```
In [1]: import pprint
        import rasterio
        from rasterio import features
        import rasterio.warp
        import matplotlib.pyplot as plt
        import numpy as np
        from scipy.ndimage import zoom
        import os
        import torch
        import torch.nn as nn
        import torch.nn.functional as F
        import torch.optim as optim
        import torchvision
        from torchvision import datasets, transforms
        from torch.utils.data import Dataset
        from torch.optim.lr_scheduler import StepLR
```

```
In [2]: # Base values
        channel_descriptions = ['M11', 'I2', 'I1', 'NDVI_last', 'EVI2_last', 'total precipi
        min_values = [np.float32(-100.0),
          np.float32(-100.0),
          np.float32(-100.0),
          np.float32(-9863.268),
          np.float32(-4422.217),
          np.float32(0.0),
          np.float32(0.0),
          np.float32(0.0),
          np.float32(0.0),
          np.float32(0.0),
          np.float32(0.0),
          np.float32(0.0),
          np.float32(0.0),
          np.float32(0.0),
          np.float32(-84.0),
          np.float32(-6.72),
          np.float32(1.0),
          np.float32(0.0),
          np.float32(0.0),
          np.float32(-89.999214),
          np.float32(-13.984883),
          np.float32(0.0),
          np.float32(0.0)]
        max_values = [np.float32(15976.0),
          np.float32(15799.0),
          np.float32(15744.0),
          np.float32(9975.073),
          np.float32(9856.787),
          np.float32(122.0),
          np.float32(16.2),
          np.float32(360.0),
          np.float32(311.8),
```

localhost:8888/lab 1/152

```
np.float32(325.4),
np.float32(122.0),
np.float32(0.01888),
np.float32(63.85685),
np.float32(359.42383),
np.float32(4268.336),
np.float32(8.28),
np.float32(17.0),
np.float32(204.1875),
np.float32(14.295916),
np.float32(89.98897),
np.float32(39.505894),
np.float32(0.0122514665),
np.float32(2218.0)]
```

```
In [3]: fire_folders = []
        look_back = 5  # 5 days sequence
        all_frames = []
        data_limit, loop_counter, loop_start = 30, 0, 0
        channel_descriptions = None
        base_path = "./data"
        target_shape_h, target_shape_w = 128, 128
        print('Loading...')
        for fire_folder in os.listdir(base_path):
            loop_counter += 1
            if loop_counter - loop_start > data_limit:
                break
            if loop_counter < loop_start:</pre>
                continue
            print('Progress: {0}/{1} ({2})'.format(loop_counter - loop_start, data_limit, l
            fire_folders.append(fire_folder)
            for image name in os.listdir(base path + f"/{fire folder}"):
                file_path = base_path + f"/{fire_folder}/{image_name}"
                with rasterio.open(file_path, 'r') as geotiff:
                    src = geotiff.read()
                    channel_descriptions = geotiff.descriptions
                    zoom factor = (1, target_shape_h / src.shape[1], target_shape_w / src.s
                    resized_src = zoom(src, zoom_factor, order=1)
                    resized_src = np.nan_to_num(resized_src, copy=True)
                    all_frames.append(resized_src)
        print(f'Loading done! Count = {len(all_frames)} | Shape = {all_frames[0].shape}')
        data frames = np.stack(all frames)
        print(data_frames.shape)
```

localhost:8888/lab 2/152

```
Loading...
       Progress: 1/30 (1)
       Progress: 2/30 (2)
       Progress: 3/30 (3)
       Progress: 4/30 (4)
       Progress: 5/30 (5)
       Progress: 6/30 (6)
       Progress: 7/30 (7)
       Progress: 8/30 (8)
       Progress: 9/30 (9)
       Progress: 10/30 (10)
       Progress: 11/30 (11)
       Progress: 12/30 (12)
       Progress: 13/30 (13)
       Progress: 14/30 (14)
       Progress: 15/30 (15)
       Progress: 16/30 (16)
       Progress: 17/30 (17)
       Progress: 18/30 (18)
       Progress: 19/30 (19)
       Progress: 20/30 (20)
       Progress: 21/30 (21)
       Progress: 22/30 (22)
       Progress: 23/30 (23)
       Progress: 24/30 (24)
       Progress: 25/30 (25)
       Progress: 26/30 (26)
       Progress: 27/30 (27)
       Progress: 28/30 (28)
       Progress: 29/30 (29)
       Progress: 30/30 (30)
       Loading done! Count = 726 | Shape = (23, 128, 128)
       (726, 23, 128, 128)
In [4]: np.min(data_frames), np.max(data_frames)
Out[4]: (np.float32(-9018.023), np.float32(15955.0))
In [5]: from sklearn.preprocessing import MinMaxScaler, minmax scale
        #data_frames = (data_frames - min_bound) / (max_bound - min_bound)
        for c in range(23):
            data_frames[:, c, :, :] = (data_frames[:, c, :, :] - min_values[c]) / (max_value)
In [6]: np.min(data_frames), np.max(data_frames)
Out[6]: (np.float32(0.0), np.float32(1.0))
In [7]: X = [1]
        Y = []
        for t in range(0, data_frames.shape[0] - look_back):
            x_seq = data_frames[t:t+look_back]
                                                             # shape: (5, 23, H, W)
            y_mask = data_frames[t + look_back, 22]
                                                              # fire mask from channel 22
```

localhost:8888/lab 3/152

```
X.append(x_seq)
                                 # binarize
             Y.append(y mask)
         X = np.stack(X) # shape: (273, 5, 23, 128, 128)
         Y = np.expand_dims(np.stack(Y), axis=1) # shape: (273, 1, 128, 128)
         print(X.shape)
         print(Y.shape)
        (721, 5, 23, 128, 128)
        (721, 1, 128, 128)
In [8]: split_index = int(X.shape[0] * 0.8)
         X_train = X[:split_index]
         X_test = X[split_index:]
         Y_train = Y[:split_index]
         Y_test = Y[split_index:]
         X_train.shape, Y_train.shape, X_test.shape, Y_test.shape
Out[8]: ((576, 5, 23, 128, 128),
          (576, 1, 128, 128),
          (145, 5, 23, 128, 128),
          (145, 1, 128, 128))
In [9]: X_train = torch.tensor(X_train).float()
         Y_train = torch.tensor(Y_train).float()
         X_test = torch.tensor(X_test).float()
         Y_test = torch.tensor(Y_test).float()
         X_train.shape, Y_train.shape, X_test.shape, Y_test.shape
Out[9]: (torch.Size([576, 5, 23, 128, 128]),
          torch.Size([576, 1, 128, 128]),
          torch.Size([145, 5, 23, 128, 128]),
          torch.Size([145, 1, 128, 128]))
In [10]: class WildfireDataset(Dataset):
             def __init__(self, X, Y):
                 self.X = X
                 self.Y = Y
             def __len__(self):
                 return len(self.X)
             def __getitem__(self, i):
                 return self.X[i], self.Y[i]
         train_dataset = WildfireDataset(X_train, Y_train)
         test_dataset = WildfireDataset(X_test, Y_test)
         len(train_dataset), len(test_dataset)
```

localhost:8888/lab 4/152

```
Out[10]: (576, 145)
In [11]: from torch.utils.data import DataLoader
         batch_size = 8
         train_loader = DataLoader(train_dataset, batch_size=batch_size, shuffle=True)
         test loader = DataLoader(test dataset, batch size=batch size, shuffle=False)
In [12]: if torch.cuda.is_available():
             device = torch.device("cuda")
         elif torch.mps.is available():
             device = torch.device("mps")
         else:
             device = torch.device("cpu")
         device
Out[12]: device(type='cuda')
In [13]: import convlstm
         class Net(nn.Module):
             def __init__(self):
                  super().__init__()
                 orig_size = (330, 257)
                  self.clstm = convlstm.ConvLSTM(
                     input_size=(128, 128),
                     input_dim=23,
                     hidden dim=[64],
                     kernel_size=(3, 3),
                     num_layers=1
                 # (8, 64, 128, 128)
                  self.head = nn.Sequential(
                     nn.Conv2d(64, 1, kernel_size=3, padding=1),
                     nn.Sigmoid()
                  # (8, 1, 128, 128)
             def forward(self, x):
                  batch size = x.size(0)
                  outputs, last_states = self.clstm(x)
                 x = outputs[0][:, -1, :, :, :]
                 x = self.head(x)
                  return x
         model = Net().to(device)
         model.load_state_dict(torch.load('model-weights.pth'))
         model.eval()
```

localhost:8888/lab 5/152

```
Out[13]: Net(
            (clstm): ConvLSTM(
              (cell list): ModuleList(
                (0): ConvLSTMCell(
                  (conv): Conv2d(87, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
              )
            )
            (head): Sequential(
              (0): Conv2d(64, 1, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
              (1): Sigmoid()
            )
          )
In [14]: tolerance = 0.01
         accuracy = 0
         precision = 0
         accuracy_counter = 0
         precision_counter = 0
         show_figures = True
         eps = 1e-7
         recall = 0
         f1 = 0
         iou = 0
         dice = 0
         for batch_index, batch in enumerate(test_loader):
             x_batch, y_batch = batch[0].to(device), batch[1].to(device)
             y pred = model(x batch)
             y_batch_np = y_batch.cpu().detach().numpy()
             y_pred_np = y_pred.cpu().detach().numpy()
             for i in range(y_batch_np.shape[0]): # Each image in a batch = batch_size
                  split val = 0.15
                  value_true = y_batch_np[i, 0]
                  value_true = np.piecewise(value_true, [value_true < split_val, value_true >
                  if show_figures:
                      plt.figure()
                      plt.title(f"Actual {precision_counter}")
                      plt.imshow(value_true)
                  split_val = 0.5
                  value_pred = y_pred_np[i, 0]
                  value_pred = np.piecewise(value_pred, [value_pred < split_val, value_pred >
                  if show_figures:
                      plt.figure()
                      plt.title(f"Predicted {precision_counter}")
                      plt.imshow(value_pred)
                  correct = np.abs(value_pred - value_true) <= tolerance</pre>
                  accuracy += correct.sum() / correct.size
                  accuracy_counter += 1
                  # TP: True Positives
```

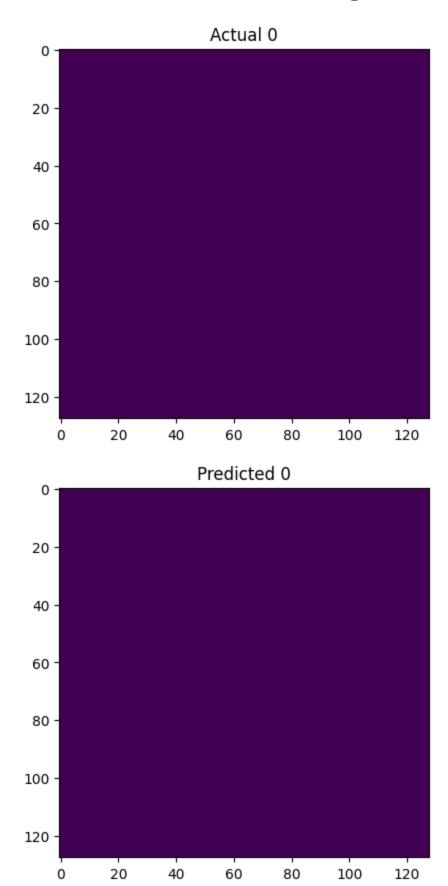
localhost:8888/lab 6/152

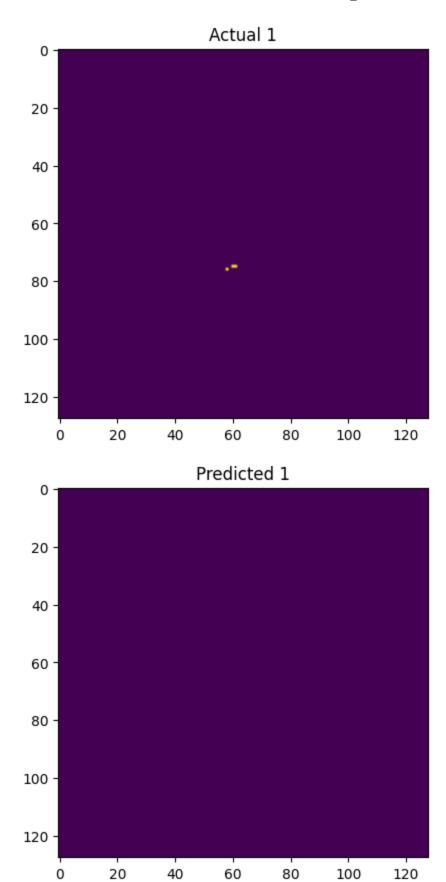
```
# FP: False Positives
        TP = np.logical_and(value_pred == 1, value_true == 1).sum()
        FP = np.logical and(value pred == 1, value true == 0).sum()
        FN = np.logical_and(value_pred == 0, value_true == 1).sum()
       TN = np.logical_and(value_pred == 0, value_true == 0).sum()
        precision += TP / (TP + FP + 1e-7) # Add small epsilon to avoid division b
        recall += TP / (TP + FN + eps)
        iou += TP / (TP + FP + FN + eps)
        dice += 2 * TP / (2 * TP + FP + FN + eps)
        precision_counter += 1
avg_accuracy = accuracy / accuracy_counter
avg precision = precision / precision counter
avg_recall = recall / precision_counter
avg_iou = iou / precision_counter
avg_dice = dice / precision_counter
f1 = 2 * avg_precision * avg_recall / (avg_precision + avg_recall + eps)
print(f"Precision: {avg_precision}; Recall: {avg_recall}; IoU: {avg_iou}, Dice: {av
```

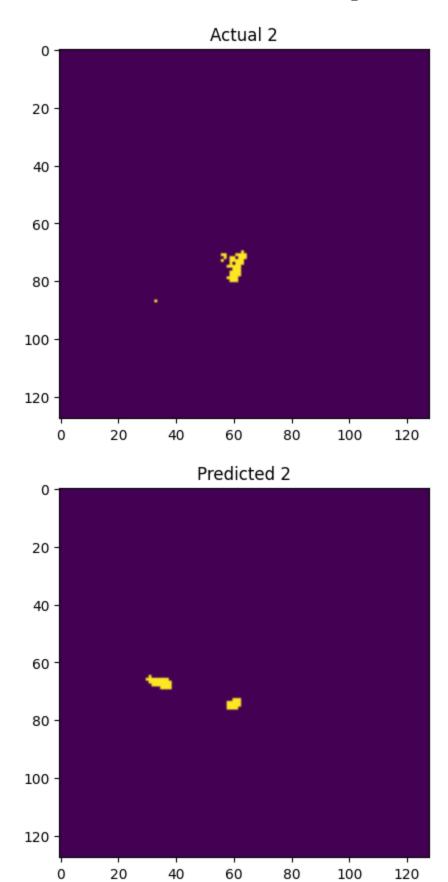
C:\Users\neelr\AppData\Local\Temp\ipykernel\_22156\1429483292.py:25: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot interface
(`matplotlib.pyplot.figure`) are retained until explicitly closed and may consume to
o much memory. (To control this warning, see the rcParam `figure.max\_open\_warning`).
Consider using `matplotlib.pyplot.close()`.
 plt.figure()

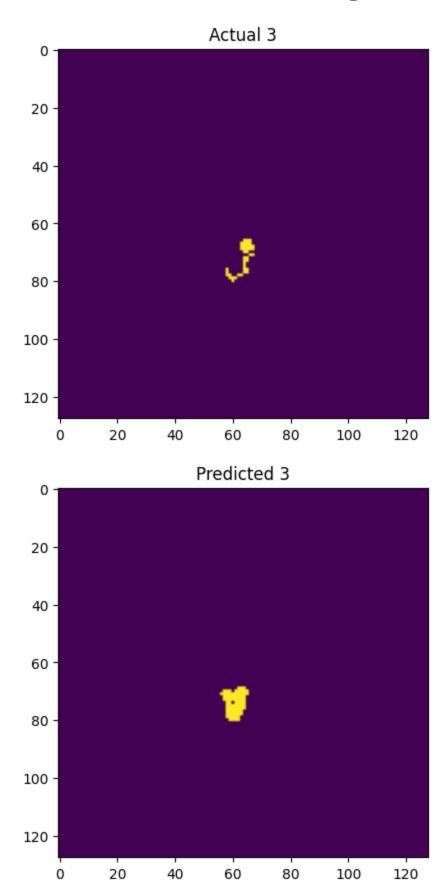
Precision: 0.16340744335717702; Recall: 0.2953673548888081; IoU: 0.1233657159752968 4, Dice: 0.1875543678133153, F1: 0.21040917669245868

localhost:8888/lab 7/152

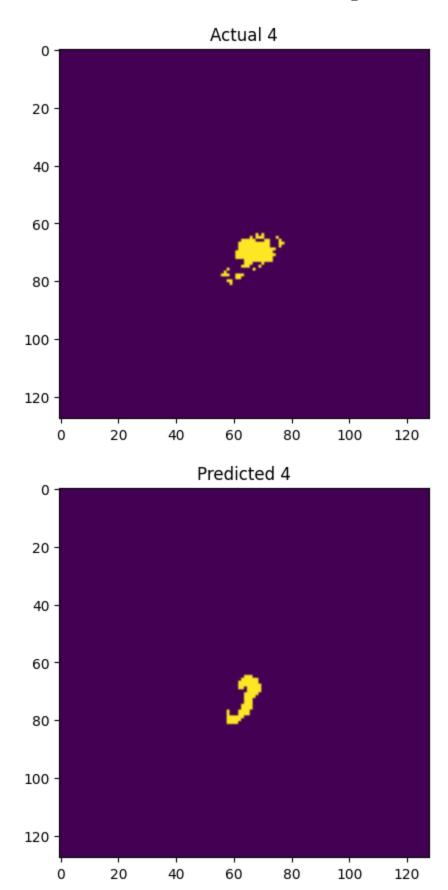


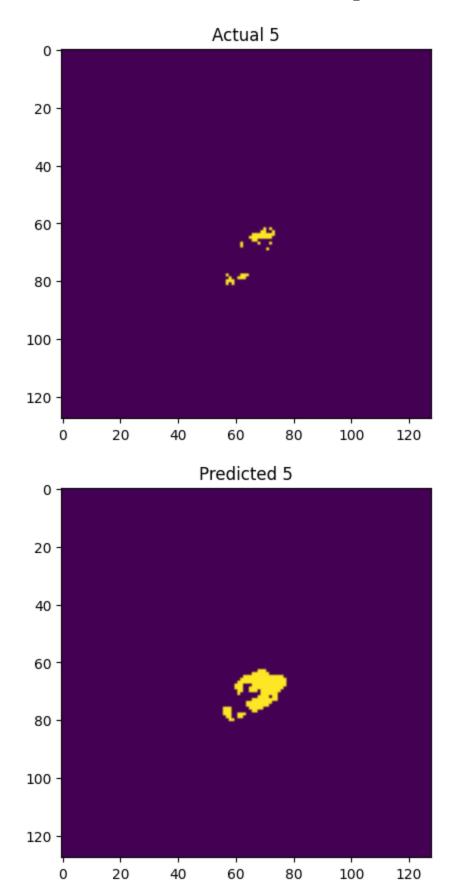


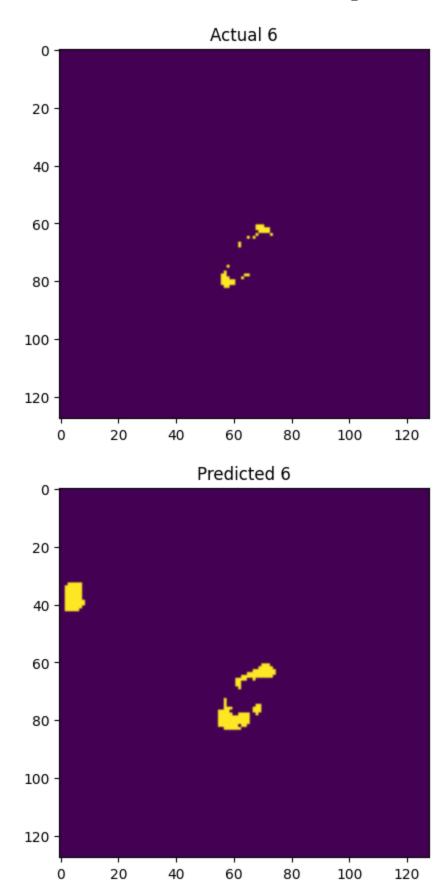


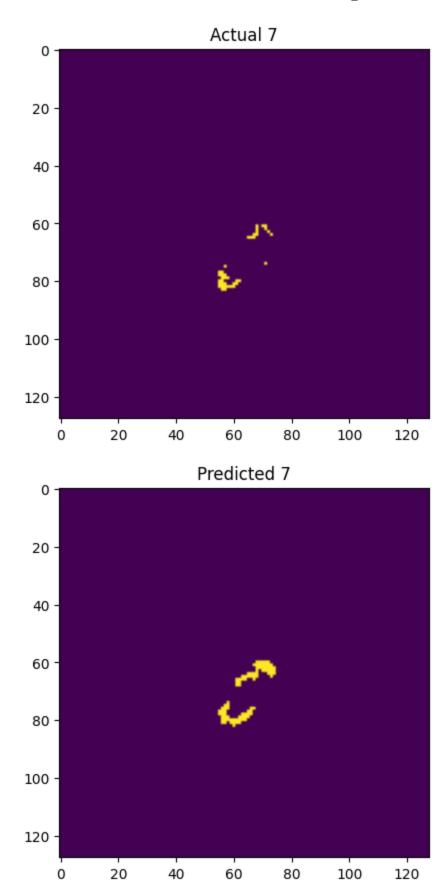


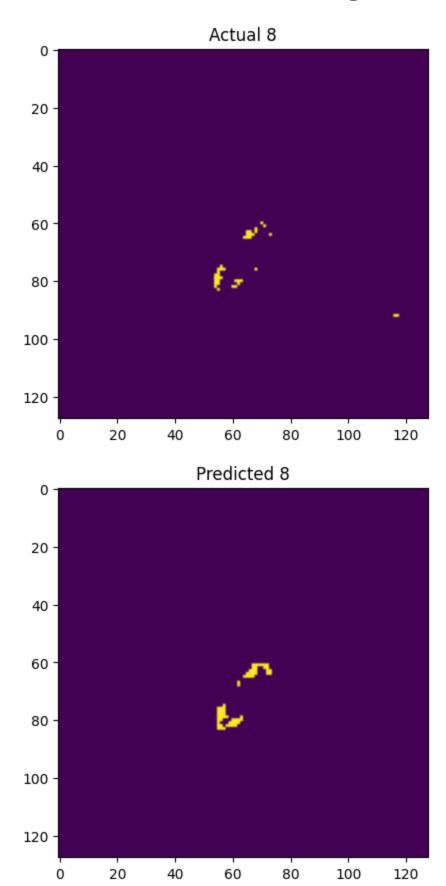
localhost:8888/lab 11/152

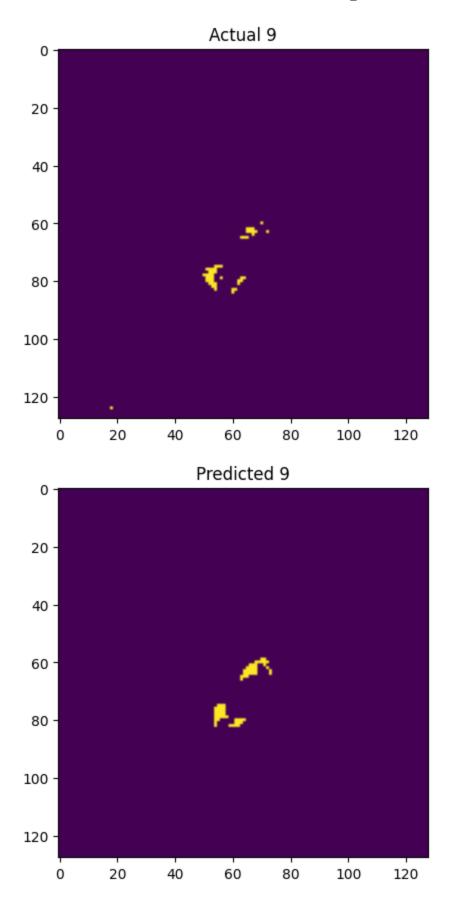


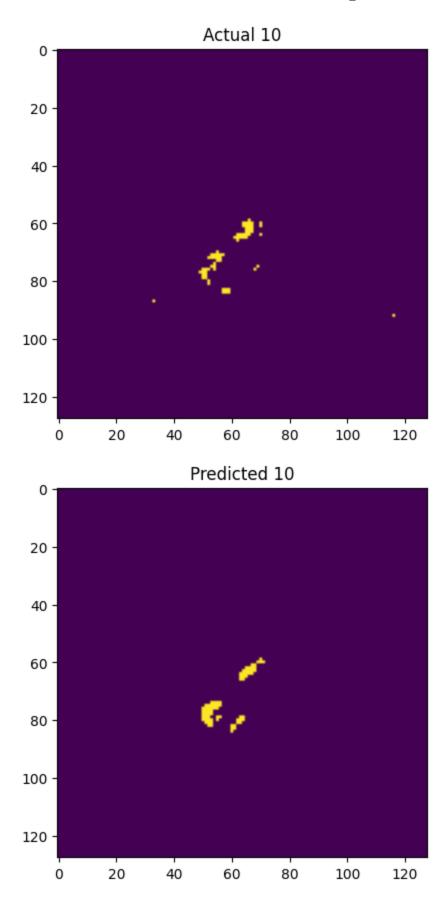




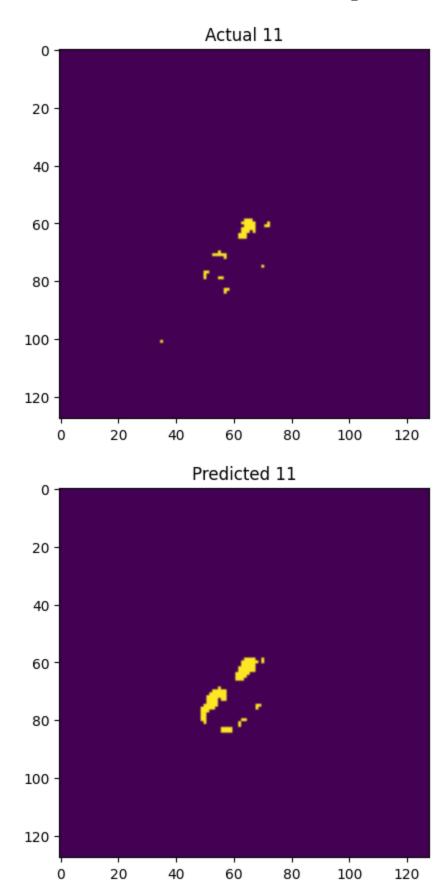


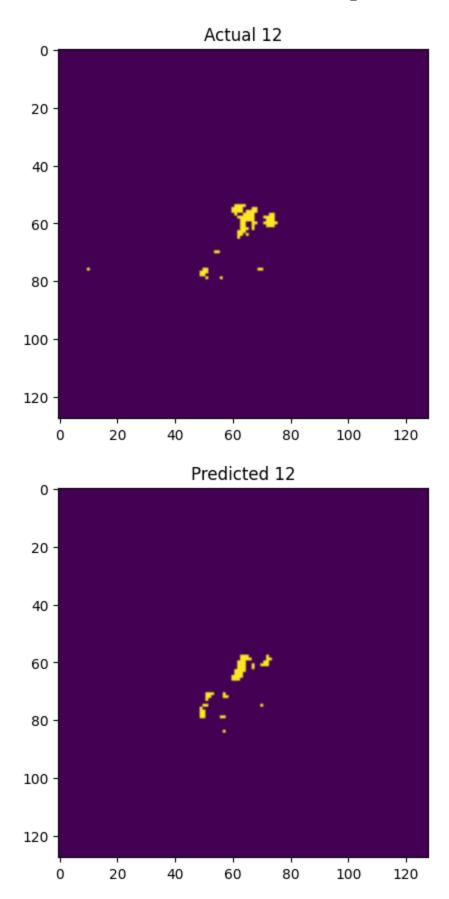




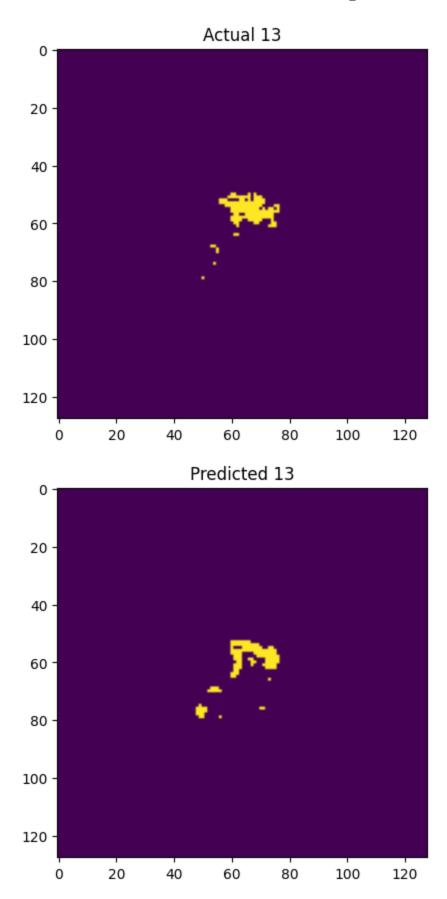


localhost:8888/lab 18/152

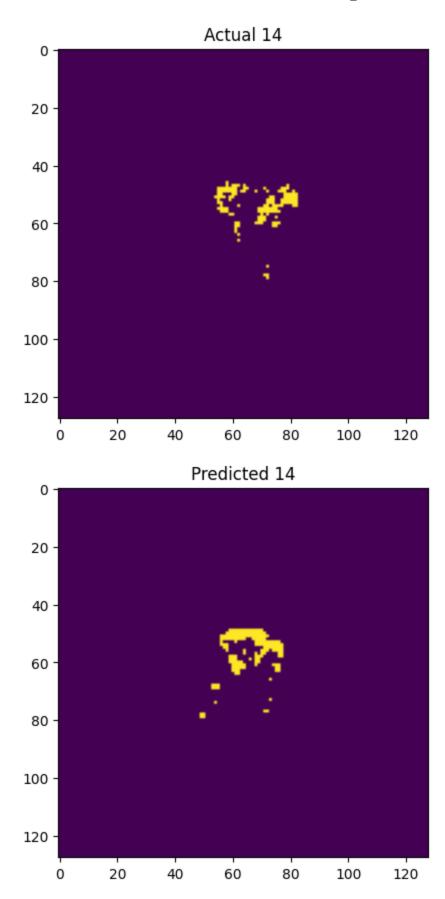




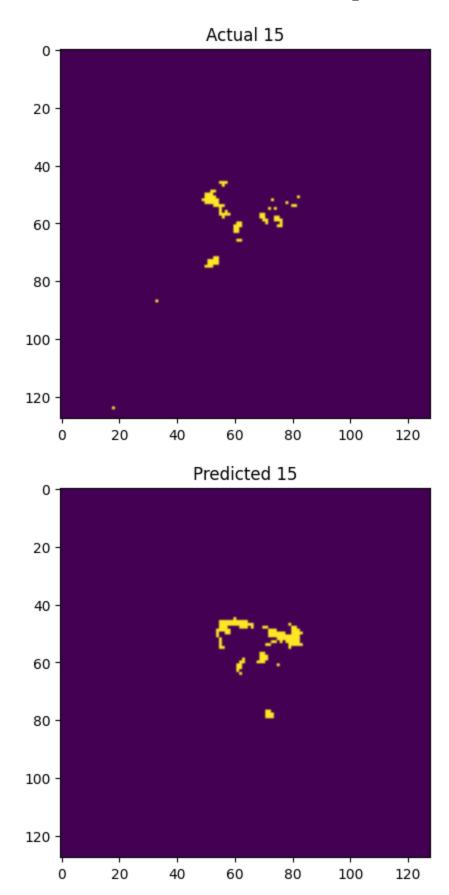
localhost:8888/lab 20/152



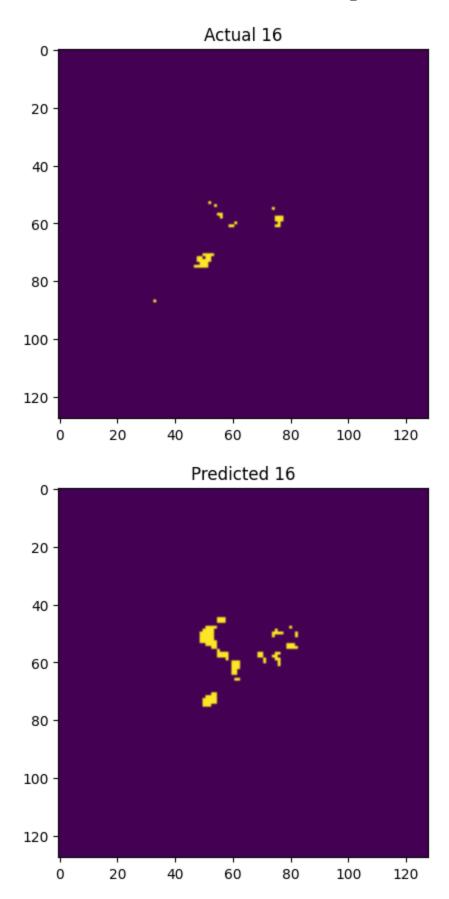
localhost:8888/lab 21/152



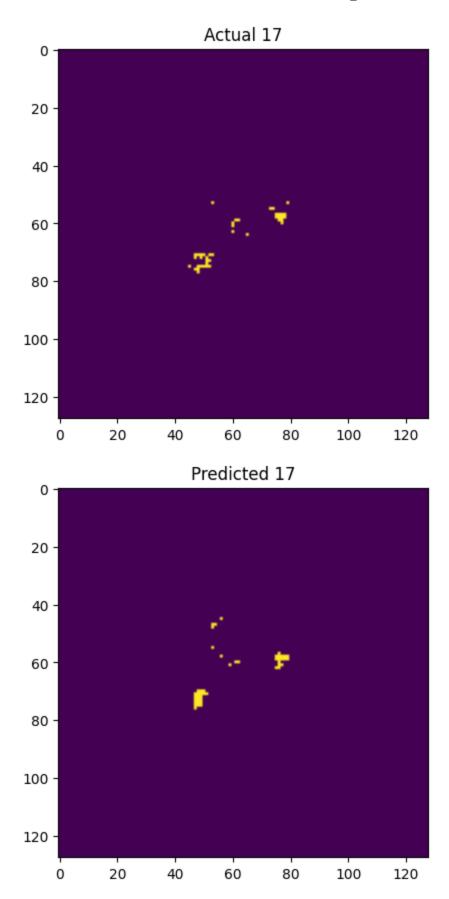
localhost:8888/lab 22/152



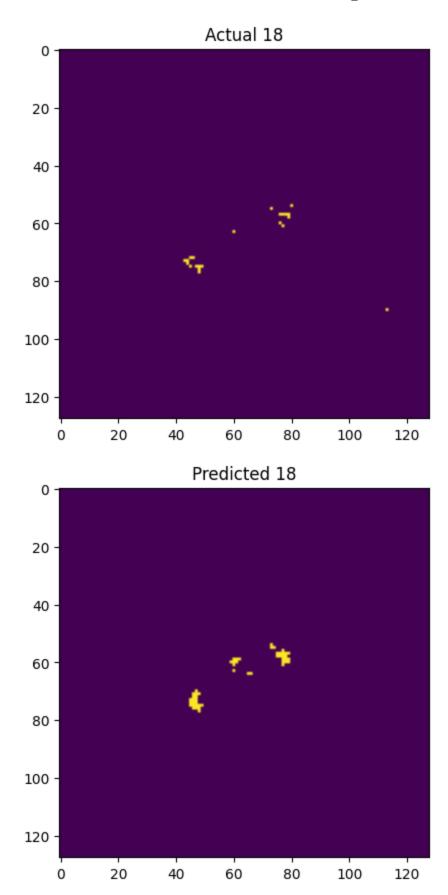
localhost:8888/lab 23/152



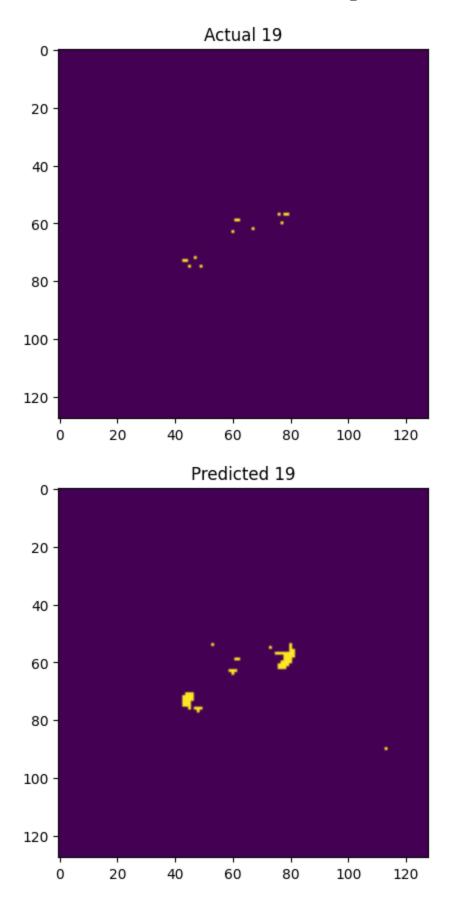
localhost:8888/lab 24/152



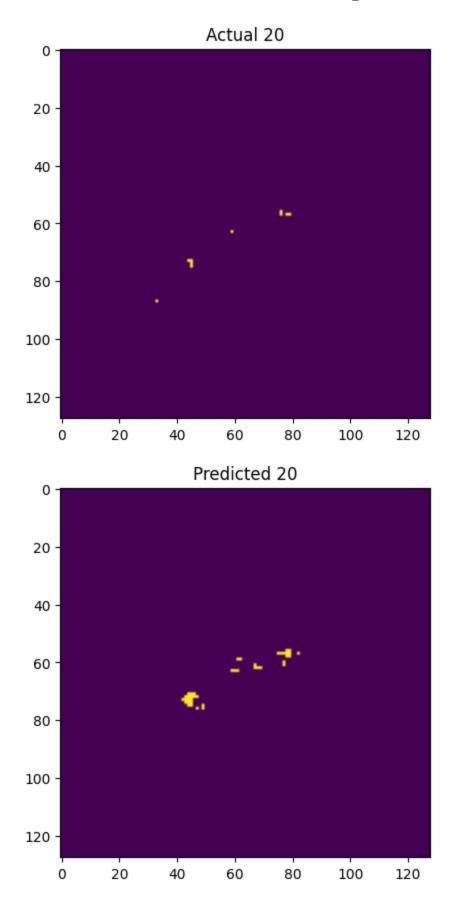
localhost:8888/lab 25/152



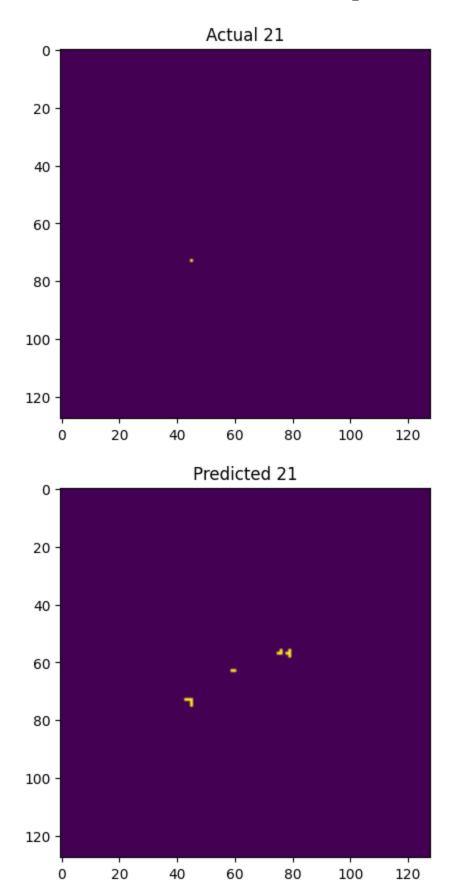
localhost:8888/lab 26/152



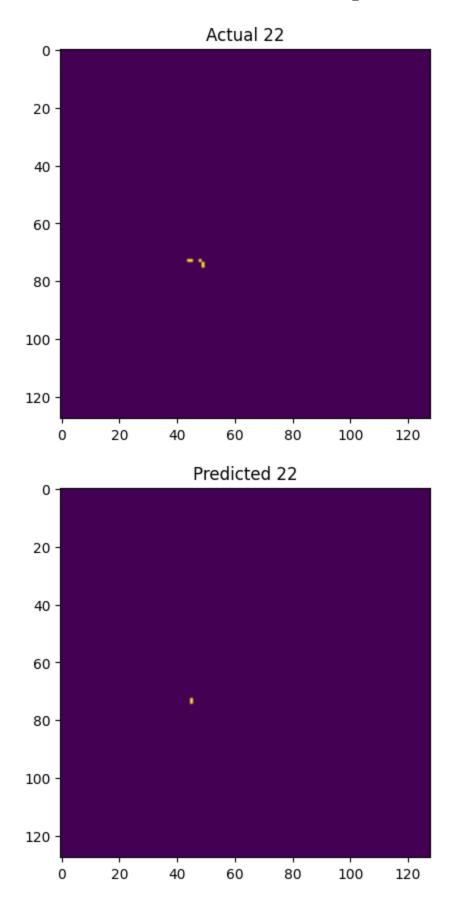
localhost:8888/lab 27/152



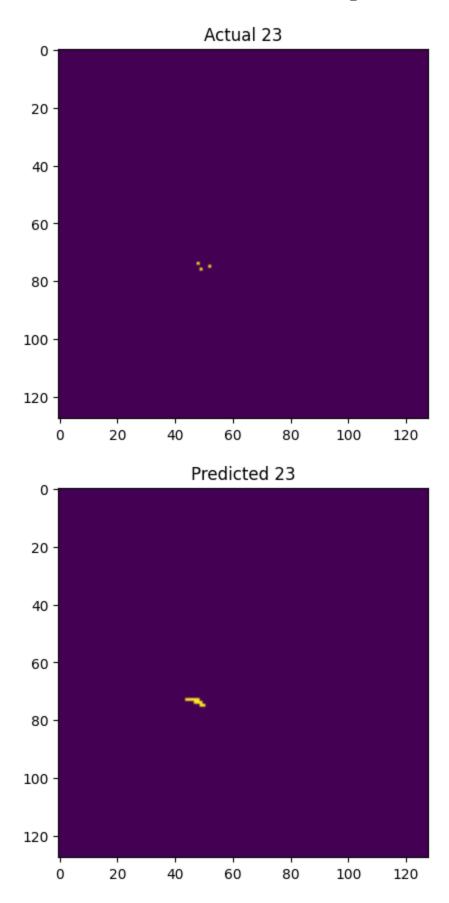
localhost:8888/lab 28/152



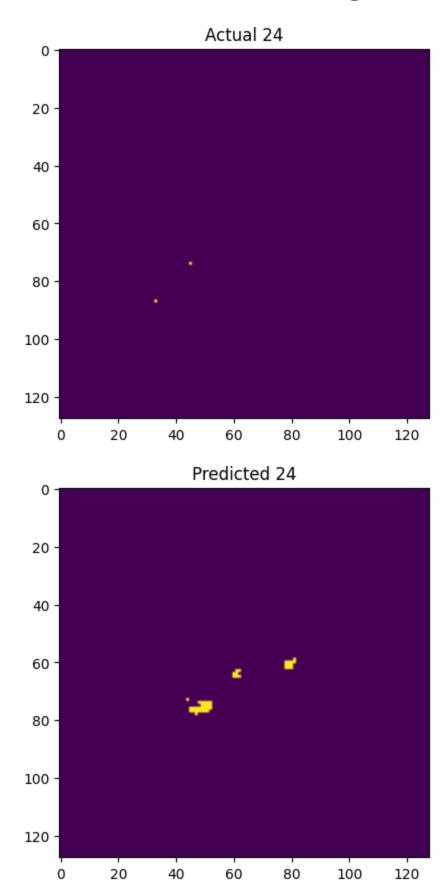
localhost:8888/lab 29/152



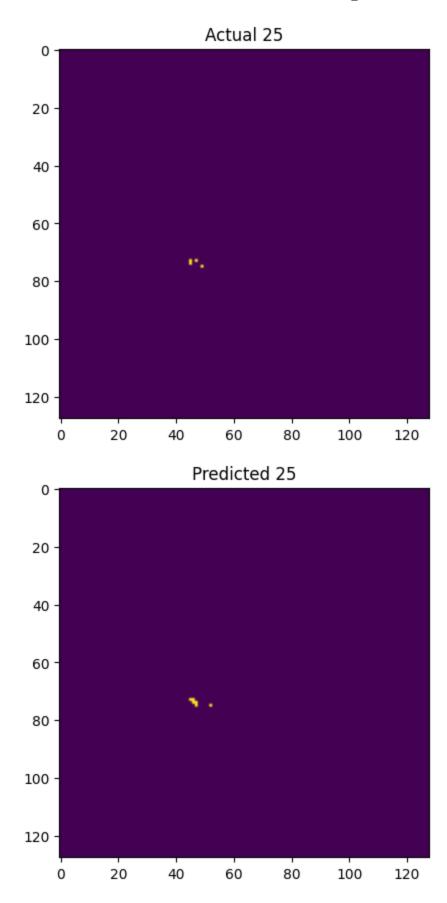
localhost:8888/lab 30/152



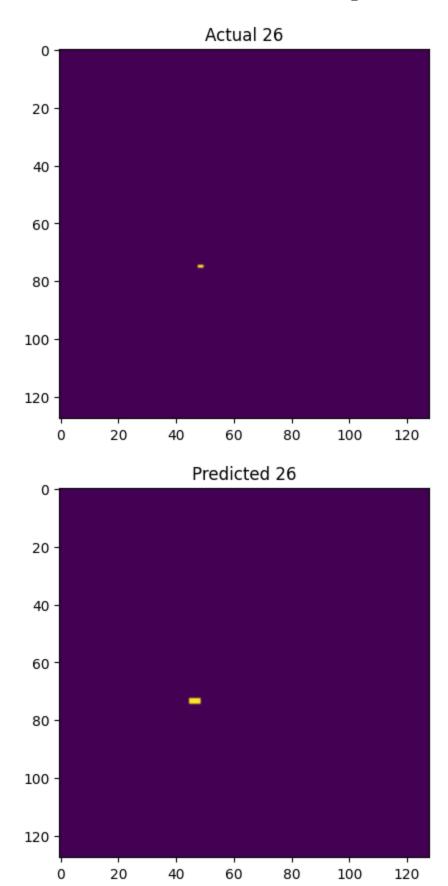
localhost:8888/lab 31/152



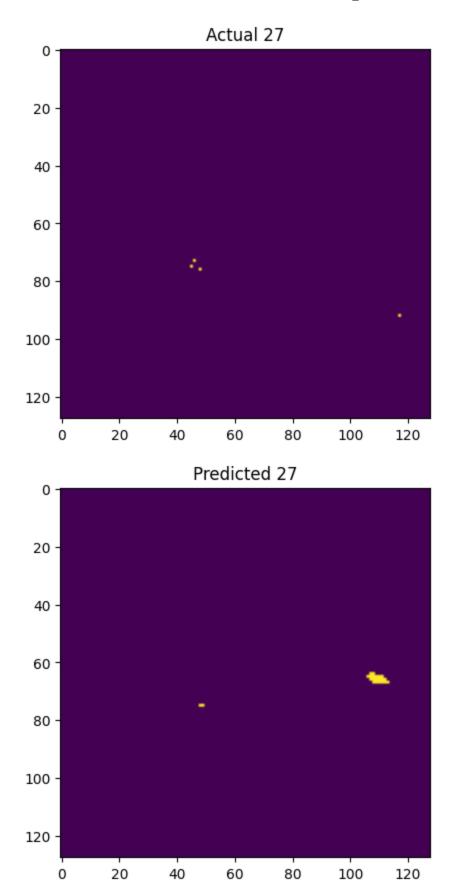
localhost:8888/lab 32/152



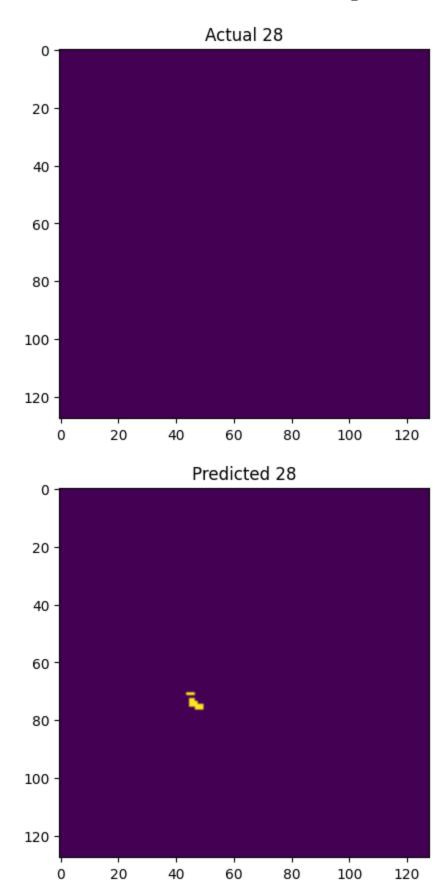
localhost:8888/lab 33/152



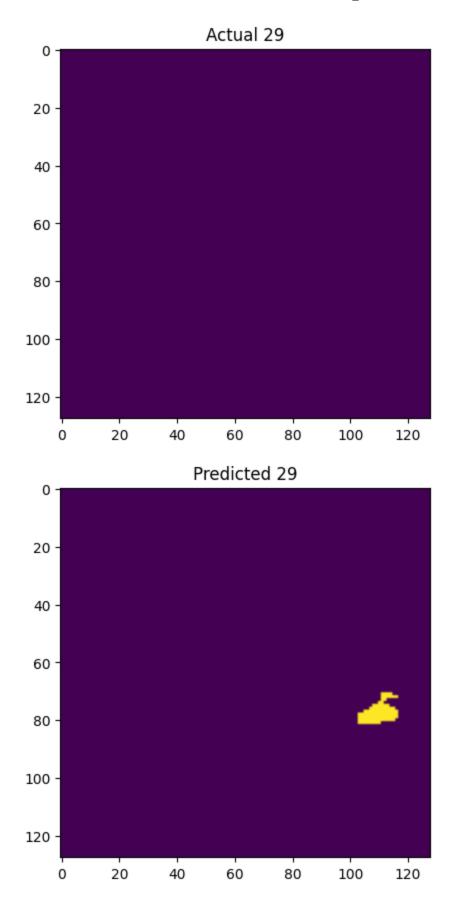
localhost:8888/lab 34/152



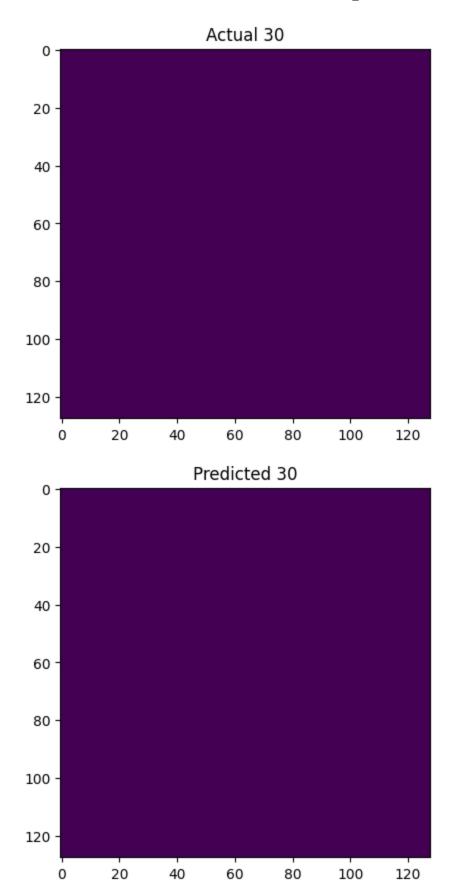
localhost:8888/lab 35/152



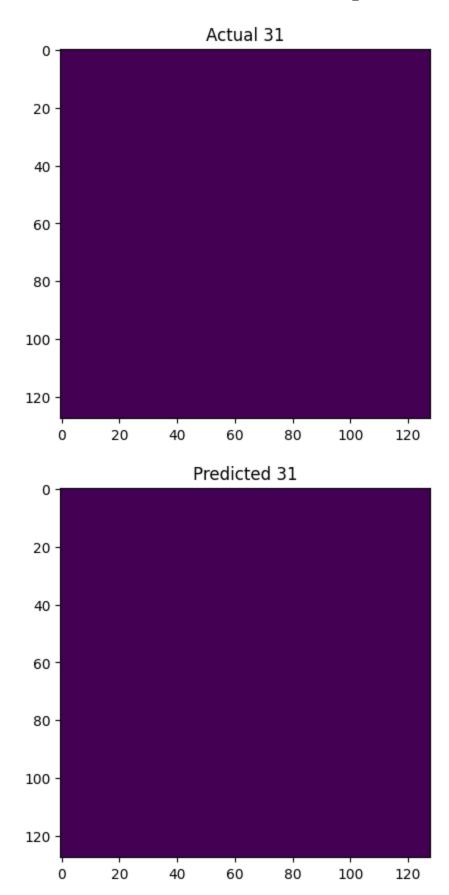
localhost:8888/lab 36/152



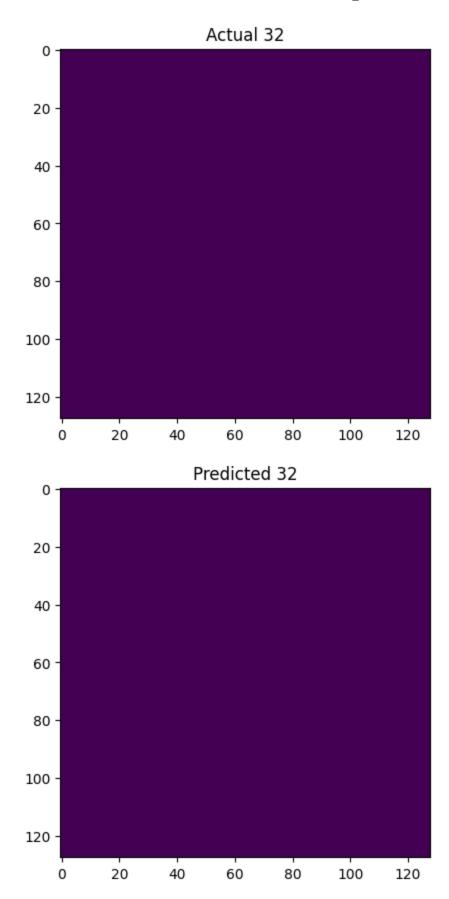
localhost:8888/lab 37/152



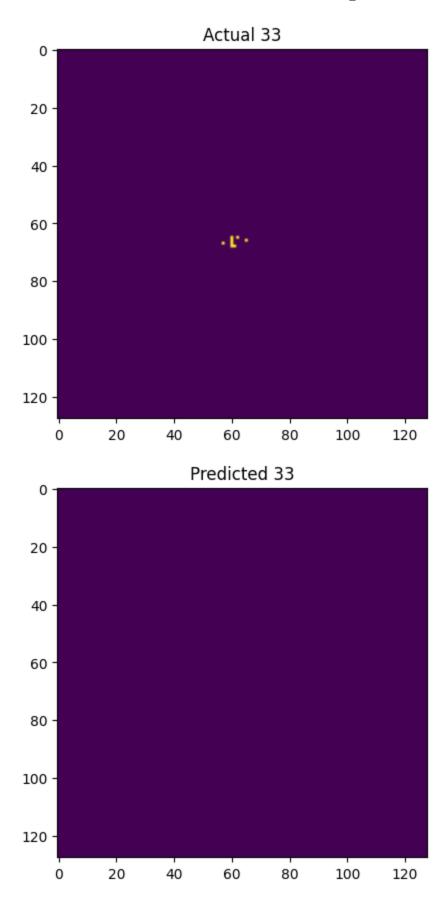
localhost:8888/lab 38/152



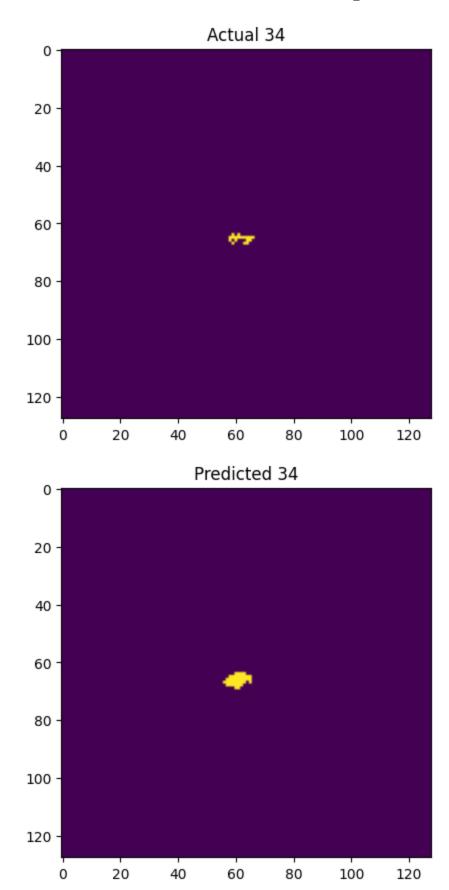
localhost:8888/lab 39/152



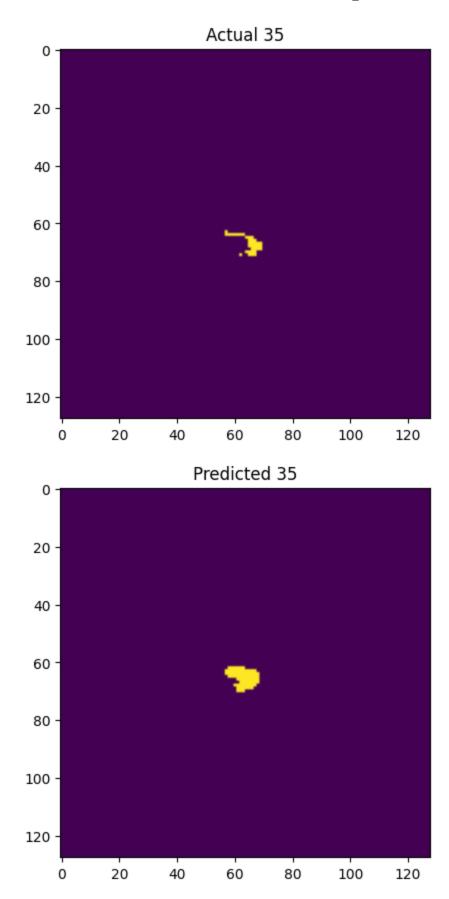
localhost:8888/lab 40/152



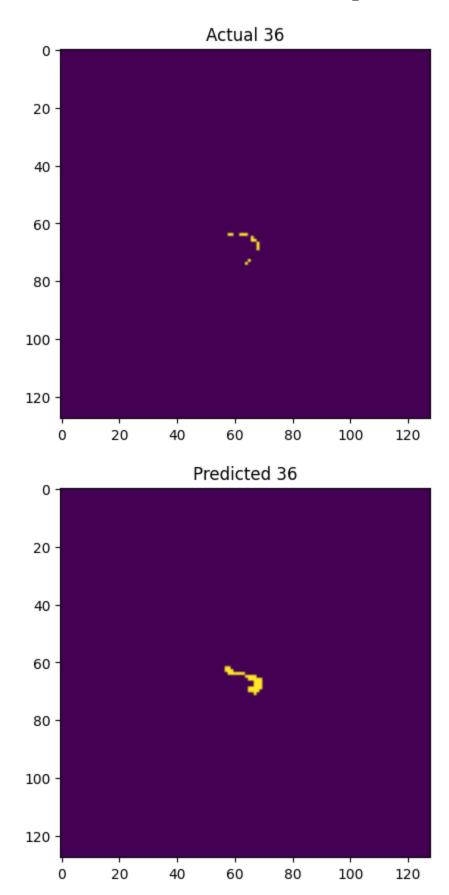
localhost:8888/lab 41/152



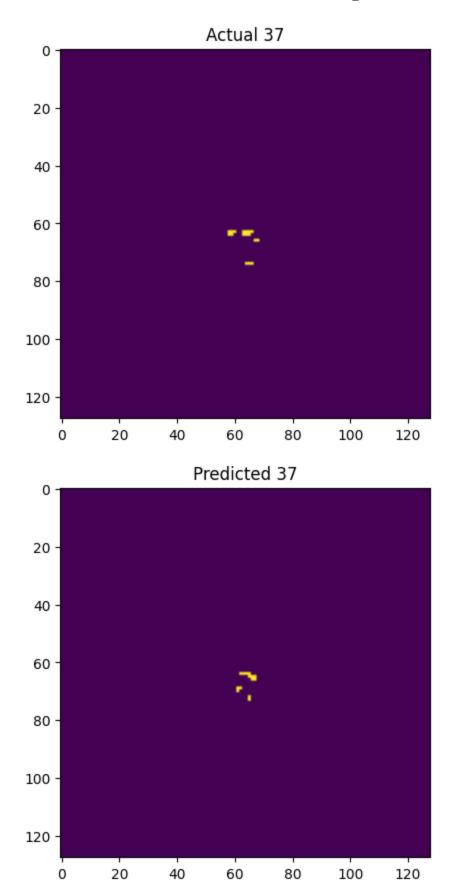
localhost:8888/lab 42/152



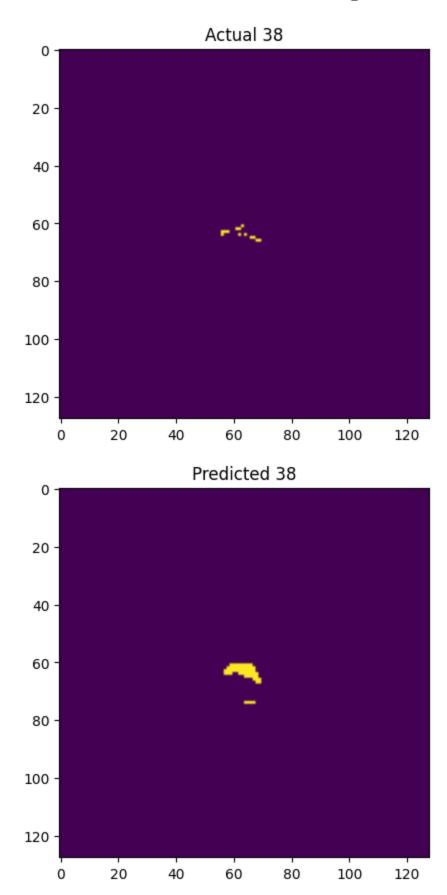
localhost:8888/lab 43/152



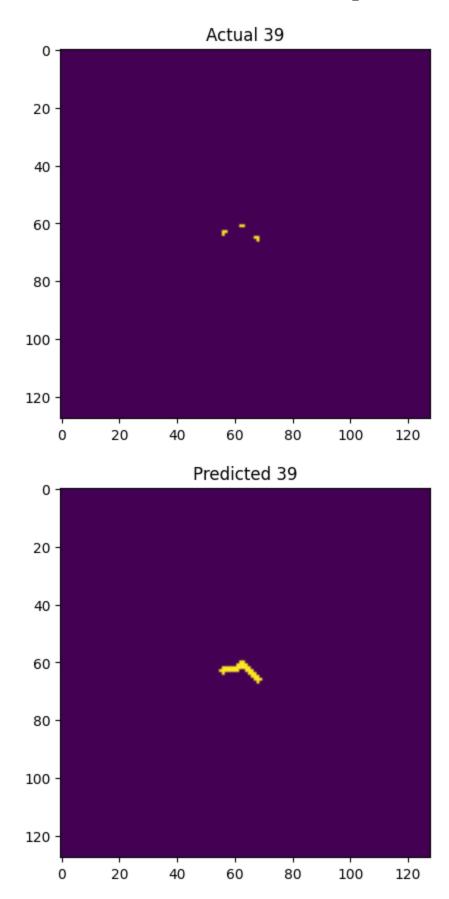
localhost:8888/lab 44/152



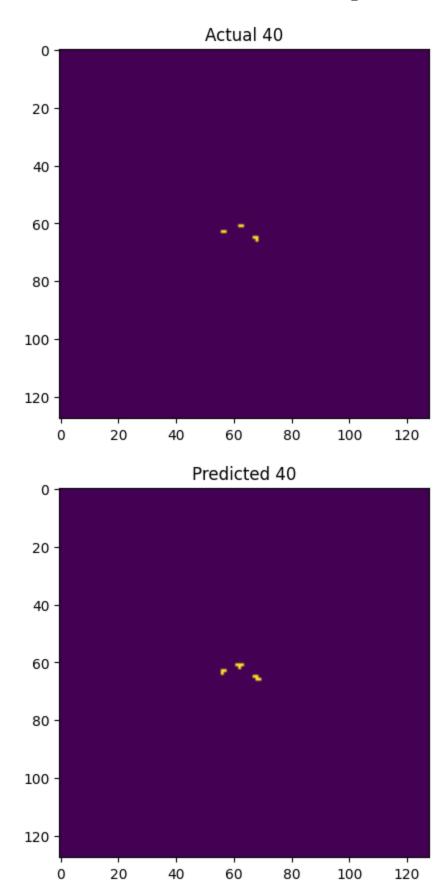
localhost:8888/lab 45/152

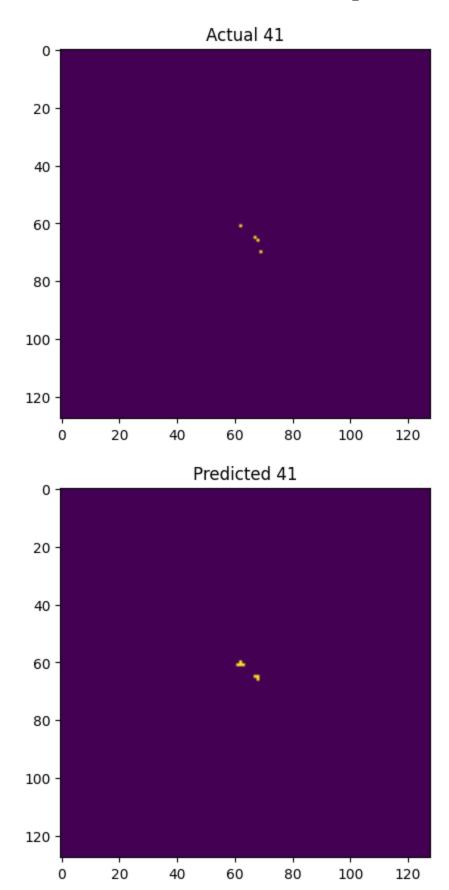


localhost:8888/lab 46/152

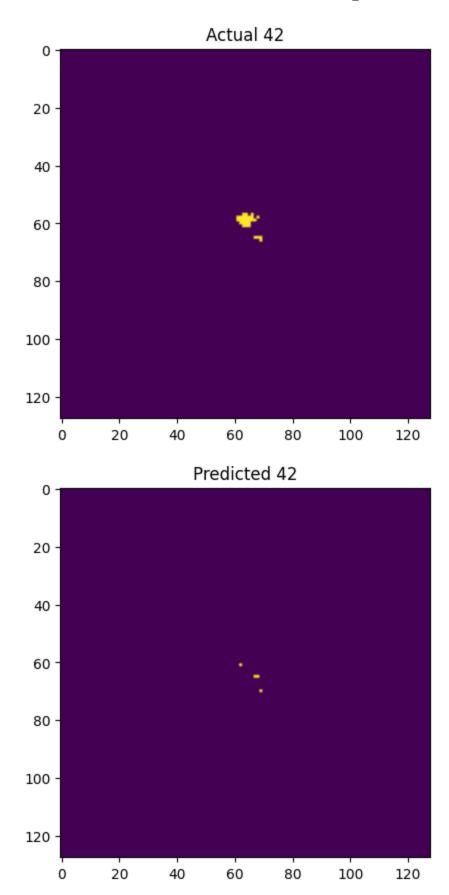


localhost:8888/lab 47/152

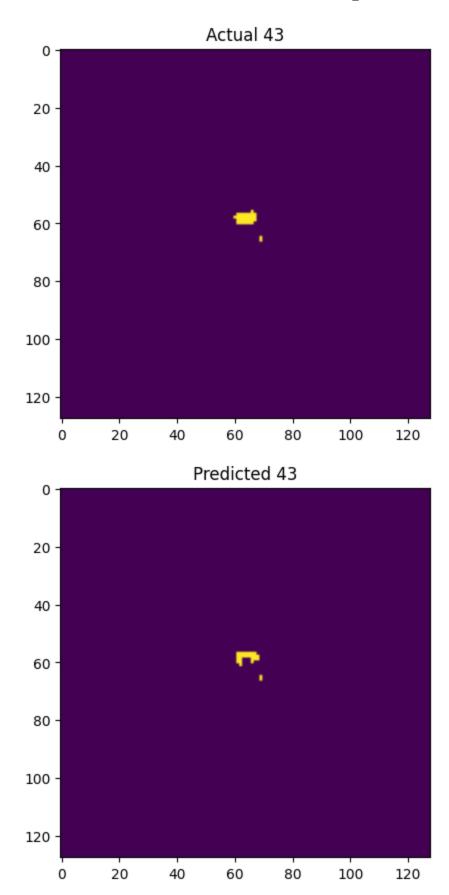




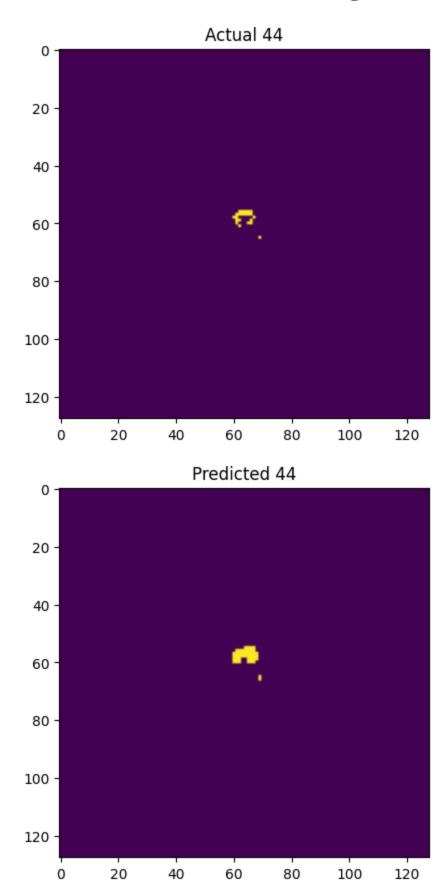
localhost:8888/lab 49/152



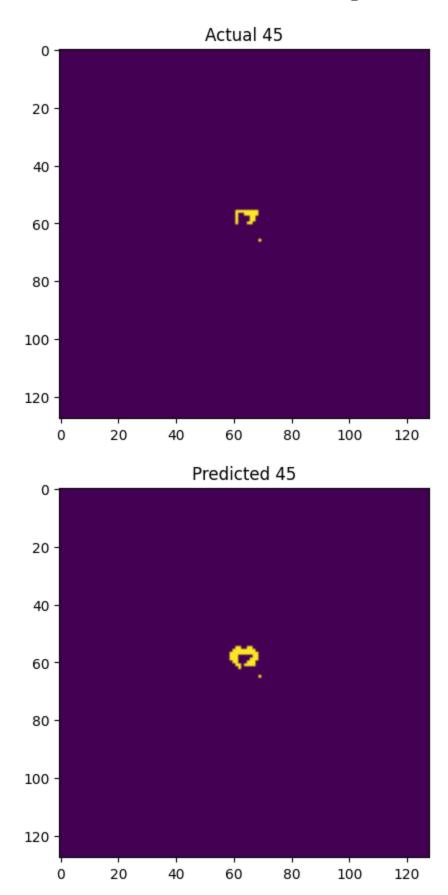
localhost:8888/lab 50/152



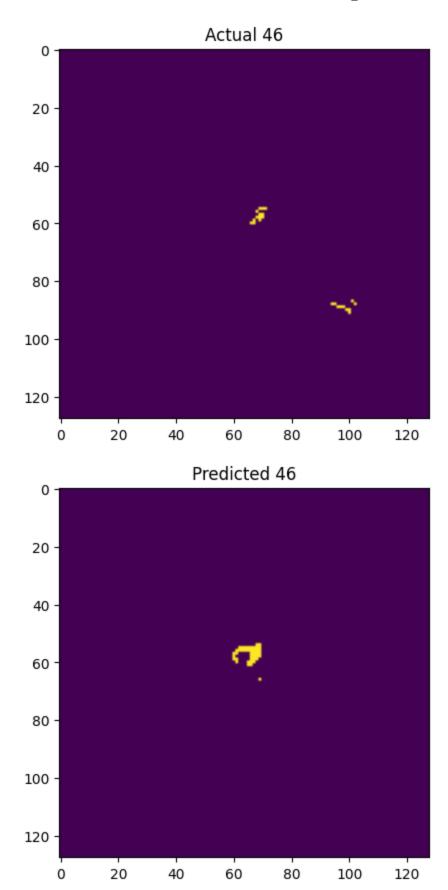
localhost:8888/lab 51/152



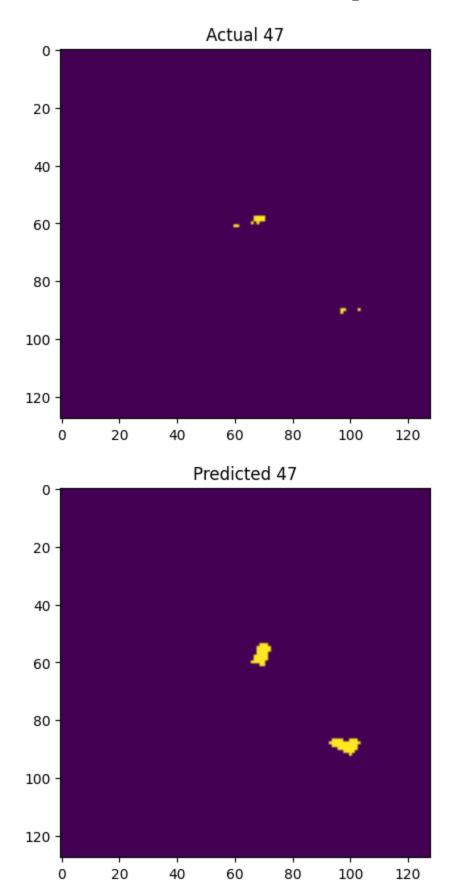
localhost:8888/lab 52/152



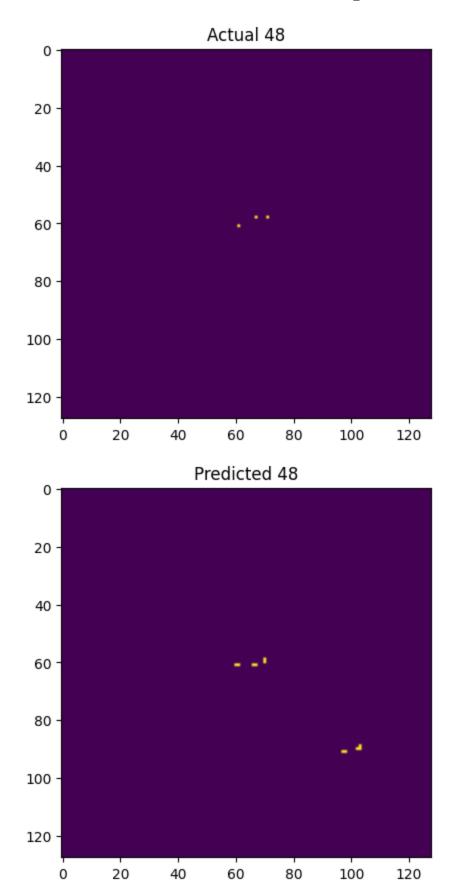
localhost:8888/lab 53/152



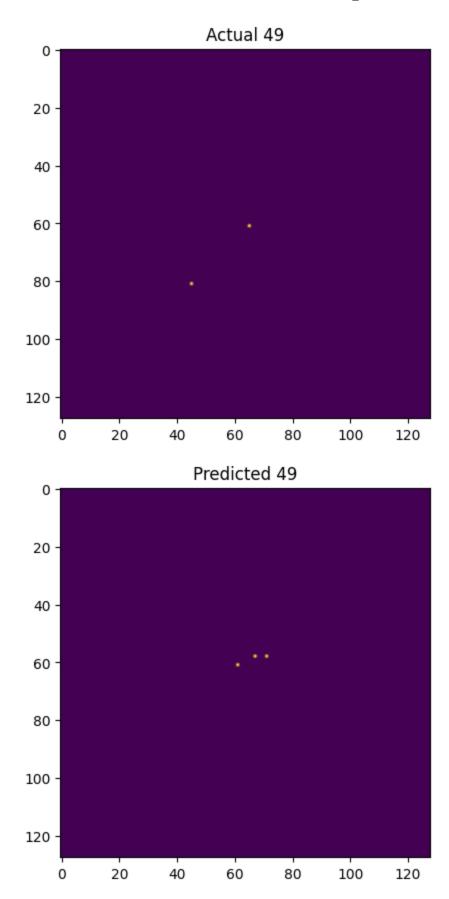
localhost:8888/lab 54/152



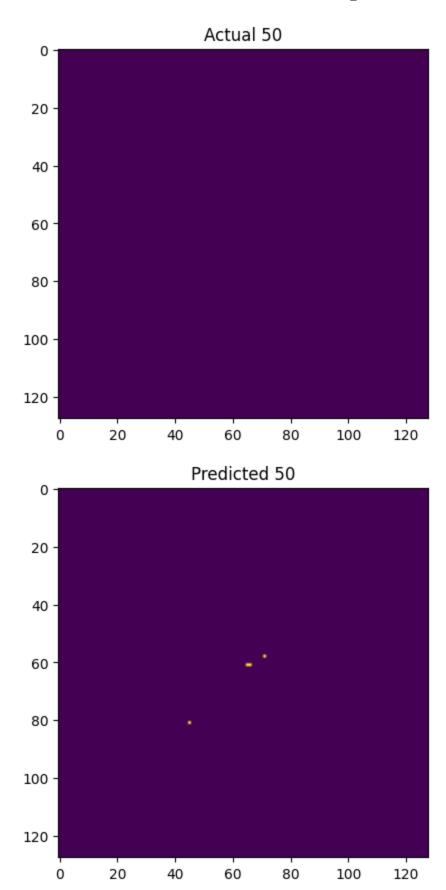
localhost:8888/lab 55/152



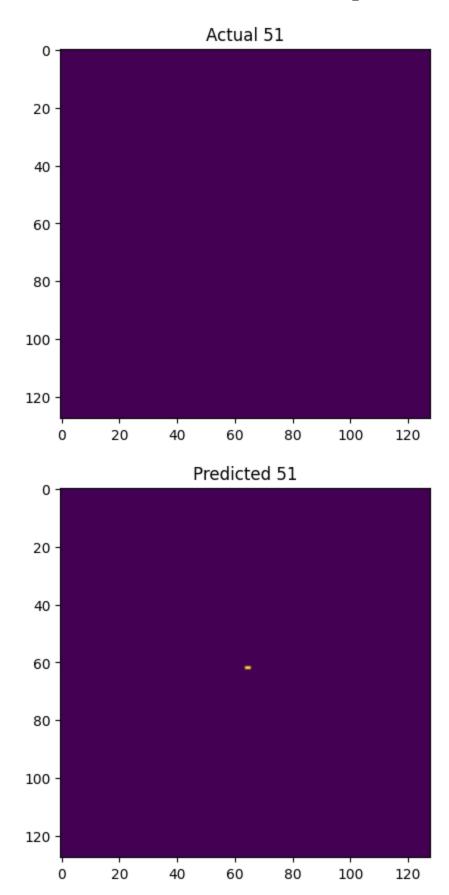
localhost:8888/lab 56/152



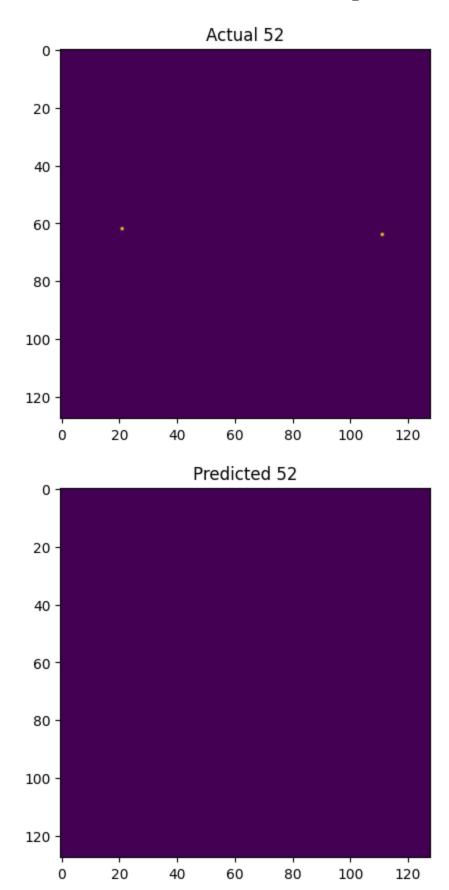
localhost:8888/lab 57/152



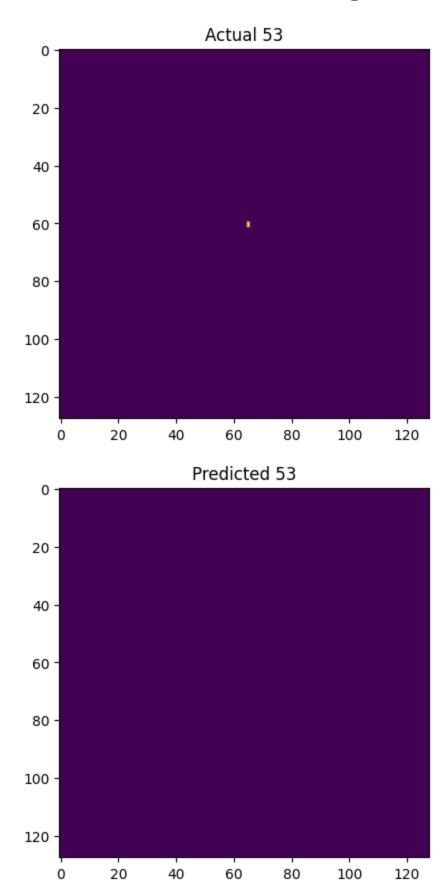
localhost:8888/lab 58/152



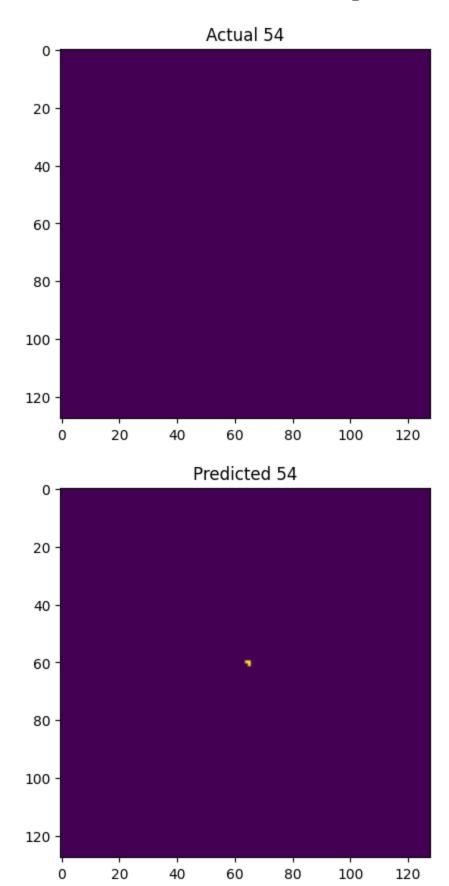
localhost:8888/lab 59/152



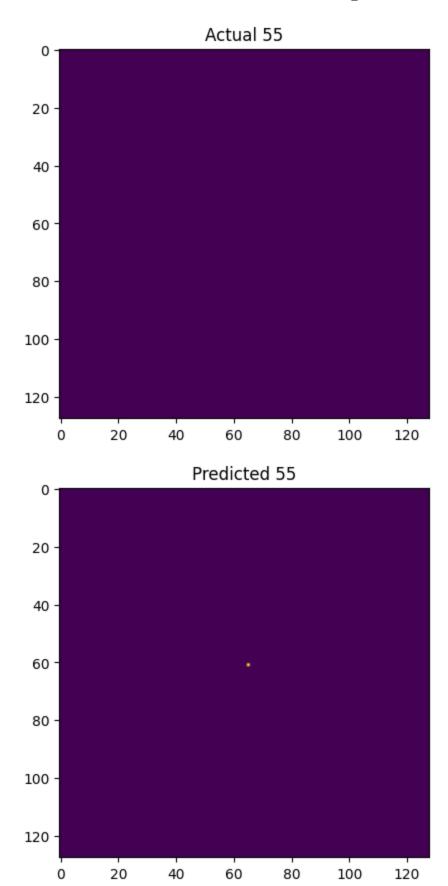
localhost:8888/lab 60/152



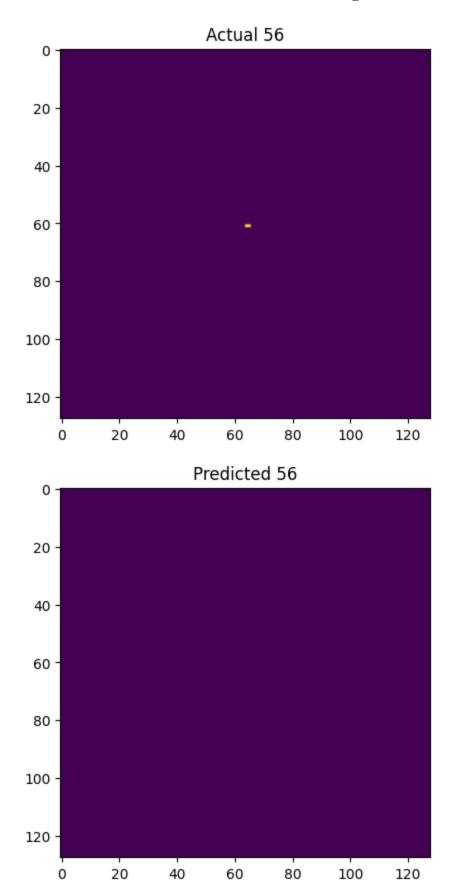
localhost:8888/lab 61/152



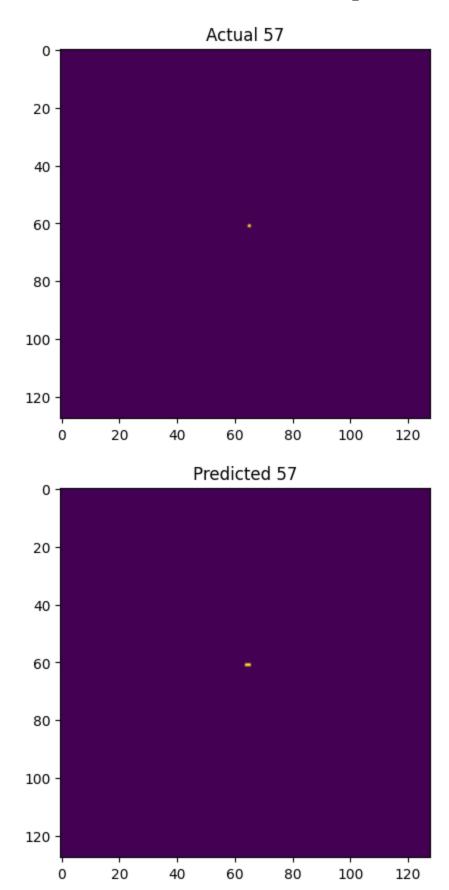
localhost:8888/lab 62/152



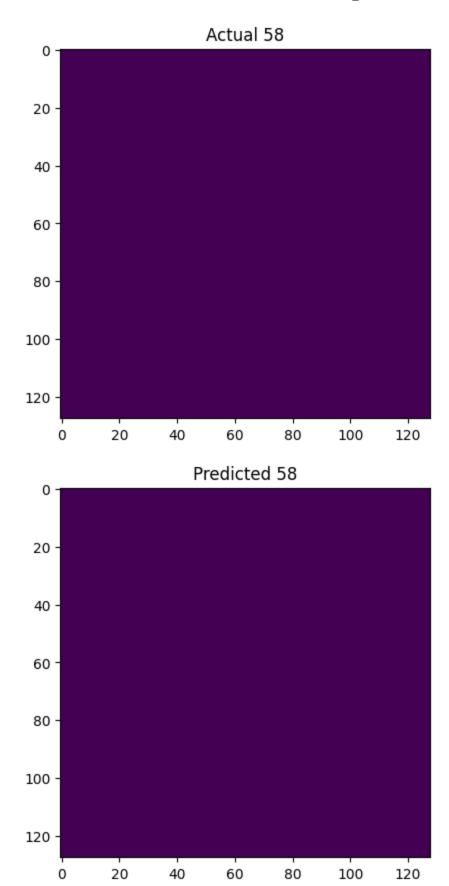
localhost:8888/lab 63/152



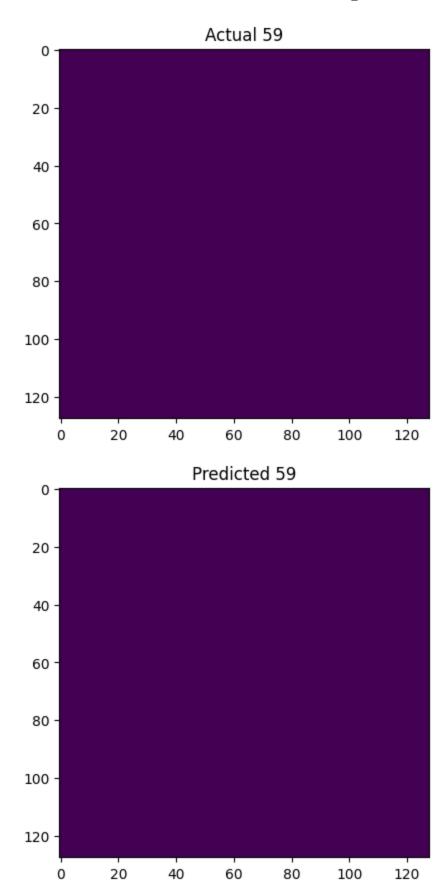
localhost:8888/lab 64/152



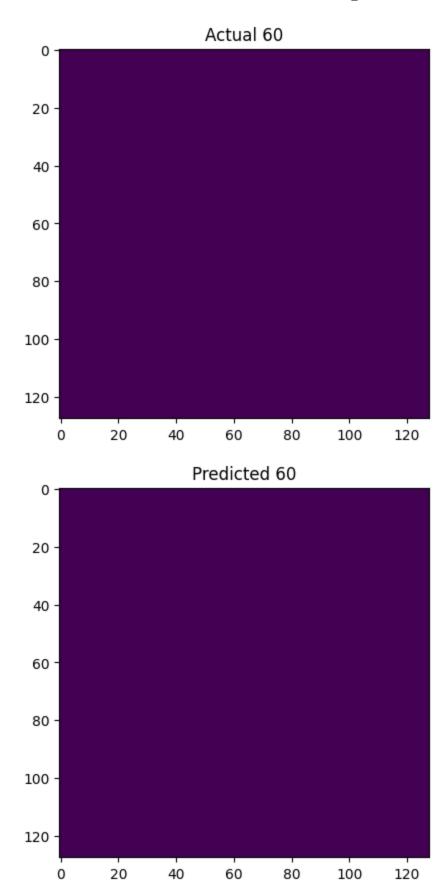
localhost:8888/lab 65/152



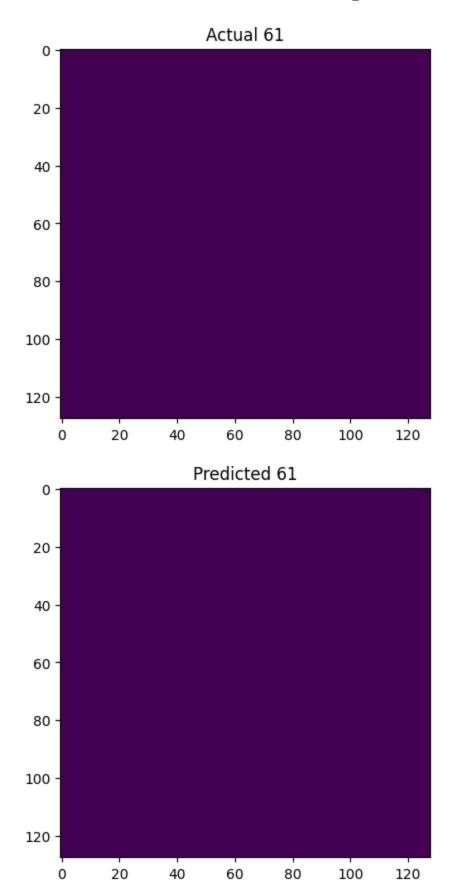
localhost:8888/lab 66/152



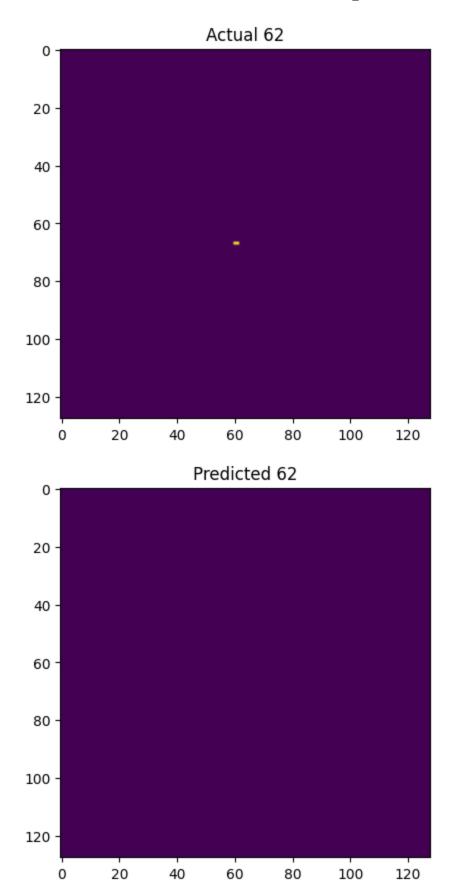
localhost:8888/lab 67/152



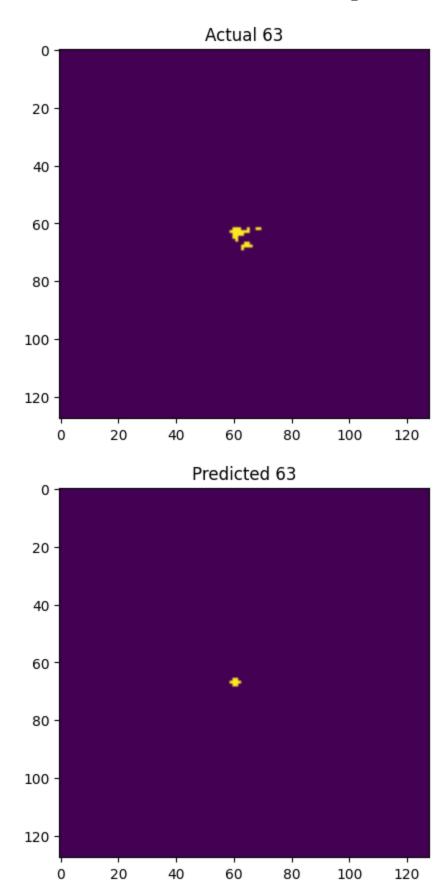
localhost:8888/lab 68/152



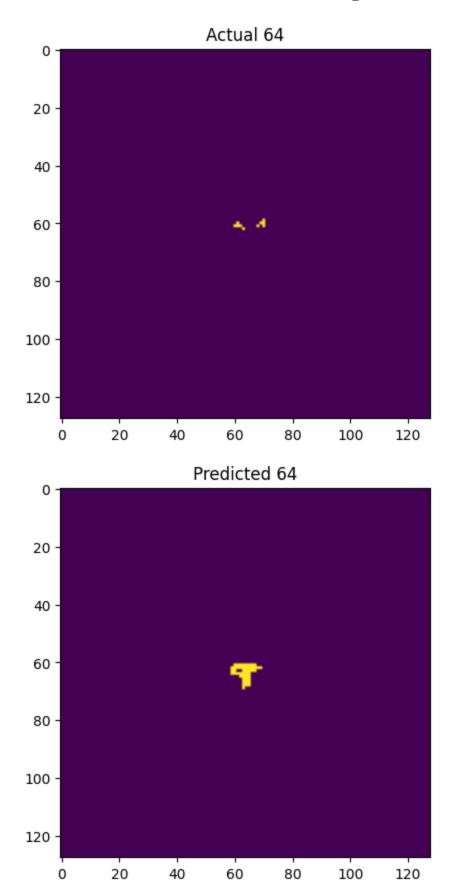
localhost:8888/lab 69/152



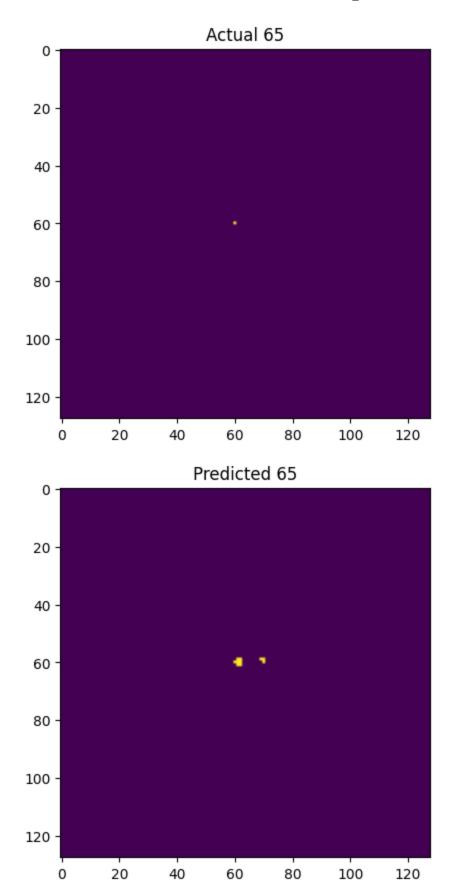
localhost:8888/lab 70/152



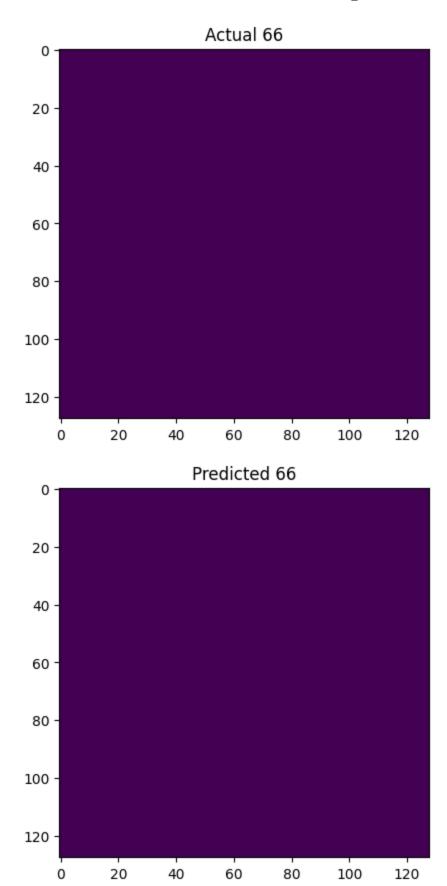
localhost:8888/lab 71/152



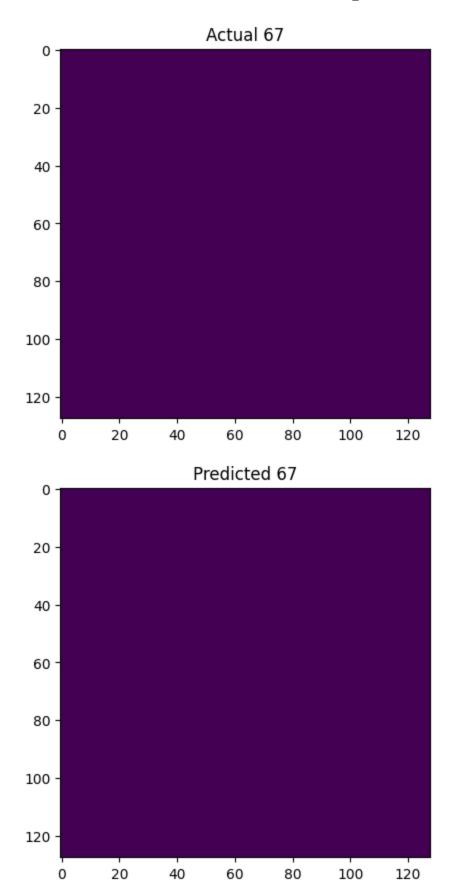
localhost:8888/lab 72/152



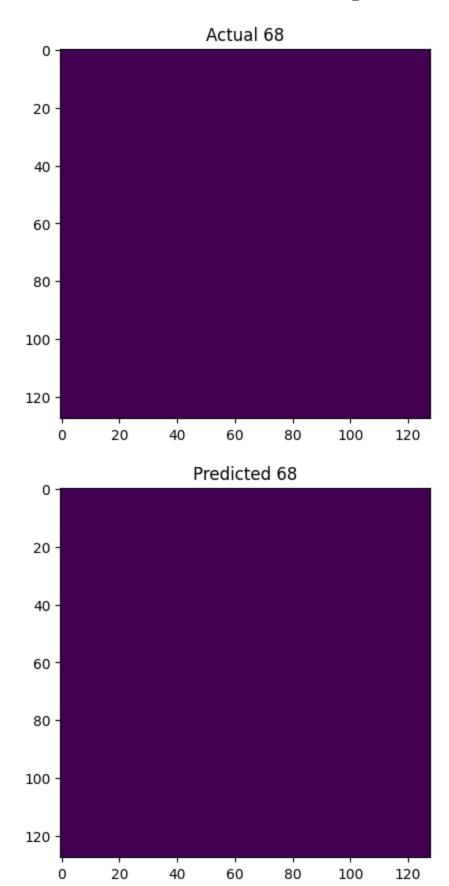
localhost:8888/lab 73/152



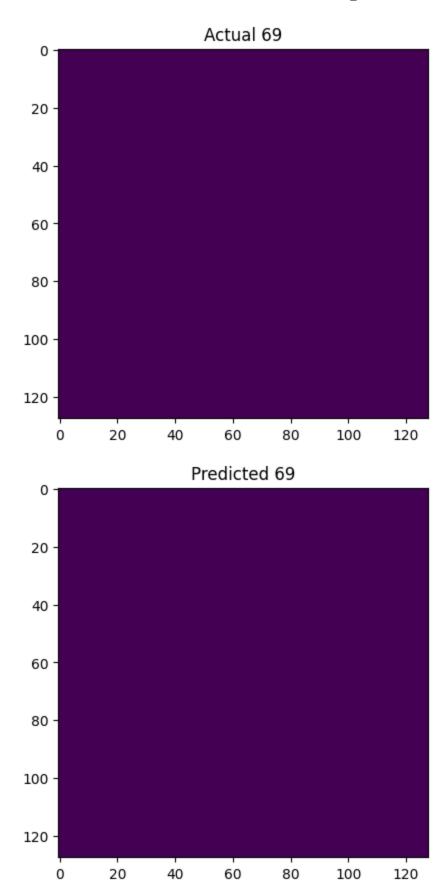
localhost:8888/lab 74/152



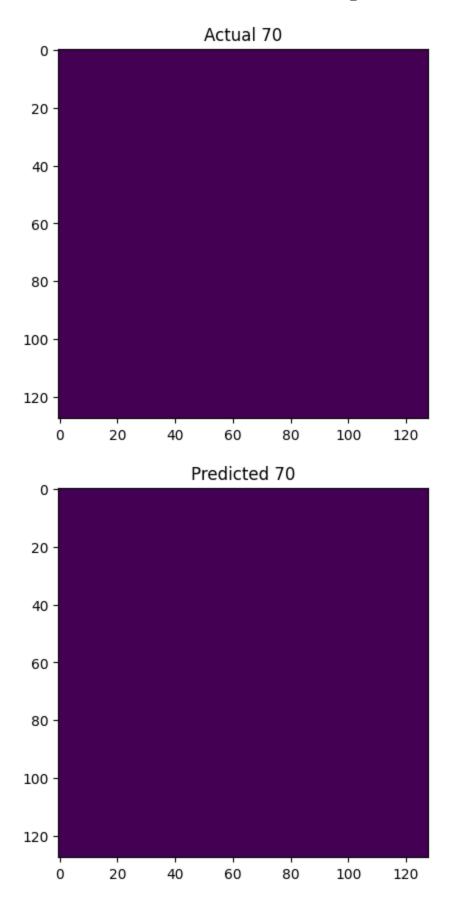
localhost:8888/lab 75/152



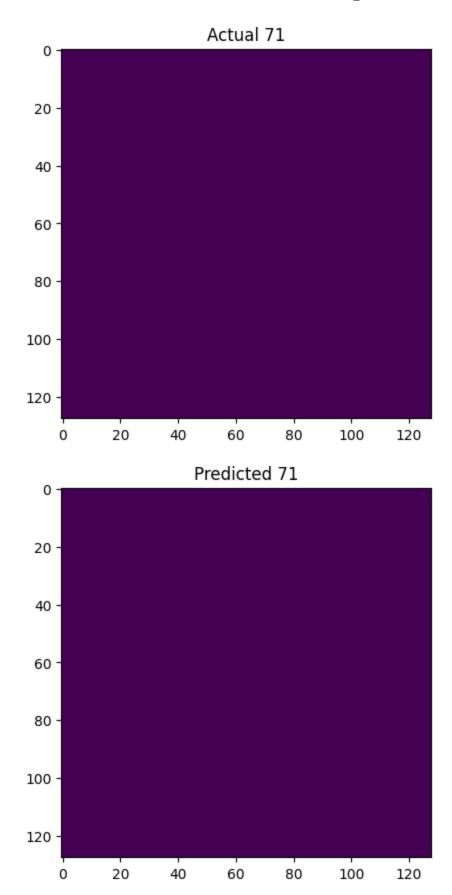
localhost:8888/lab 76/152



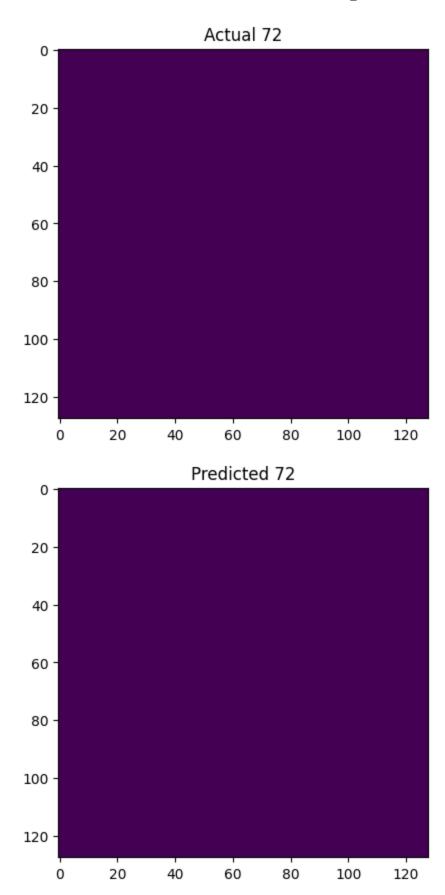
localhost:8888/lab 77/152



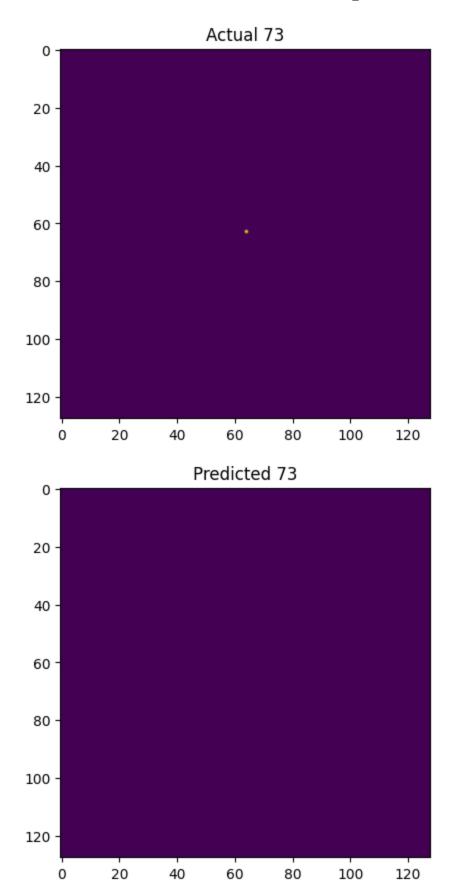
localhost:8888/lab 78/152



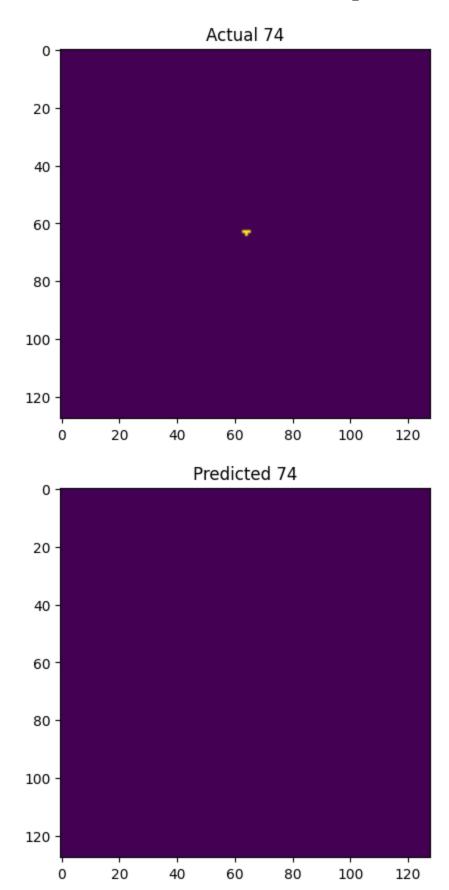
localhost:8888/lab 79/152



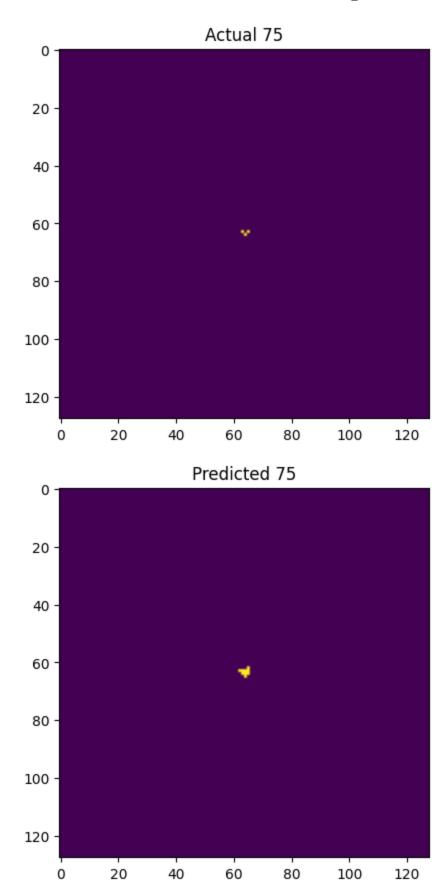
localhost:8888/lab 80/152



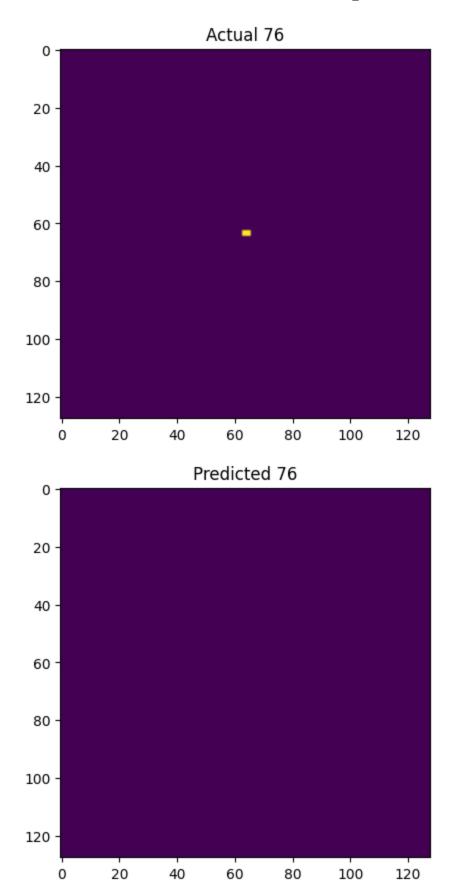
localhost:8888/lab 81/152



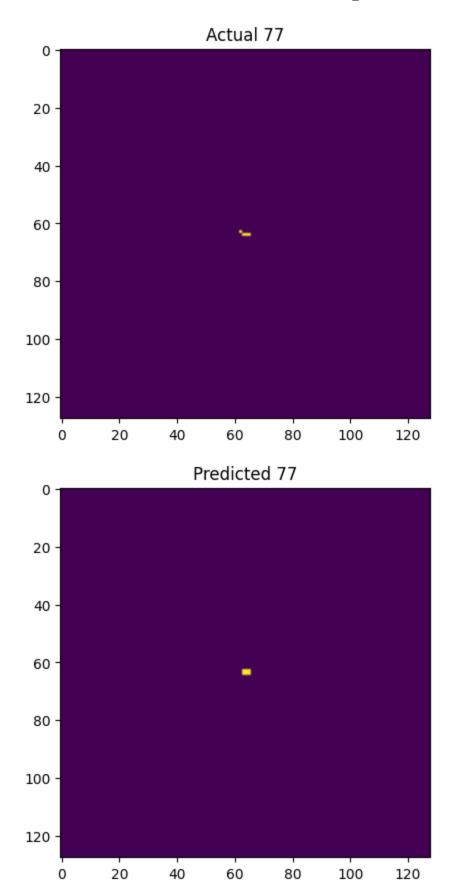
localhost:8888/lab 82/152



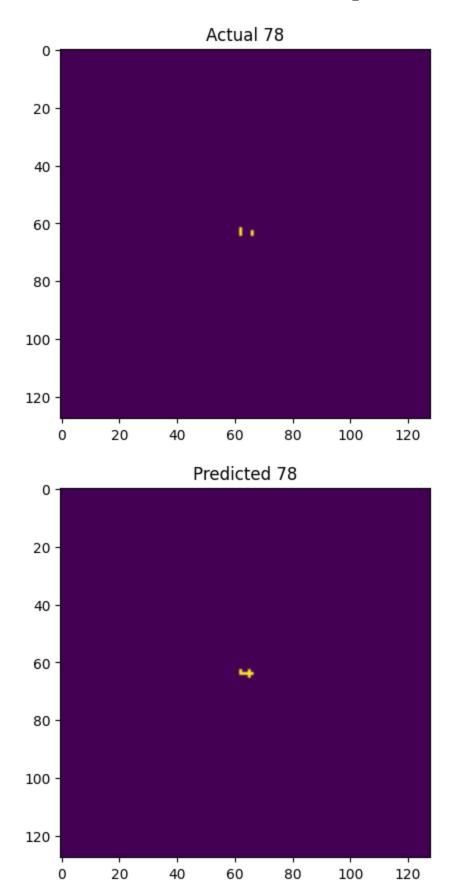
localhost:8888/lab 83/152



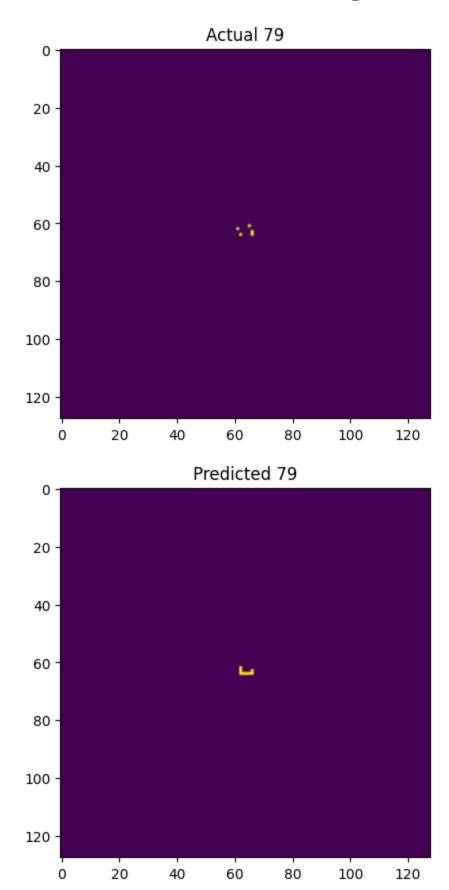
localhost:8888/lab 84/152



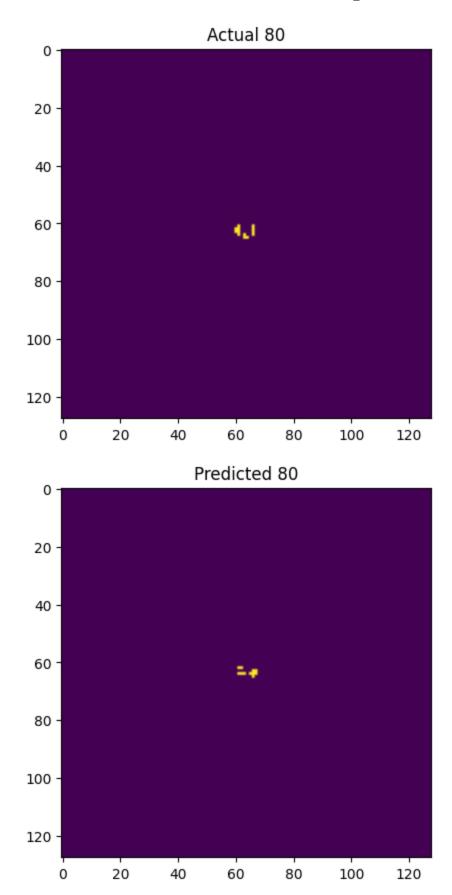
localhost:8888/lab 85/152



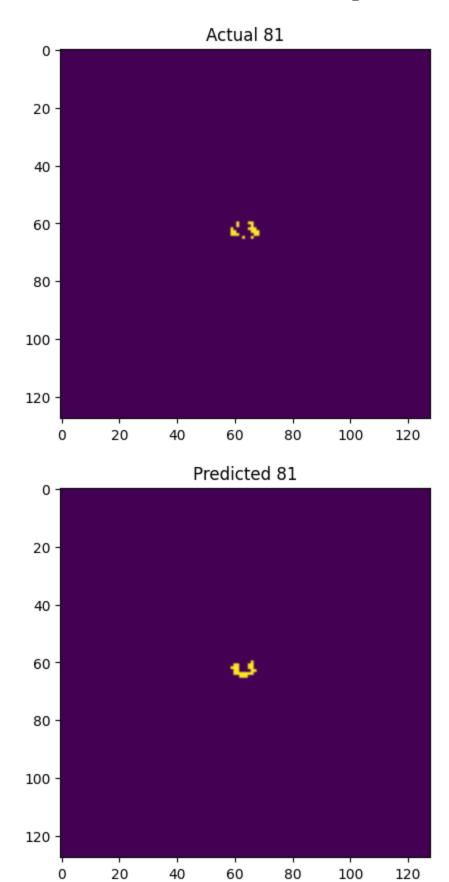
localhost:8888/lab 86/152



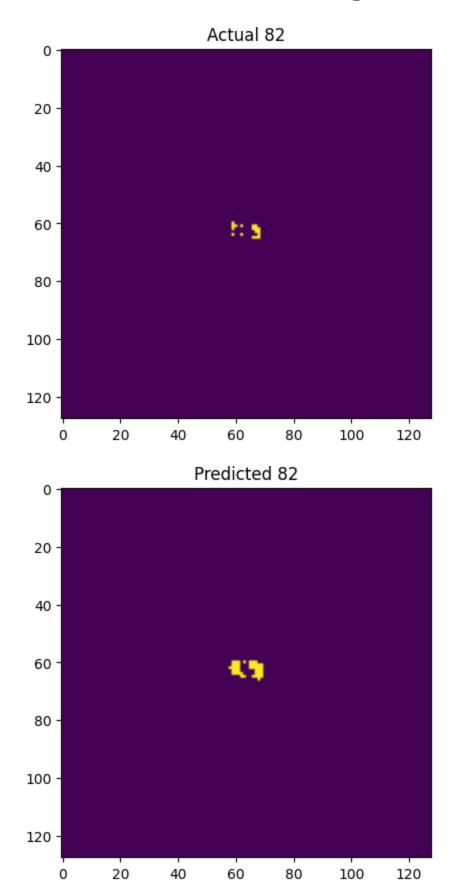
localhost:8888/lab 87/152



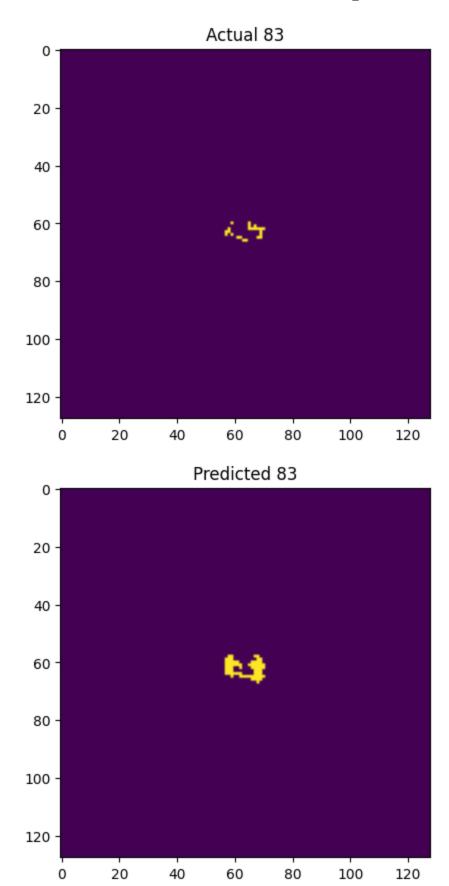
localhost:8888/lab 88/152



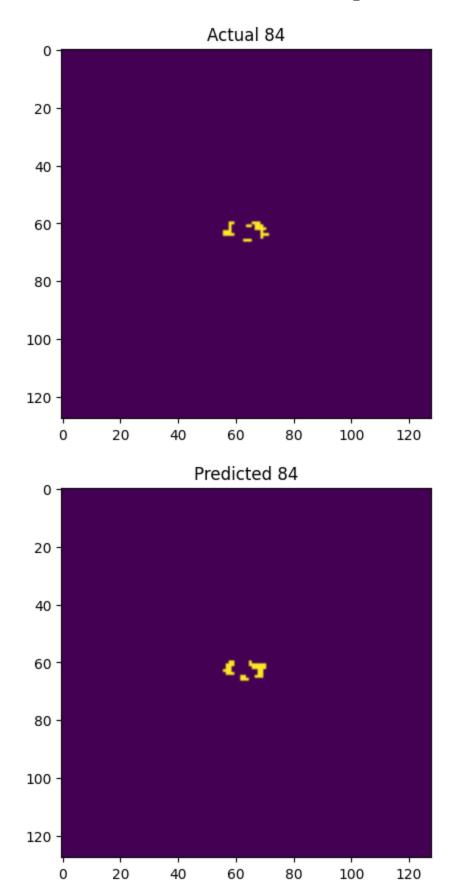
localhost:8888/lab 89/152



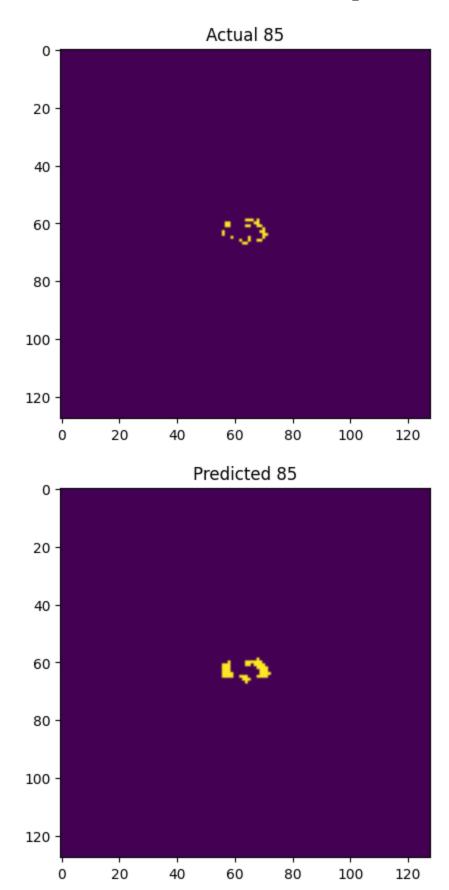
localhost:8888/lab 90/152



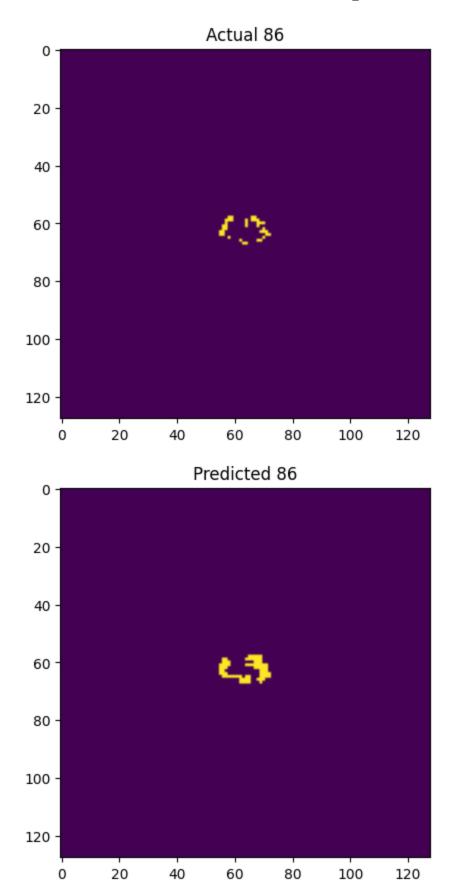
localhost:8888/lab 91/152



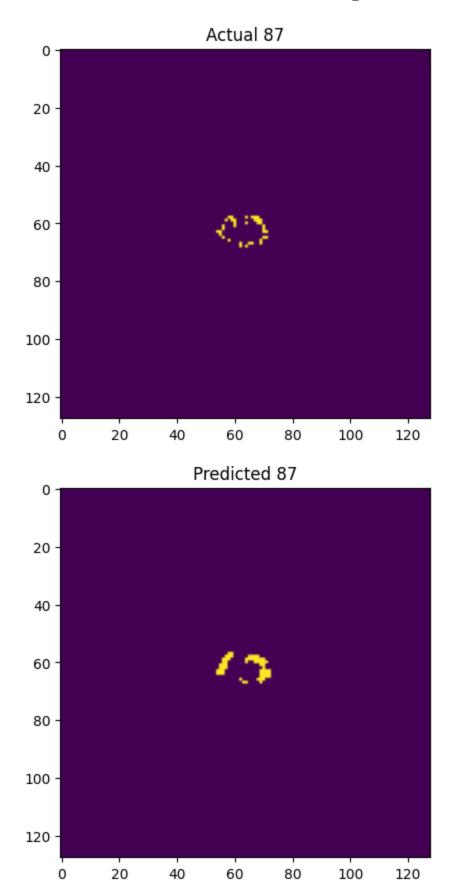
localhost:8888/lab 92/152



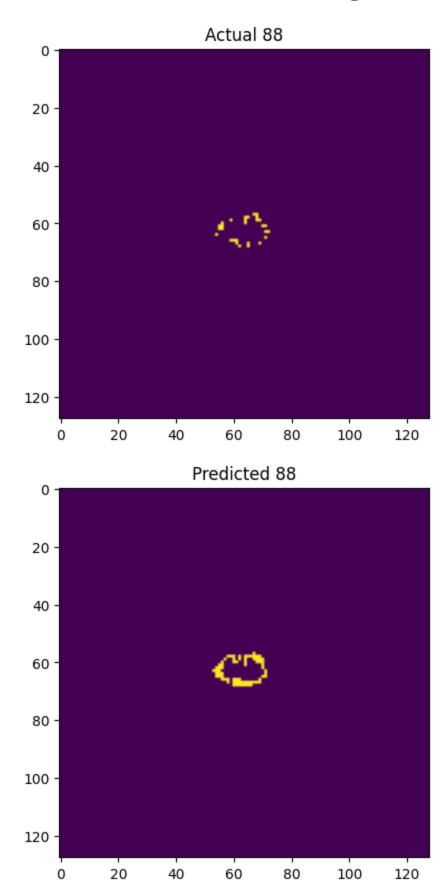
localhost:8888/lab 93/152



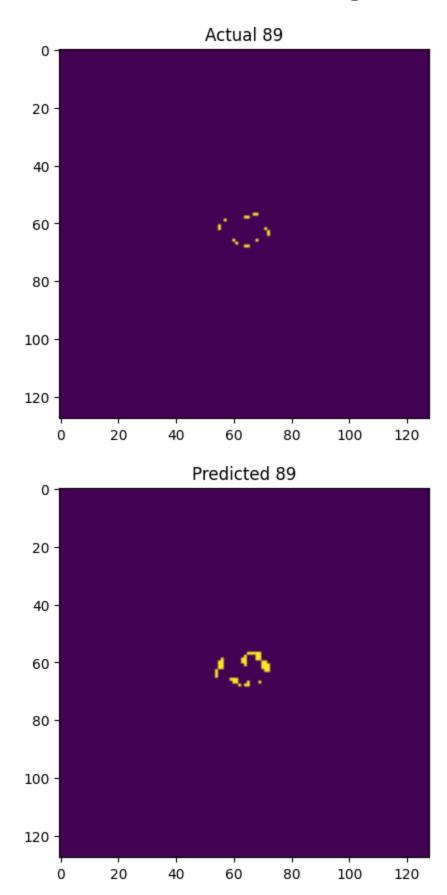
localhost:8888/lab 94/152



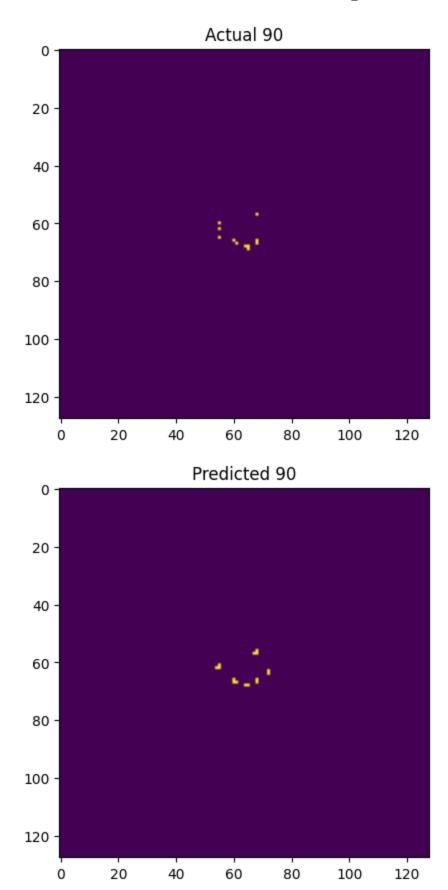
localhost:8888/lab 95/152



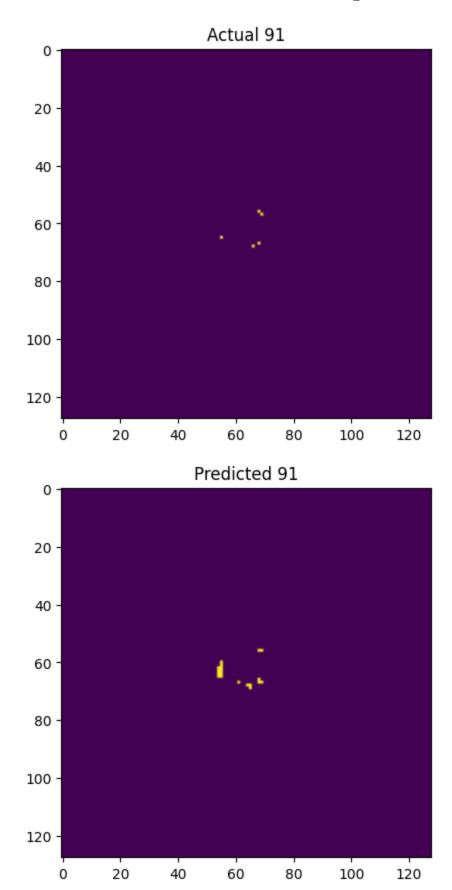
localhost:8888/lab 96/152



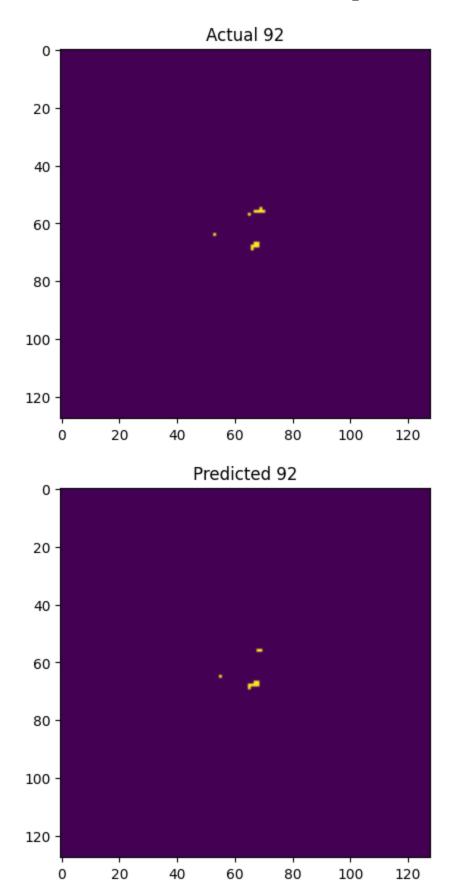
localhost:8888/lab 97/152



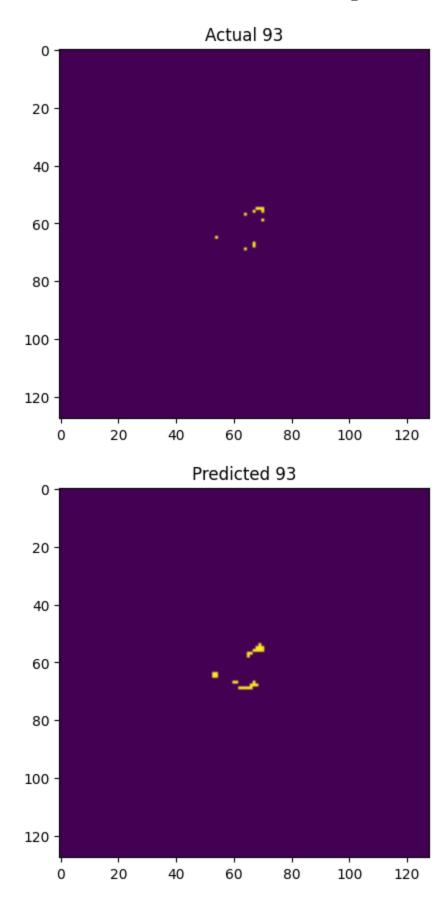
localhost:8888/lab 98/152



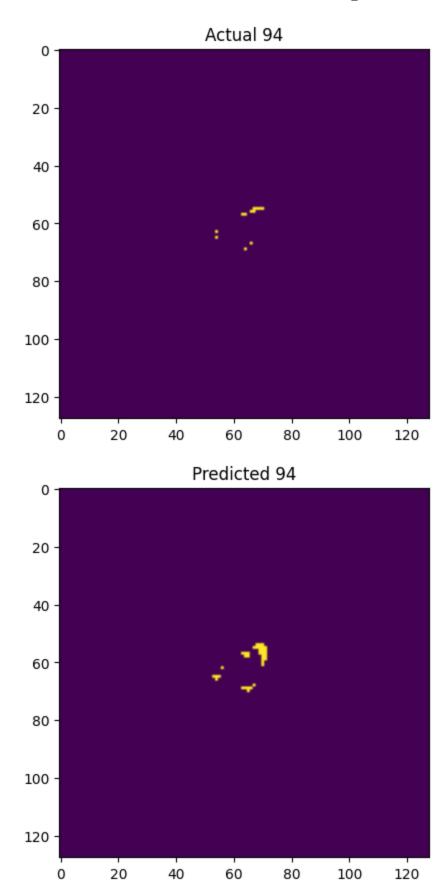
localhost:8888/lab 99/152



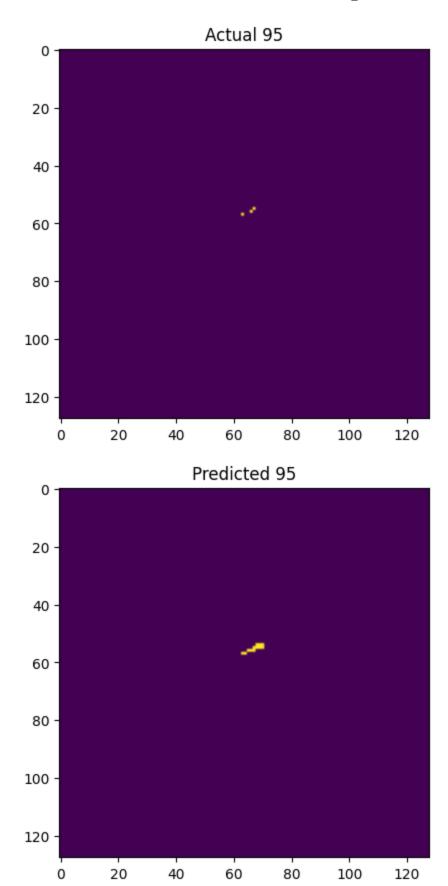
localhost:8888/lab 100/152



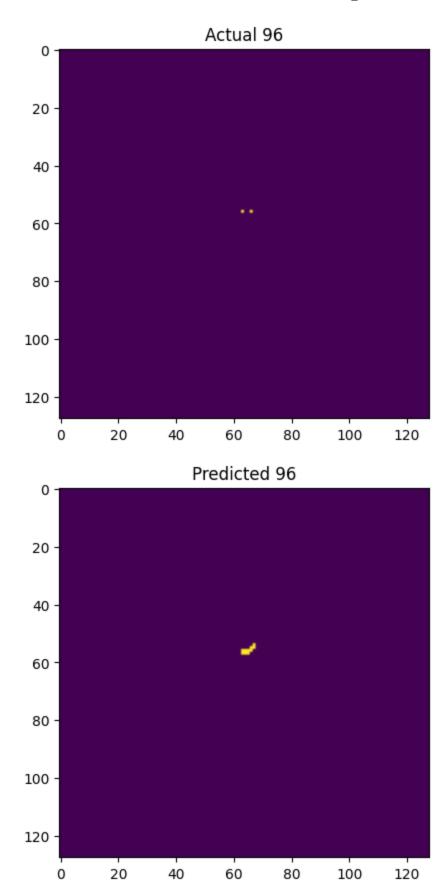
localhost:8888/lab 101/152



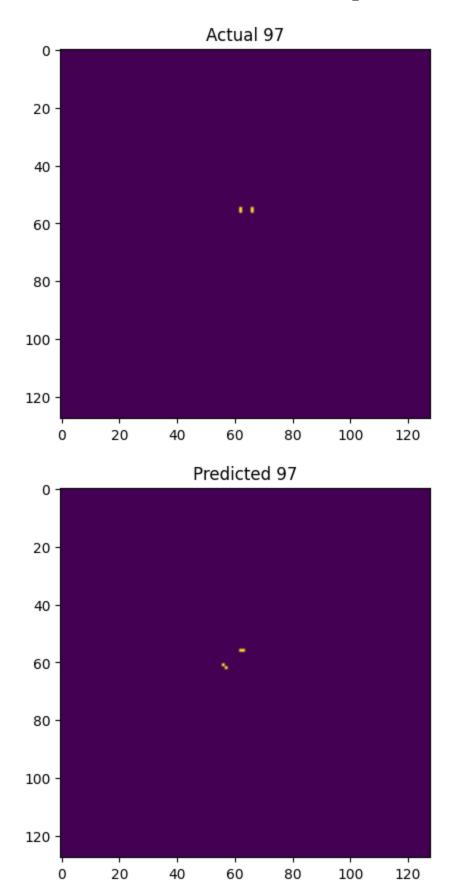
localhost:8888/lab 102/152



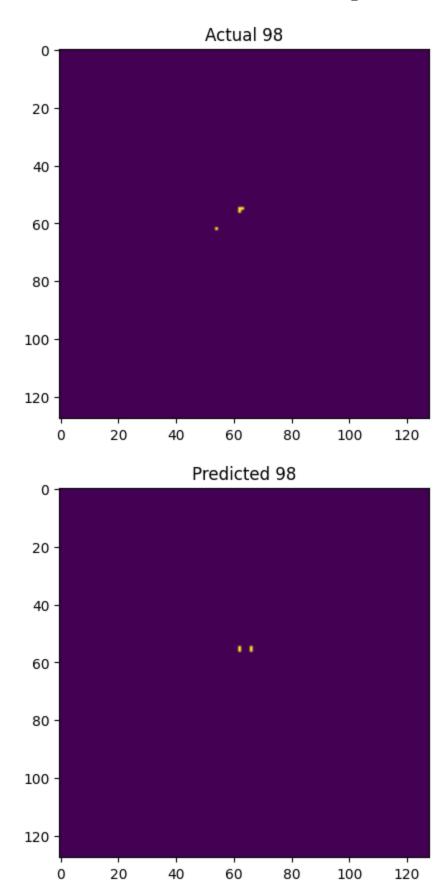
localhost:8888/lab 103/152



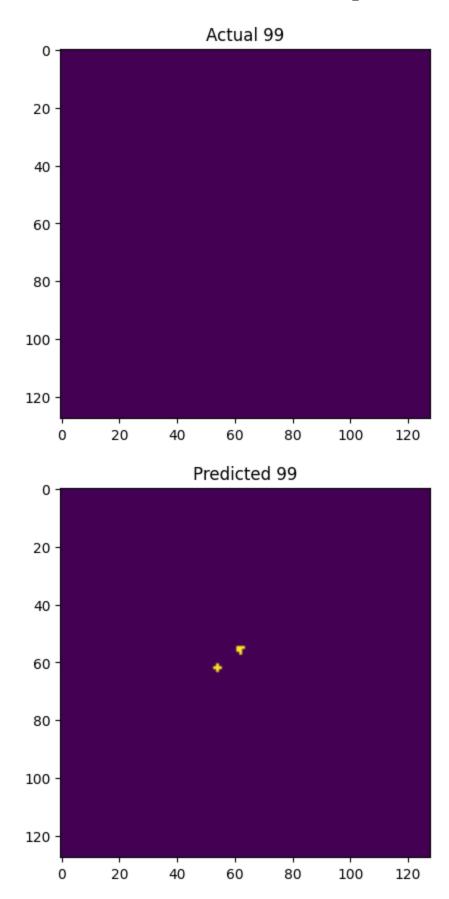
localhost:8888/lab 104/152



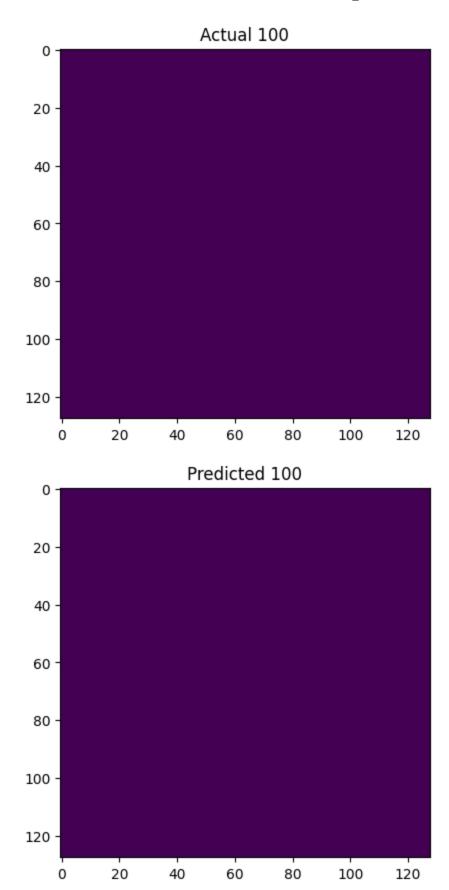
localhost:8888/lab 105/152



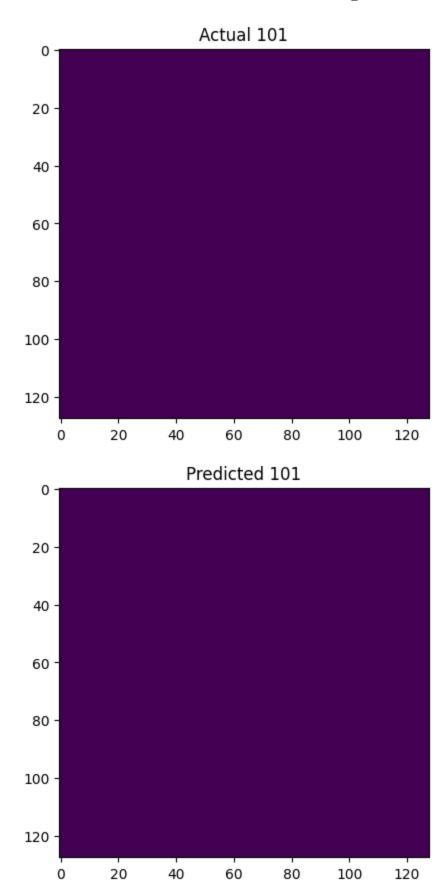
localhost:8888/lab 106/152



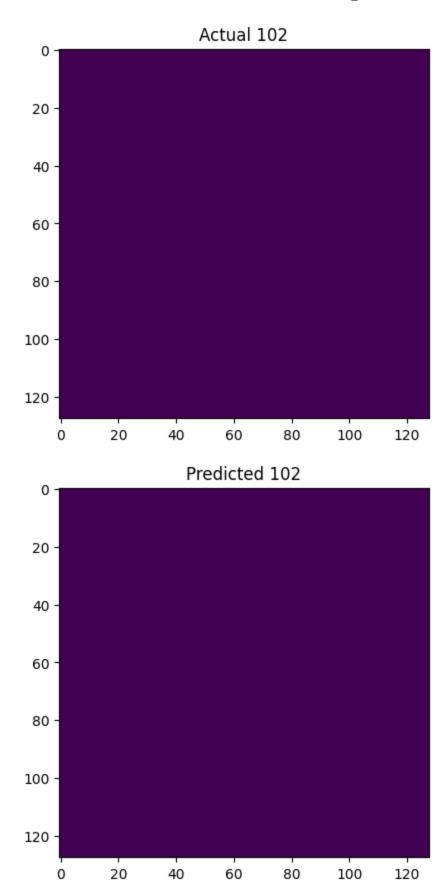
localhost:8888/lab 107/152



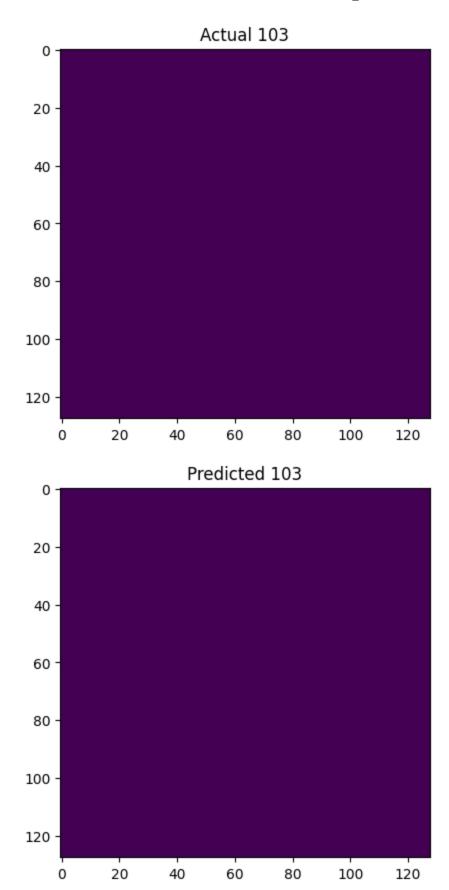
localhost:8888/lab 108/152



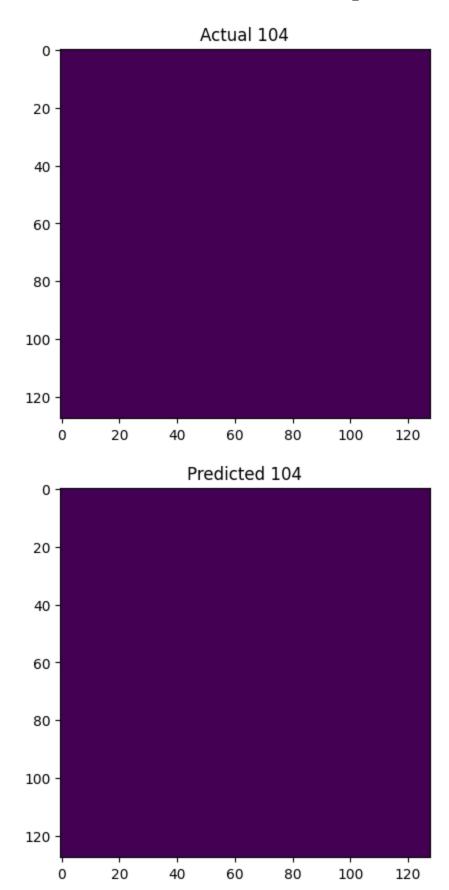
localhost:8888/lab 109/152



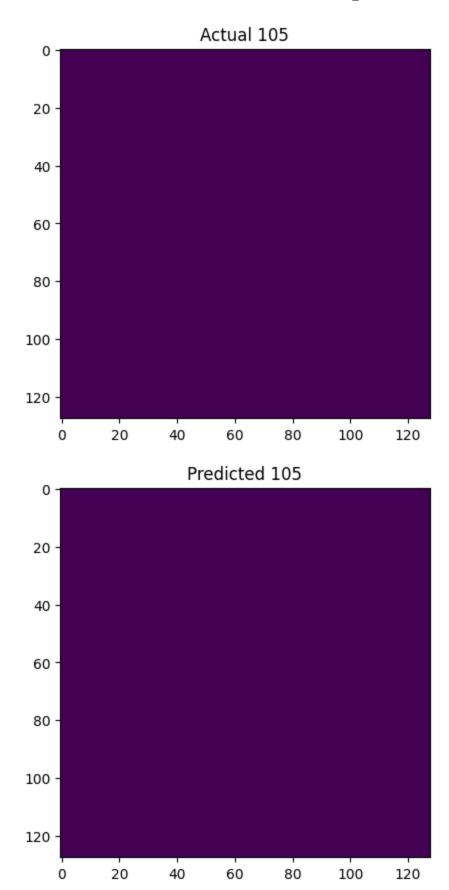
localhost:8888/lab 110/152



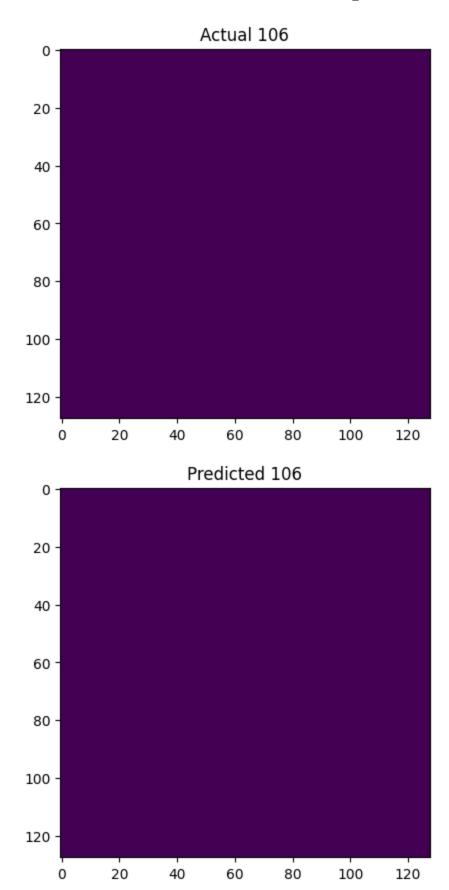
localhost:8888/lab 111/152



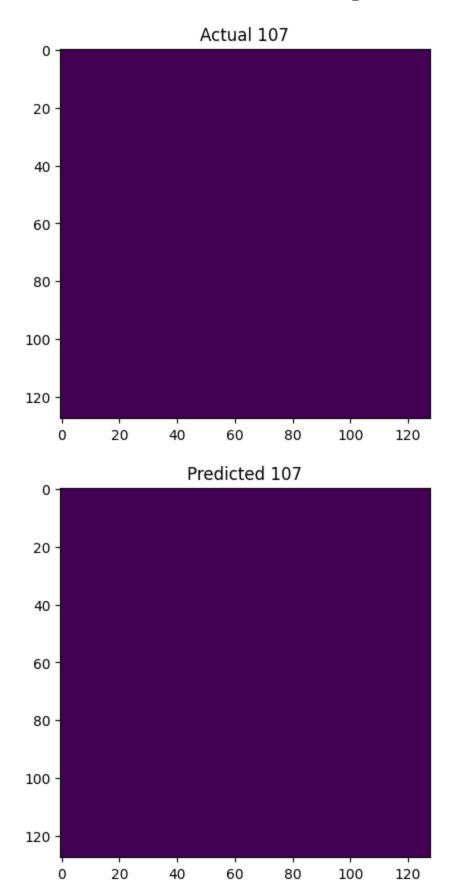
localhost:8888/lab 112/152



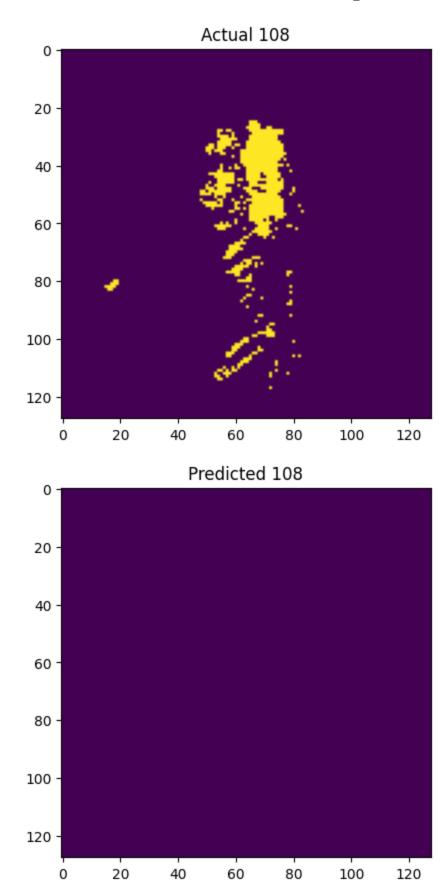
localhost:8888/lab 113/152



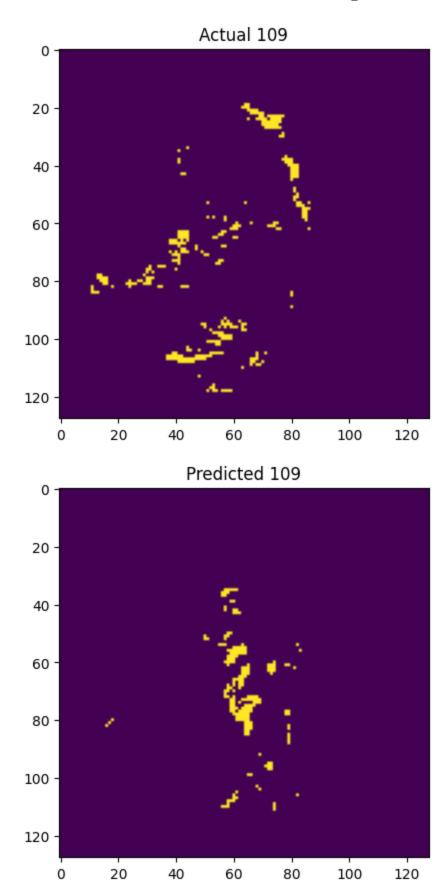
localhost:8888/lab 114/152



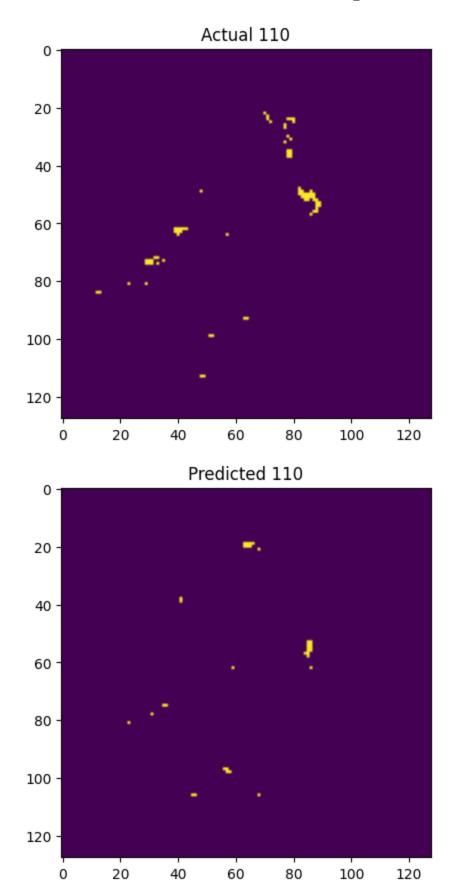
localhost:8888/lab 115/152



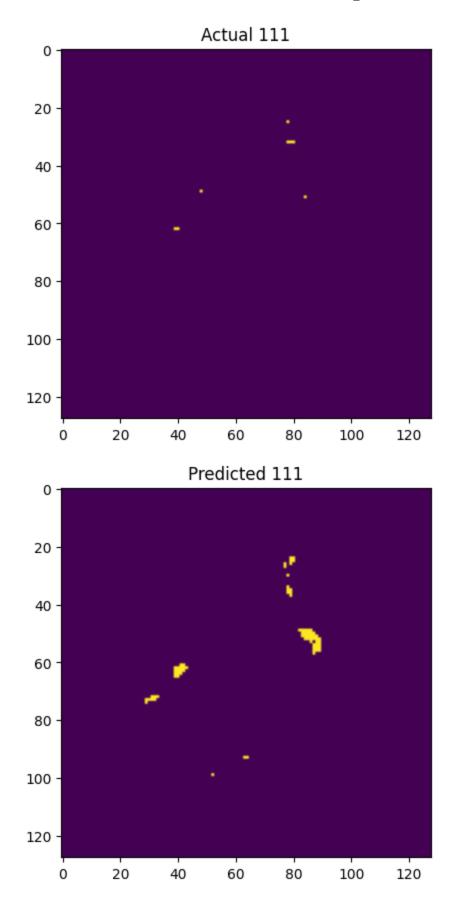
localhost:8888/lab 116/152



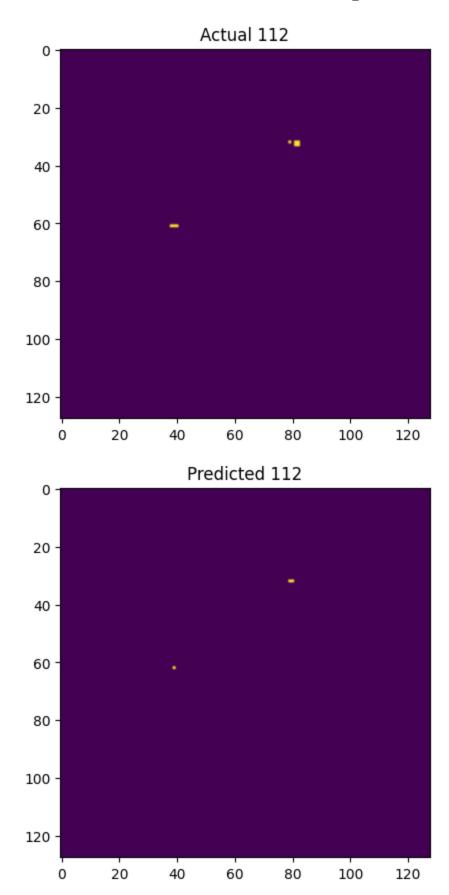
localhost:8888/lab 117/152



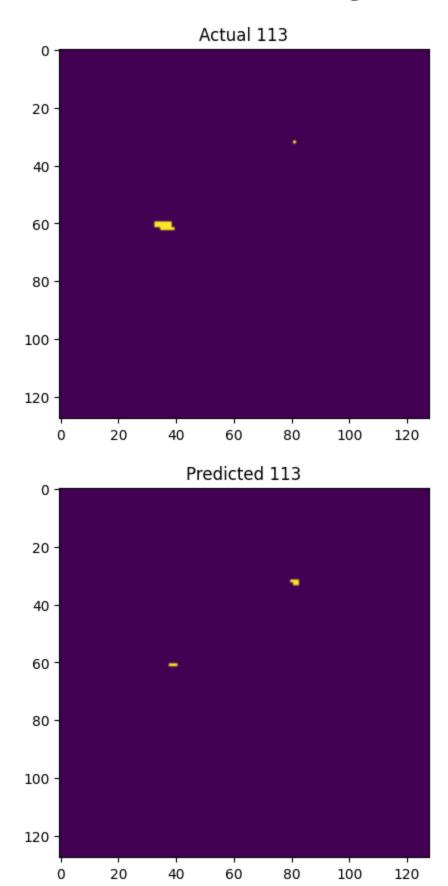
localhost:8888/lab 118/152



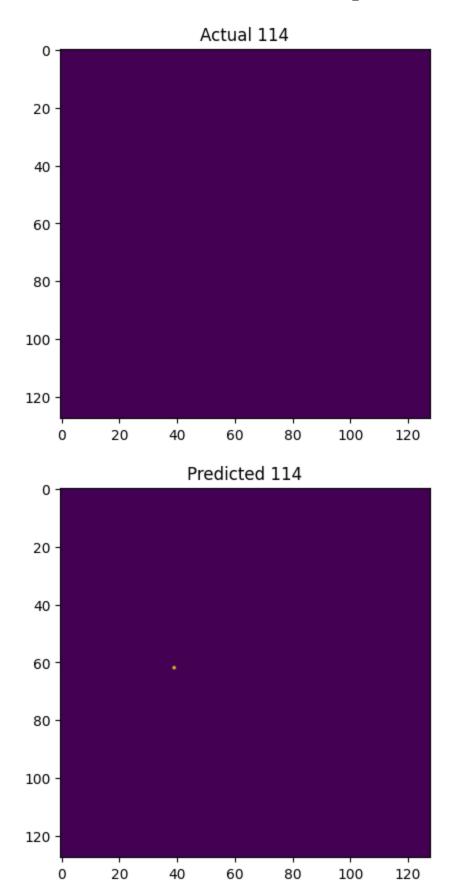
localhost:8888/lab 119/152



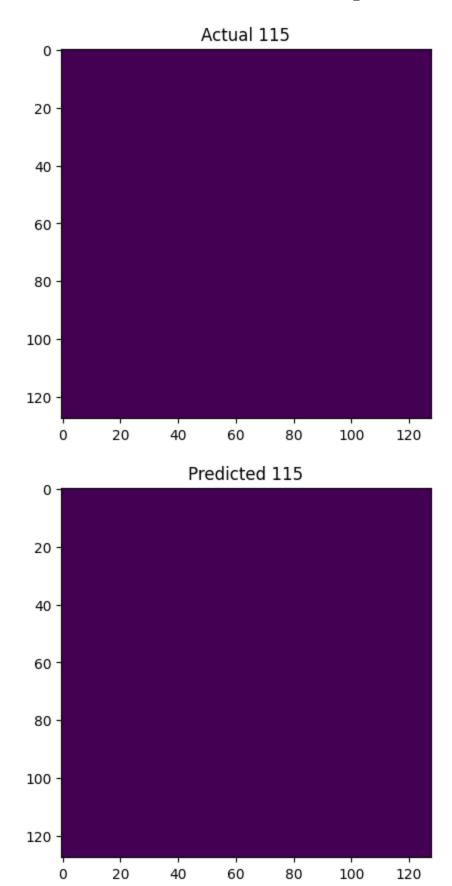
localhost:8888/lab 120/152



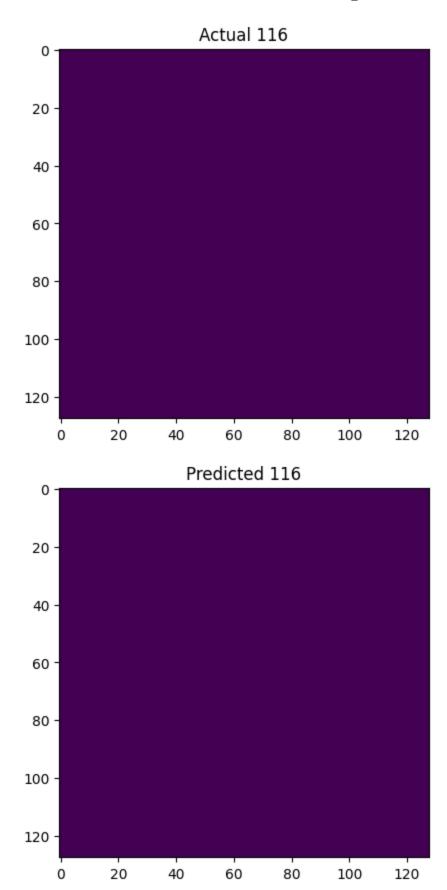
localhost:8888/lab 121/152



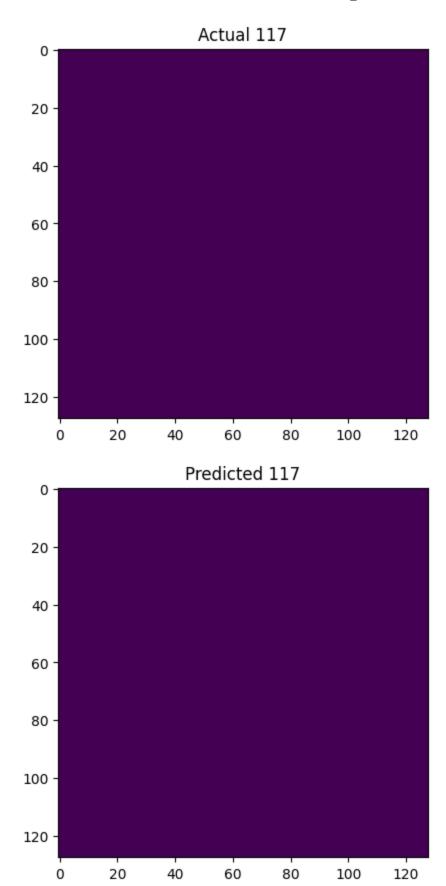
localhost:8888/lab 122/152



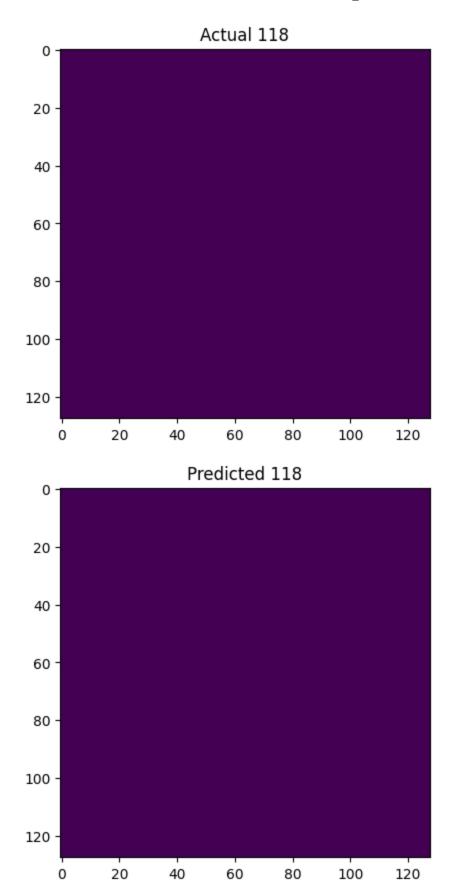
localhost:8888/lab 123/152



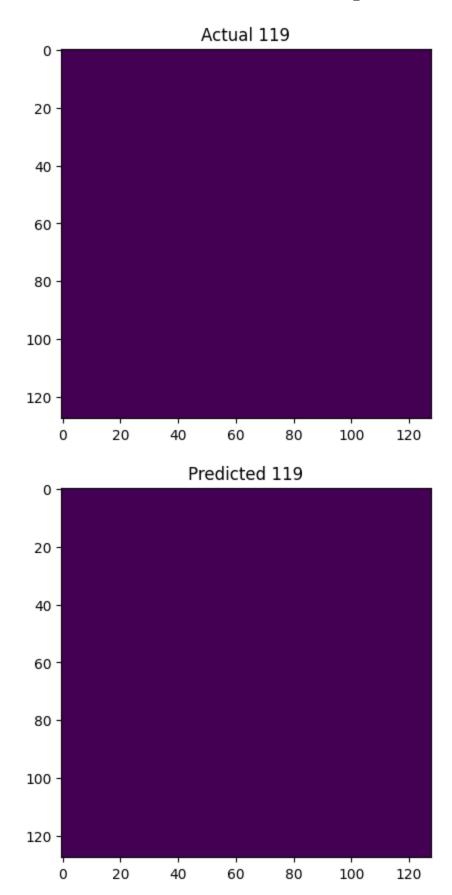
localhost:8888/lab 124/152



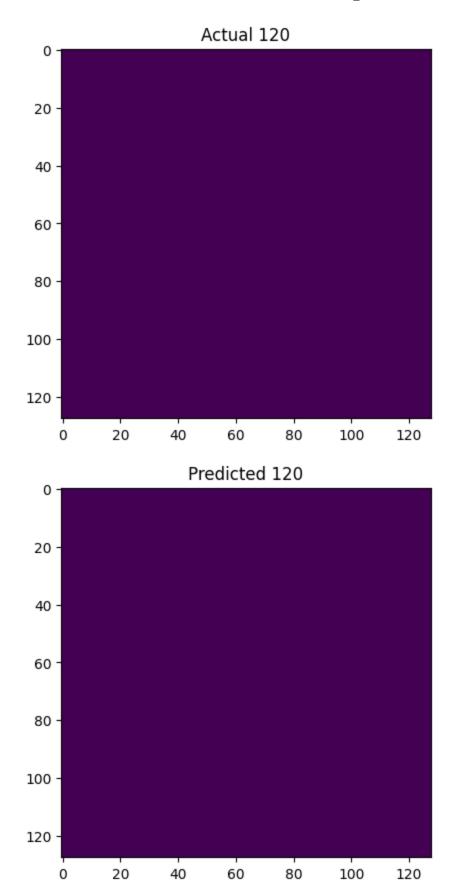
localhost:8888/lab 125/152



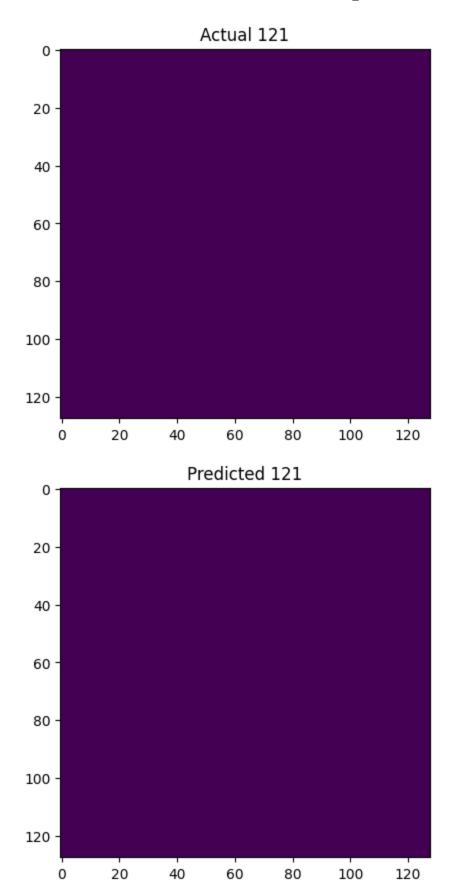
localhost:8888/lab 126/152



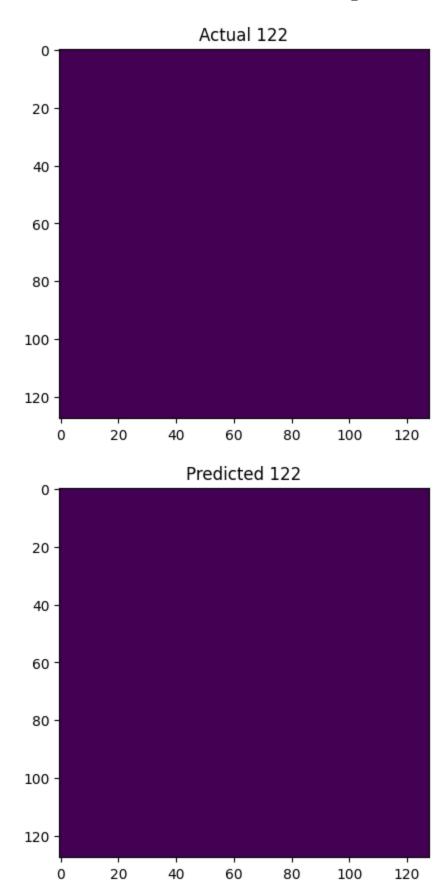
localhost:8888/lab 127/152



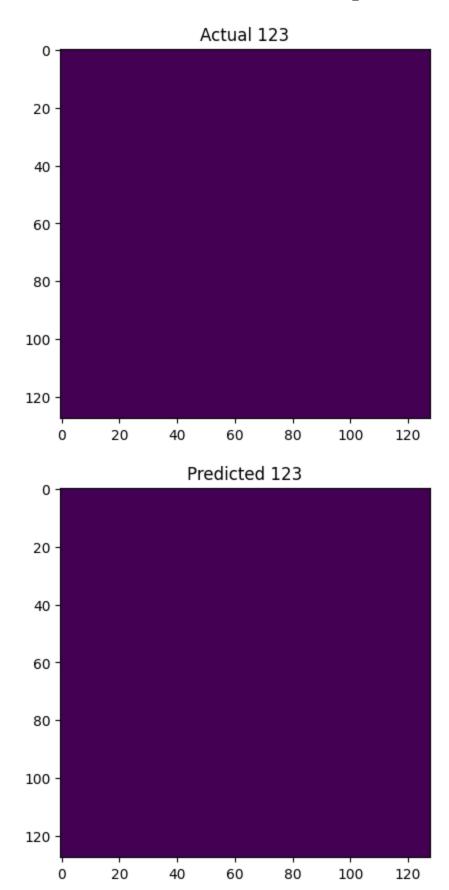
localhost:8888/lab 128/152



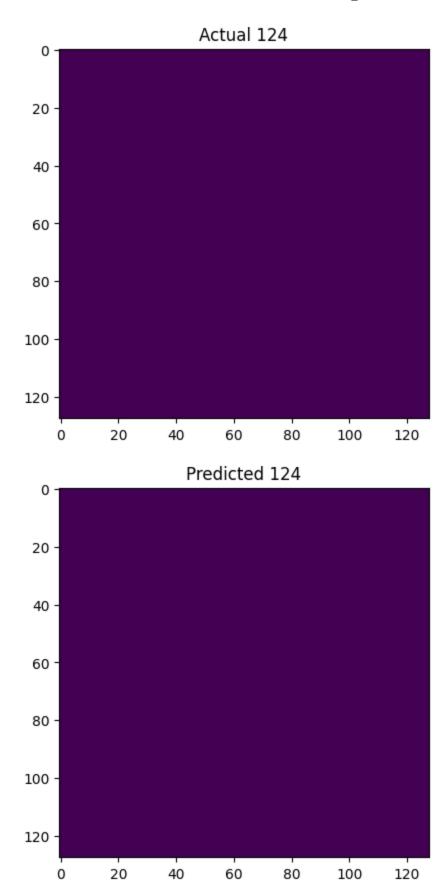
localhost:8888/lab 129/152



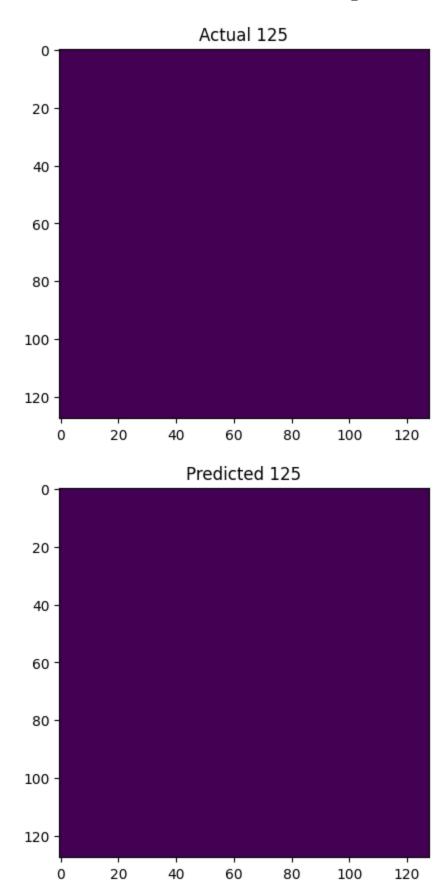
localhost:8888/lab 130/152



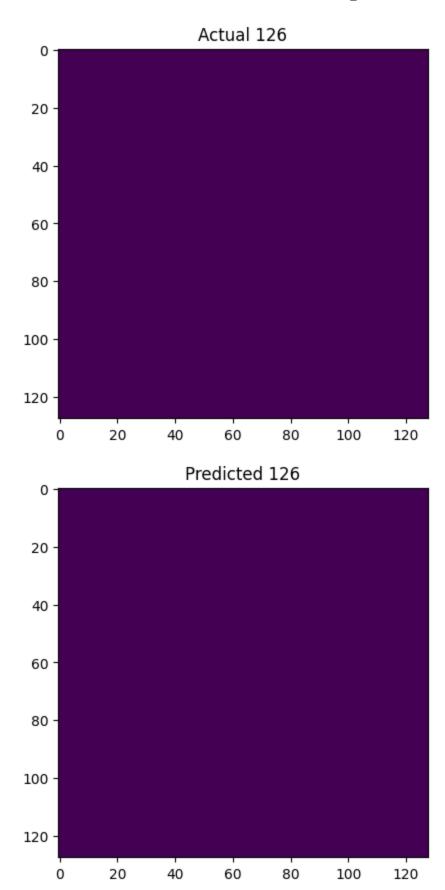
localhost:8888/lab 131/152



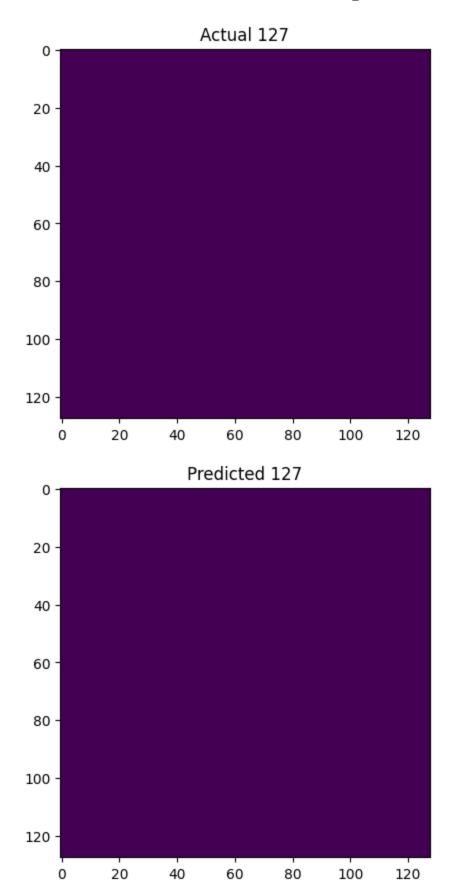
localhost:8888/lab 132/152



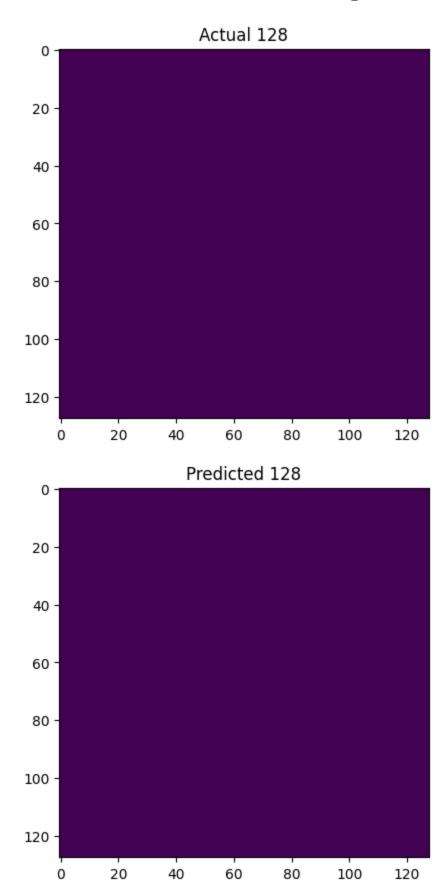
localhost:8888/lab 133/152



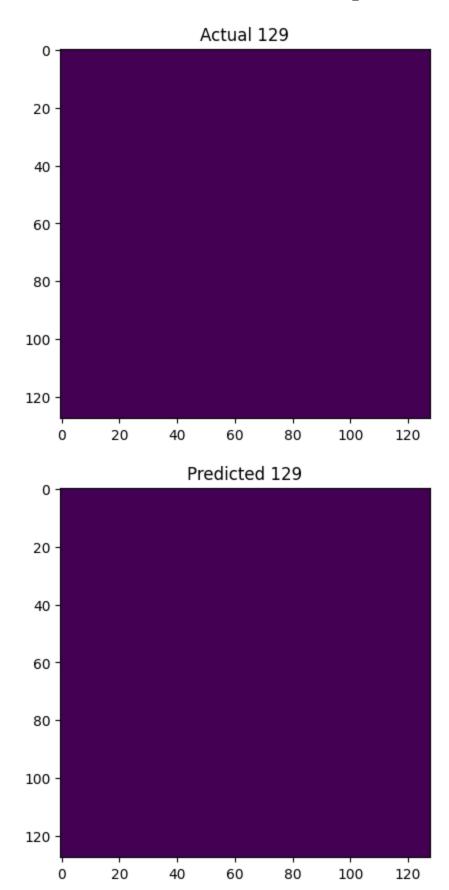
localhost:8888/lab 134/152



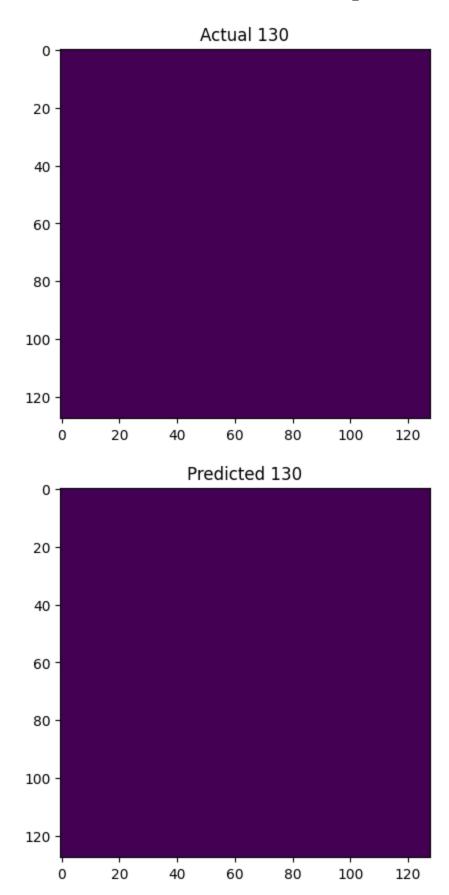
localhost:8888/lab 135/152



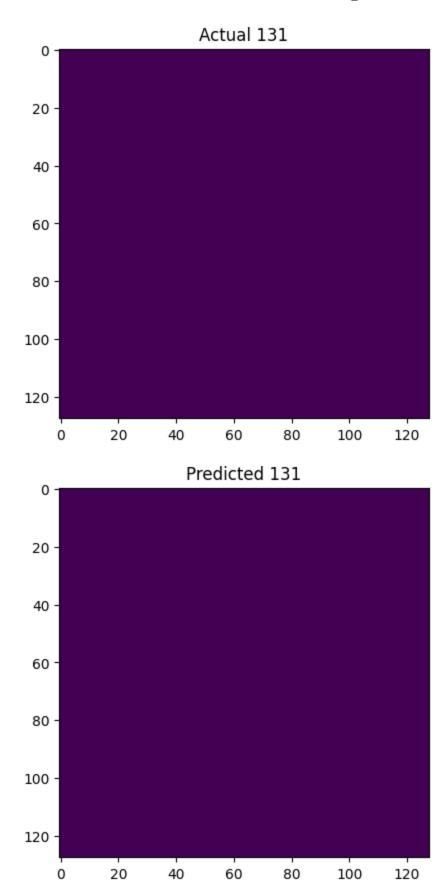
localhost:8888/lab 136/152



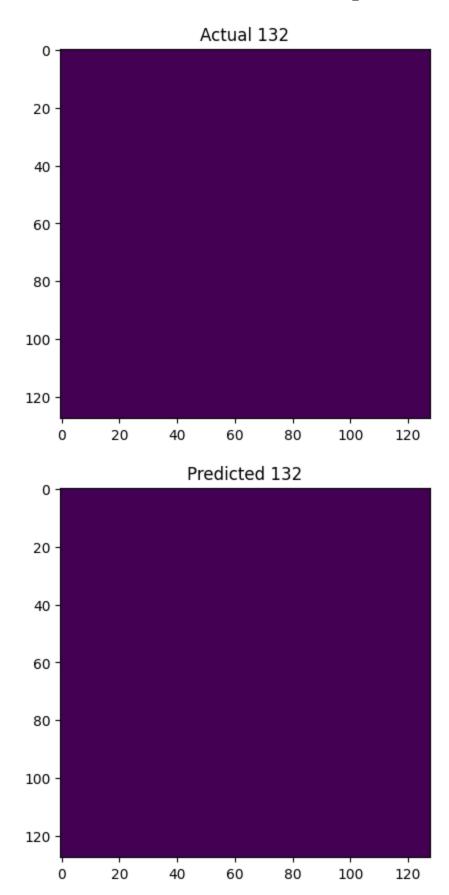
localhost:8888/lab 137/152



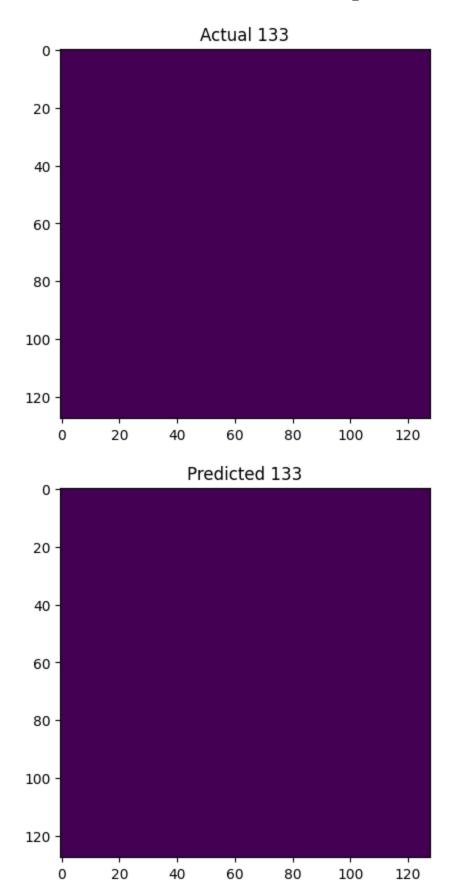
localhost:8888/lab 138/152



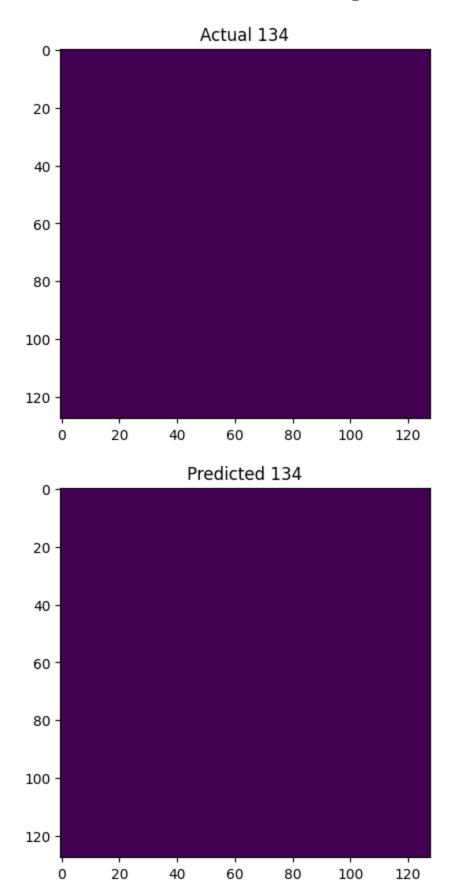
localhost:8888/lab 139/152



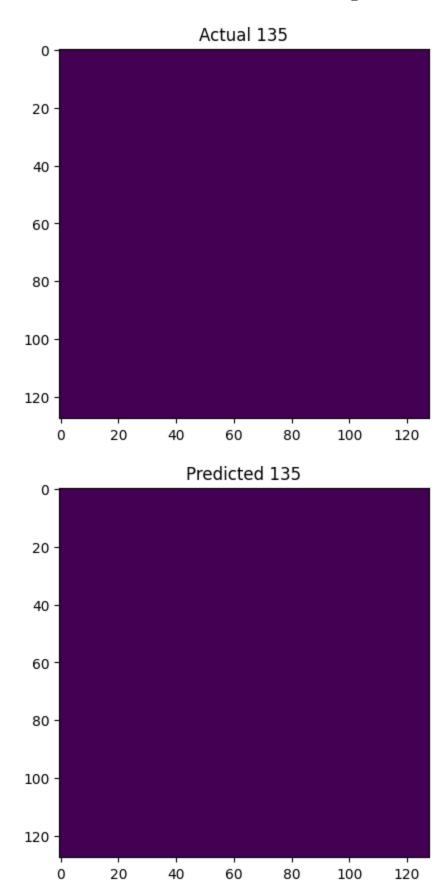
localhost:8888/lab 140/152



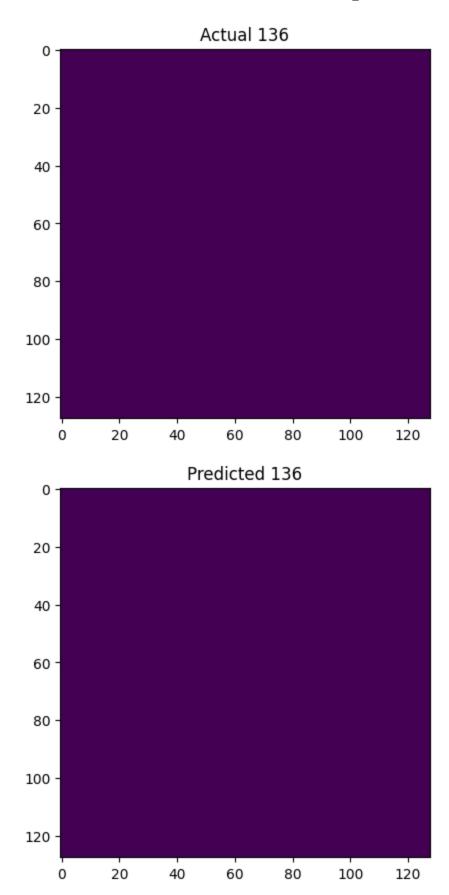
localhost:8888/lab 141/152



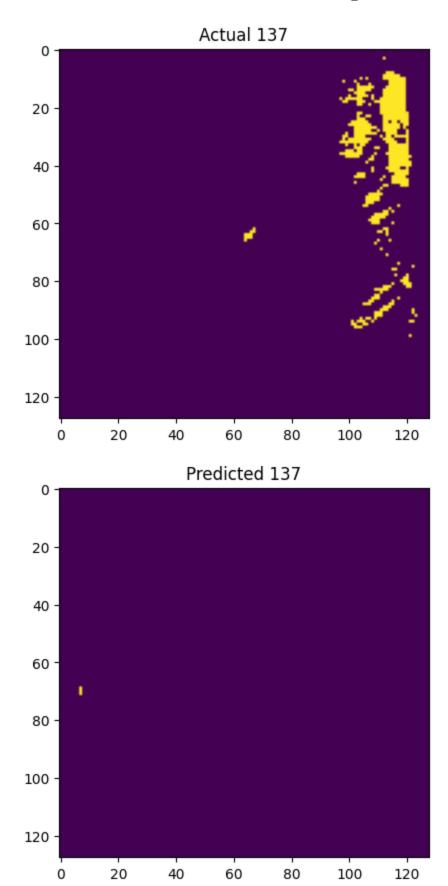
localhost:8888/lab 142/152



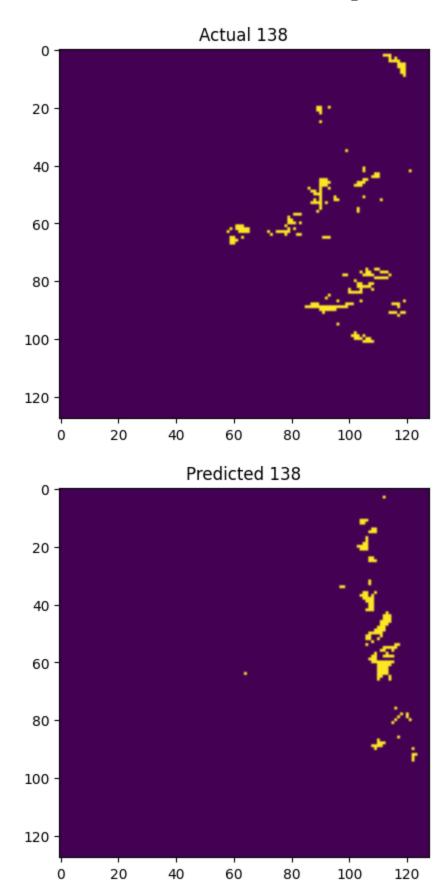
localhost:8888/lab 143/152



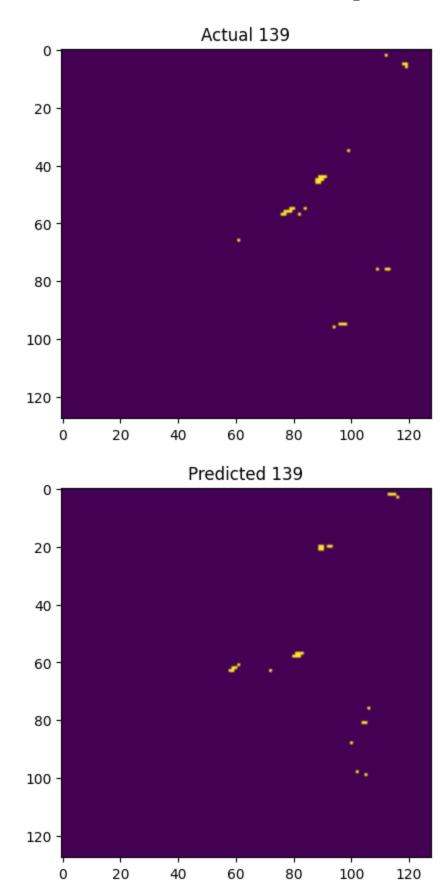
localhost:8888/lab 144/152



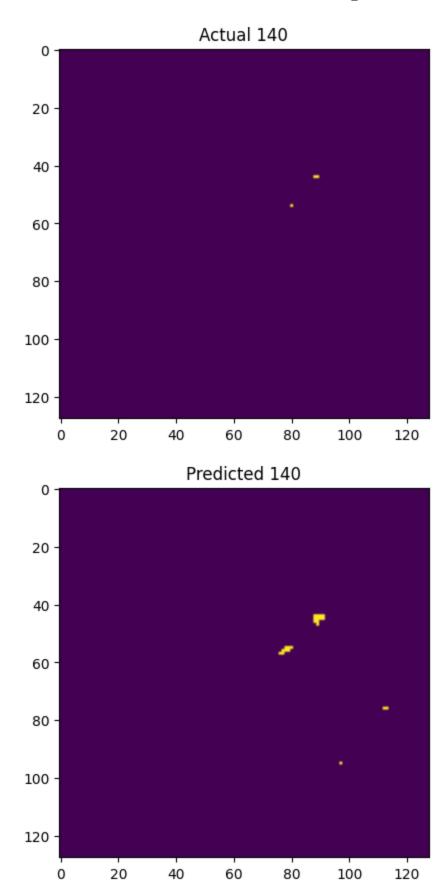
localhost:8888/lab 145/152



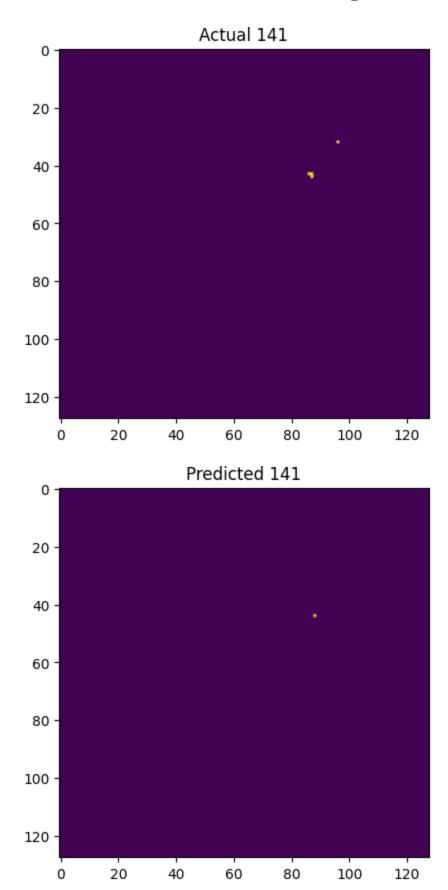
localhost:8888/lab 146/152



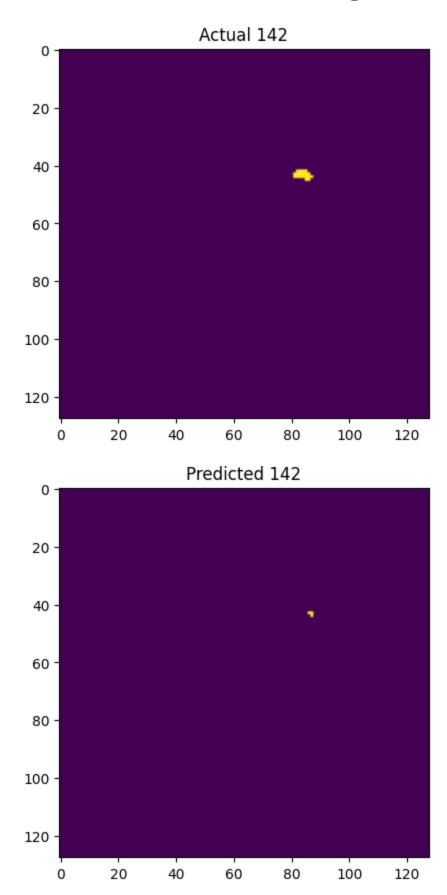
localhost:8888/lab 147/152



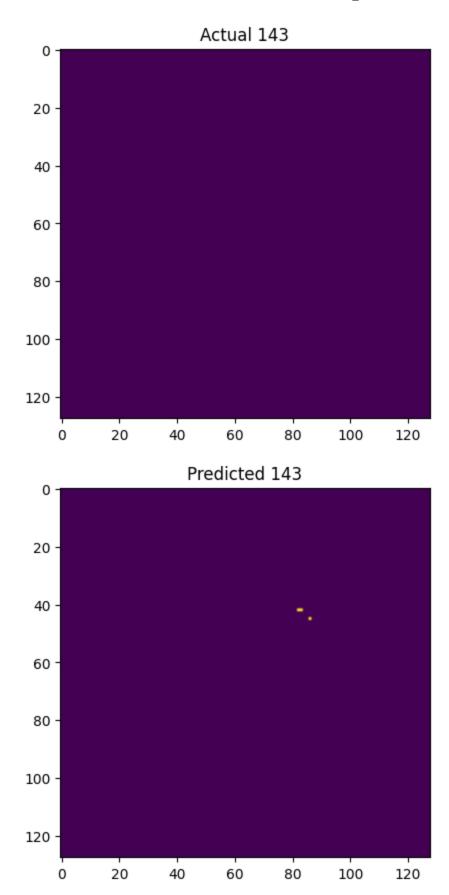
localhost:8888/lab 148/152



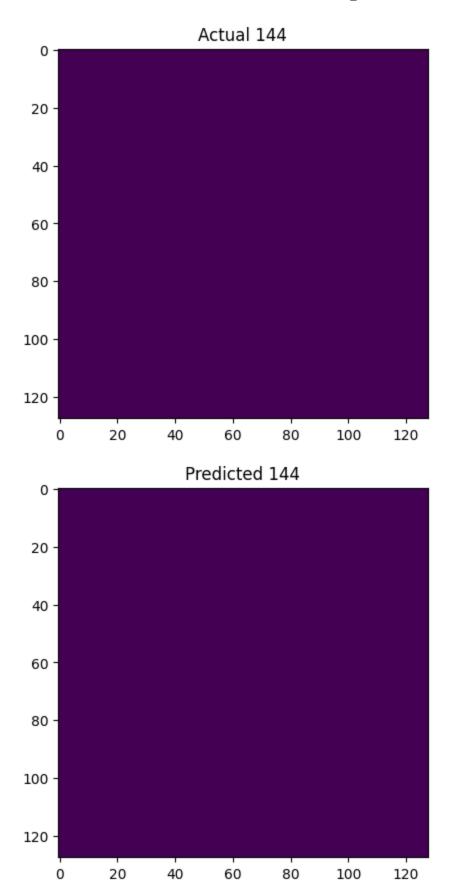
localhost:8888/lab 149/152



localhost:8888/lab 150/152



localhost:8888/lab 151/152



In [ ]:

localhost:8888/lab 152/152