model_time=0.1549 | dice_0=0.9942 |
dice_1=0.9063 | dice_2=0.8724 | dice_mean=0.9243 | loss=0.0305
8/10 * Epoch 8 (valid): _timer/_fps=19.5776 | _timer/batch_time=0.1573 |
_timer/data_time=0.0196 | _timer/model_time=0.1377 | dice_0=0.9876 |
dice_1=0.8488 | dice_2=0.7385 | dice_mean=0.8583 | loss=0.0897
9/10 * Epoch (train): 100% 178/178 [01:22<00:00,  2.15it/s, _timer/_fps=7.189,
_timer/batch_time=0.417, _timer/data_time=0.313, _timer/model_time=0.104,
loss=0.019]
9/10 * Epoch (valid): 100% 20/20 [00:03<00:00,  6.15it/s, _timer/_fps=20.706,
_timer/batch_time=0.145, _timer/data_time=0.012, _timer/model_time=0.133,
loss=0.022]
[2020-03-20 03:45:33,709]
9/10 * Epoch 9 (train): _timer/_fps=6.4782 | _timer/batch_time=0.4637 |
_timer/data_time=0.3089 | _timer/model_time=0.1547 | dice_0=0.9951 |
dice_1=0.9631 | dice_2=0.8868 | dice_mean=0.9483 | loss=0.0245
9/10 * Epoch 9 (valid): _timer/_fps=19.2671 | _timer/batch_time=0.1602 |
_timer/data_time=0.0216 | _timer/model_time=0.1386 | dice_0=0.9901 |
dice_1=0.9512 | dice_2=0.8037 | dice_mean=0.9150 | loss=0.0617
10/10 * Epoch (train): 100% 178/178 [01:22<00:00,  2.15it/s, _timer/_fps=7.131,
_timer/batch_time=0.421, _timer/data_time=0.317, _timer/model_time=0.103,
loss=0.018]
10/10 * Epoch (valid): 100% 20/20 [00:03<00:00,  6.21it/s, _timer/_fps=20.265,
_timer/batch_time=0.148, _timer/data_time=0.013, _timer/model_time=0.135,
loss=0.019]
```

```
[2020-03-20 03:47:01,070]
10/10 * Epoch 10 (train): _timer/_fps=6.4627 | _timer/batch_time=0.4648 |
_timer/data_time=0.3094 | _timer/model_time=0.1554 | dice_0=0.9955 |
dice_1=0.9705 | dice_2=0.8971 | dice_mean=0.9544 | loss=0.0218
10/10 * Epoch 10 (valid): _timer/_fps=19.4141 | _timer/batch_time=0.1585 |
_timer/data_time=0.0214 | _timer/model_time=0.1371 | dice_0=0.9904 |
dice_1=0.9289 | dice_2=0.8159 | dice_mean=0.9117 | loss=0.0619
Top best models:
content/full_model2/checkpoints/train.4.pth     0.0585
```

[0]:
```
# Test model on test dataset
test_data = SegmentationDataset("/content/drive/My Drive/RoboJackets/Split_Data/
 ↪test_images.npy", "/content/drive/My Drive/RoboJackets/Split_Data/test_masks.
 ↪npy")
```

[0]:
```
infer_loader = DataLoader(
    test_data,
    batch_size=12,
    shuffle=False,
    num_workers=4
)
```

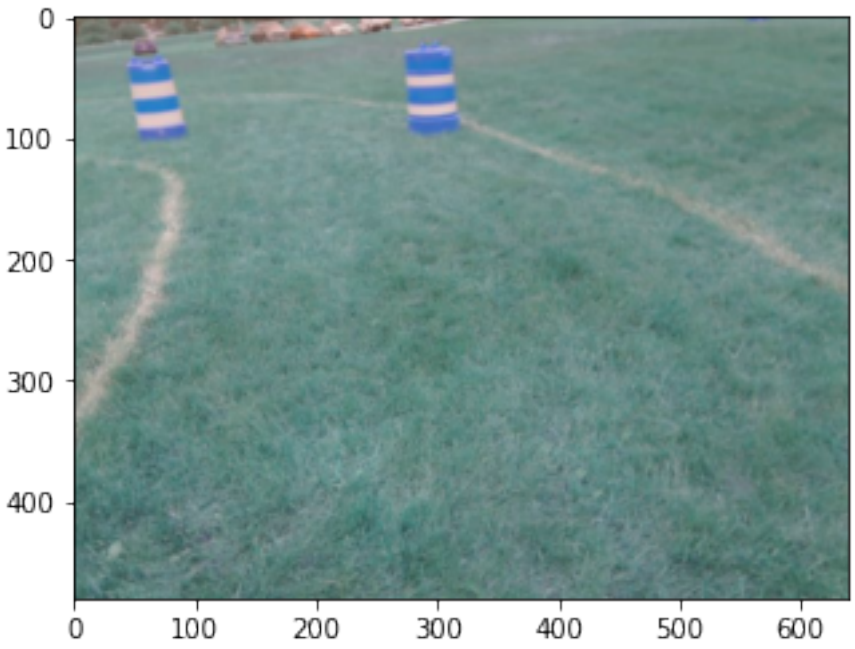
[0]:
```
# get predictions on test data
predictions = runner.predict_loader(
    model=model,
    loader=infer_loader,
    resume=f"content/full_model2/checkpoints/best.pth",
    verbose=False,
)

print(type(predictions))
print(predictions.shape)
```

```
=> loading checkpoint content/full_model2/checkpoints/best.pth
loaded checkpoint content/full_model2/checkpoints/best.pth (global epoch 4,
epoch 4, stage train)
<class 'numpy.ndarray'>
(149, 3, 480, 640)
```

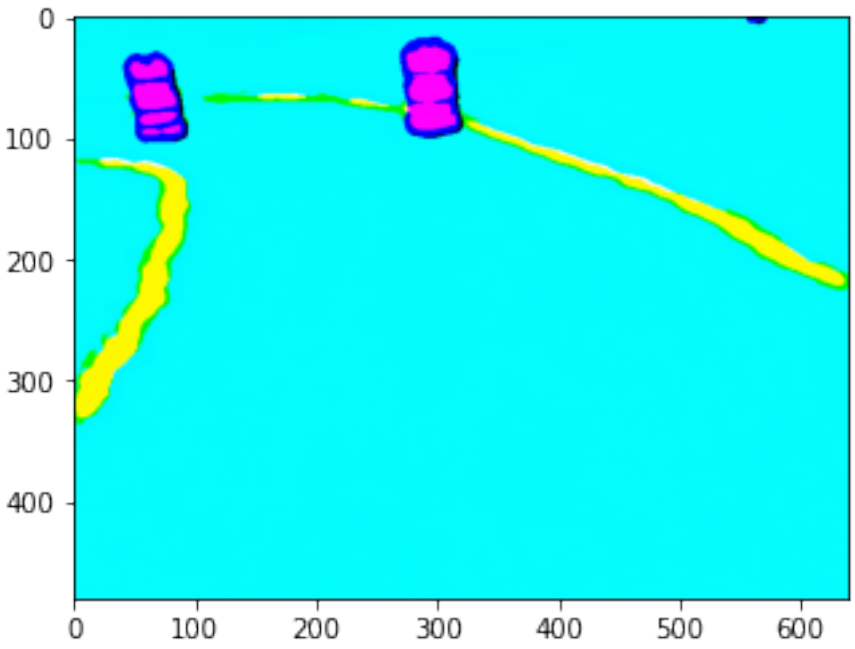
[0]:
```
show_image = np.asarray(test_data[30]['image'])
show_image = np.swapaxes(show_image, 2, 0)
show_image = np.swapaxes(show_image, 1, 0)
show_image = show_image.astype(np.uint8)
np.shape(show_image)
plt.imshow(show_image)
```

[0]:
```
<matplotlib.image.AxesImage at 0x7fefe3a8a4e0>
```



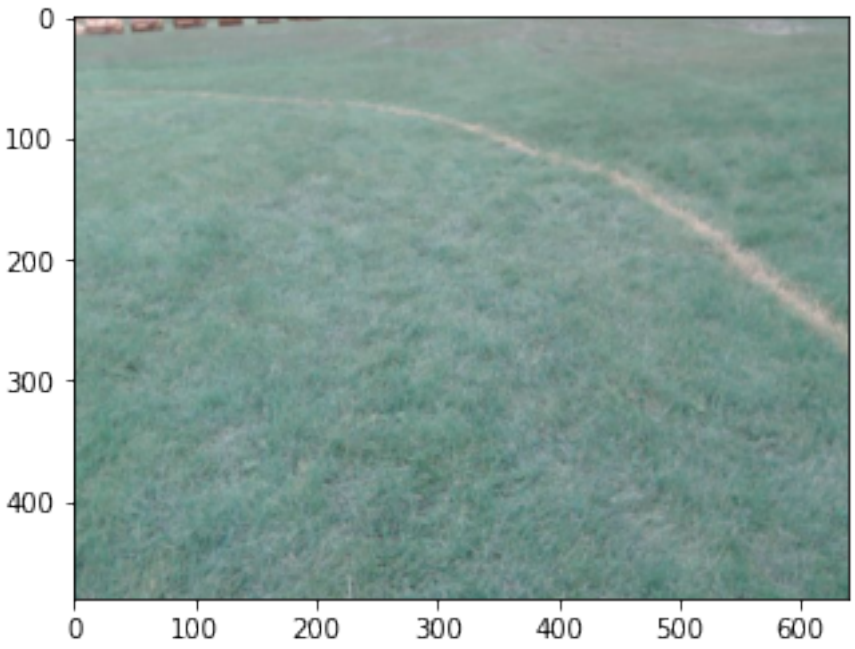
[0]:
```
show_image = np.asarray(predictions[30])
show_image = np.swapaxes(show_image, 2, 0)
show_image = np.swapaxes(show_image, 1, 0)
show_image = show_image.astype(np.uint8)
np.shape(show_image)
plt.imshow(show_image)
```

[0]:
```
<matplotlib.image.AxesImage at 0x7fefe39e4b38>
```



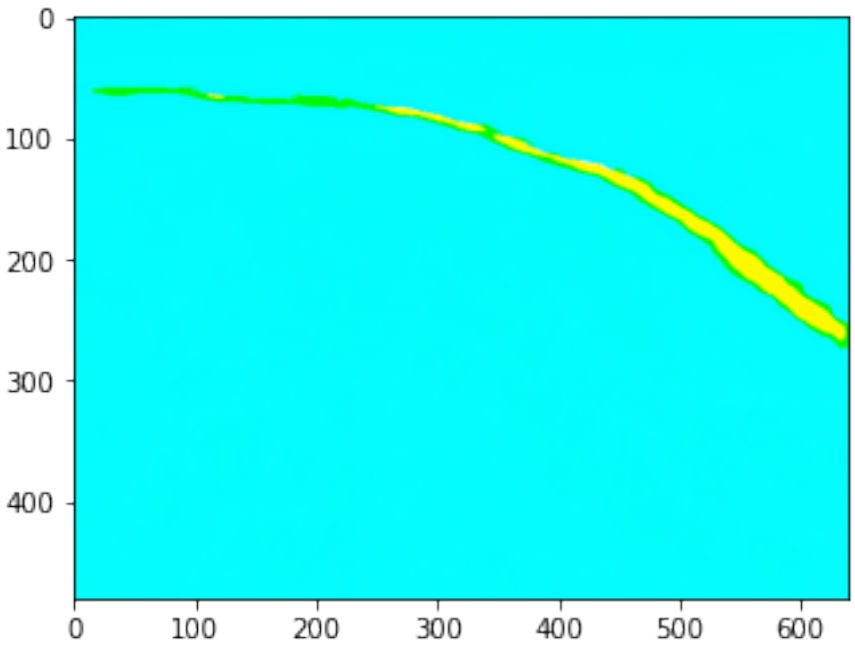
[0]:
```
show_image = np.asarray(test_data[19]['image'])
show_image = np.swapaxes(show_image, 2, 0)
show_image = np.swapaxes(show_image, 1, 0)
show_image = show_image.astype(np.uint8)
np.shape(show_image)
plt.imshow(show_image)
```

[0]:
```
<matplotlib.image.AxesImage at 0x7fefe38de978>
```



[0]:
```
show_image = np.asarray(predictions[19])
show_image = np.swapaxes(show_image, 2, 0)
show_image = np.swapaxes(show_image, 1, 0)
show_image = show_image.astype(np.uint8)
np.shape(show_image)
plt.imshow(show_image)
```

[0]:
```
<matplotlib.image.AxesImage at 0x7fefe3820748>
```



[0]:
```
%load_ext tensorboard
%tensorboard --logdir {'content/full_model2'}
```

```
<IPython.core.display.HTML object>
```

[0]:
```
LOG_DIR = './content/full_model2'

get_ipython().system_raw(
    'tensorboard --logdir {} --host 0.0.0.0 --port 6007 &'
    .format(LOG_DIR)
)
# Install
! npm install -g localtunnel

# Tunnel port 6006 (TensorBoard assumed running)
get_ipython().system_raw('lt --port 6007 >> url.txt 2>&1 &')

# Get url
! cat url.txt
```

```
/tools/node/bin/lt ->
/tools/node/lib/node_modules/localtunnel/bin/lt.js
+ localtunnel@2.0.0
updated 1 package in 2.06s
```

```
Update available 5.7.1 → 6.14.3

   Run npm i -g npm to update
```

[0]:
```
!npm i -g npm
```

```
/tools/node/bin/npm -> /tools/node/lib/node_modules/npm/bin/npm-cli.js
/tools/node/bin/npx -> /tools/node/lib/node_modules/npm/bin/npx-cli.js
+ npm@6.14.3
added 325 packages from 161 contributors, removed 423 packages and updated 59
packages in 14.469s
```