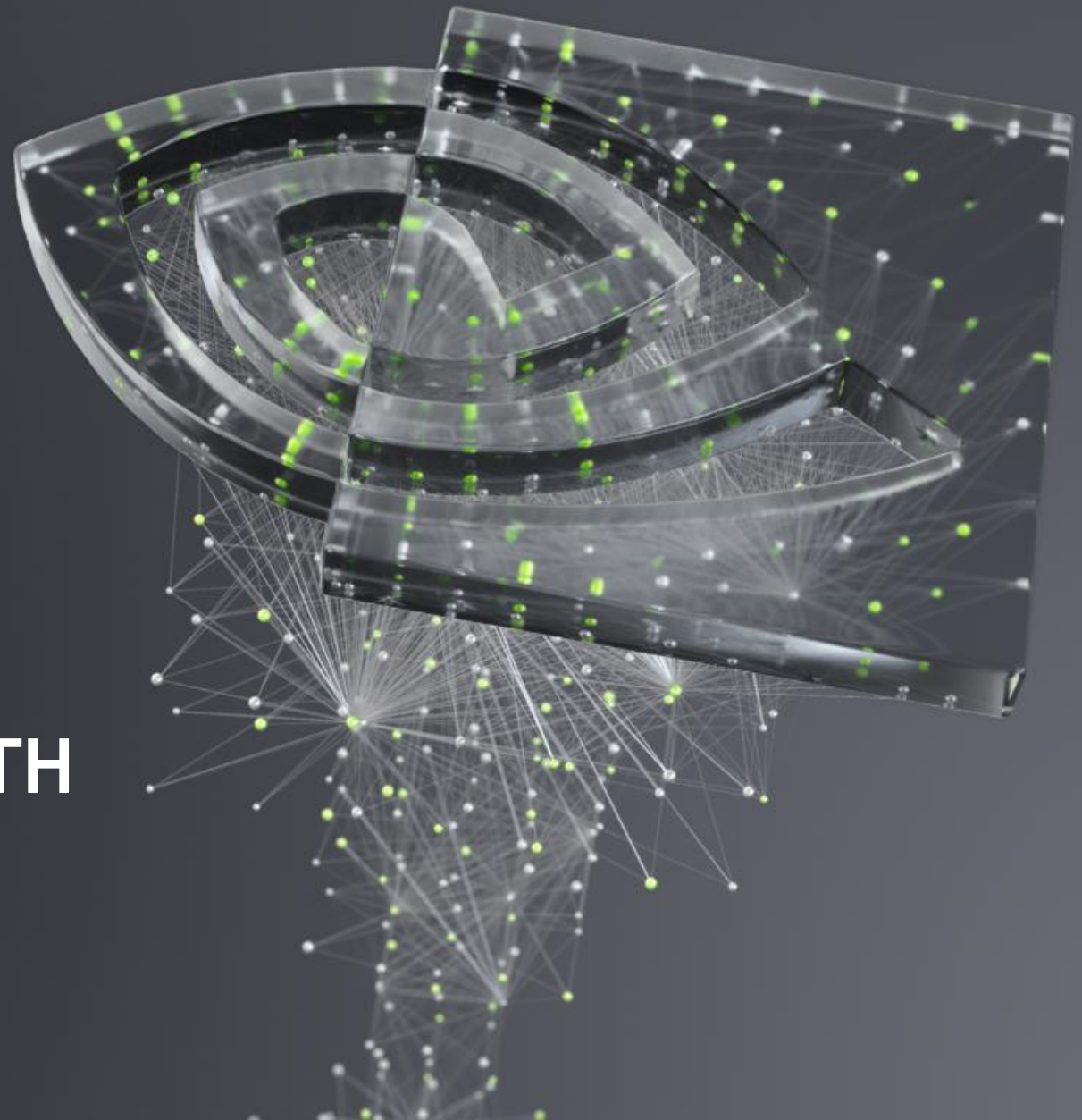


NVIDIA | DEEP LEARNING INSTITUTE

GETTING STARTED WITH IMAGE SEGMENTATION

WHAT THIS COURSE IS

Neural Networks for Medical Imaging

Discussion/Demonstration of Image Segmentation using Deep Learning

Transposed Convolutional Neural Networks

Hands-on exercises using TensorFlow for CNN training and evaluation of Image Segmentation workflow

Modern Deep Learning Frameworks

Get practice using TensorFlow 2 and the Keras API

WHAT THIS COURSE ISN'T

An introduction to Deep Neural Networks

A mathematical explanation of Convolution and CNNs

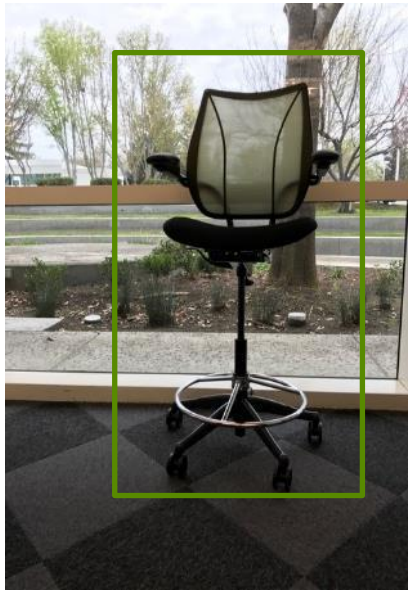
A survey of all the features and options of TensorFlow

IMAGE SEGMENTATION

COMPUTER VISION TASKS

Image Classification

Image Classification + Localization



Object Detection

Image Segmentation

(inspired by a slide used in cs231n lecture from Stanford University)

IMAGE SEGMENTATION

- “Segmentation” sometimes used to describe similar but slightly different tasks
- In this lab, semantic segmentation will be performed
  - i.e., in an image, each pixel will be placed into one of multiple classes
- In a sense it’s a classification problem where each pixel has a class, vs image recognition where each image (collection of pixels) has a class
- Specifically we’ll be looking at medical imaging data and attempting to determine where the left ventricle (LV) is
  - i.e., for each pixel is it part of LV or not?

TENSORFLOW

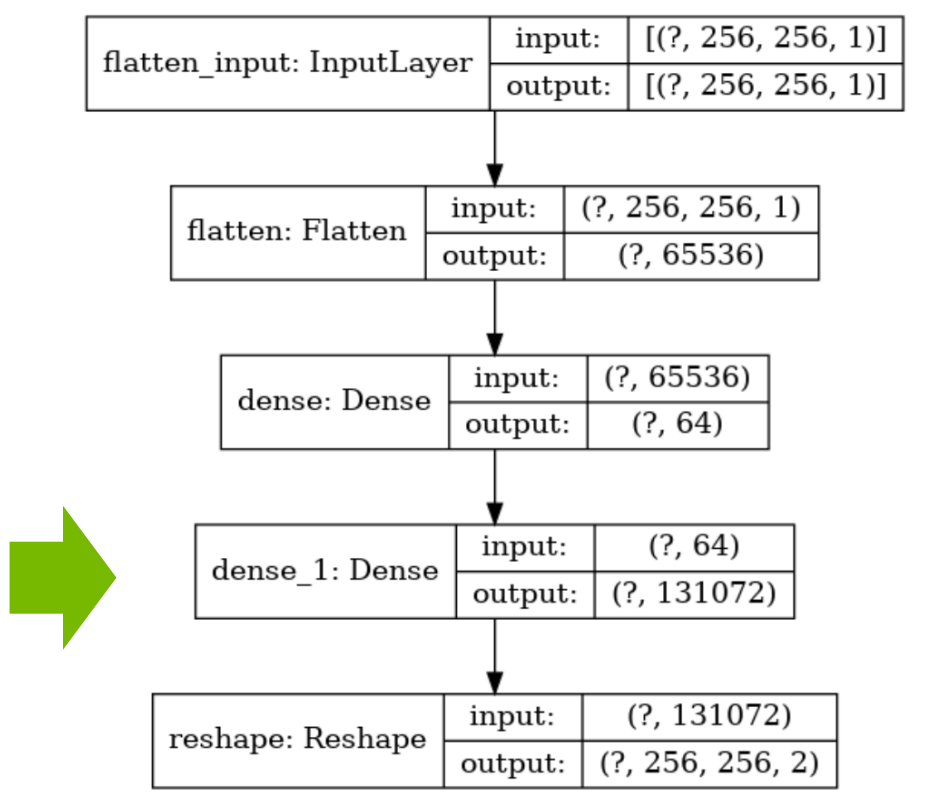
WHAT IS TENSORFLOW?
Created by Google, tensorflow.org

- "Open source software library for machine intelligence"
  - Available on GitHub
- Flexibility—express your computation as a data flow graph
  - If you can express it in TF syntax, you can run it
- Portability—CPUs and GPUs, workstation, server, mobile
- Language options—Python, C++, Go, Java
- Performance—Tuned for performance on CPUs and GPUs
  - Assign tasks to different hardware devices
  - Uses CUDNN

WHAT IS KERAS?
Created by keras.io

- "Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow"
  - Comes preinstalled with TensorFlow 2

```
# set up the model architecture
model = tf.keras.models.Sequential([
    Flatten(input_shape=[256, 256, 1]),
    Dense(64, activation='relu'),
    Dense(256*256*2, activation='softmax'),
    Reshape((256, 256, 2))
])
```



SAMPLE WORKFLOW

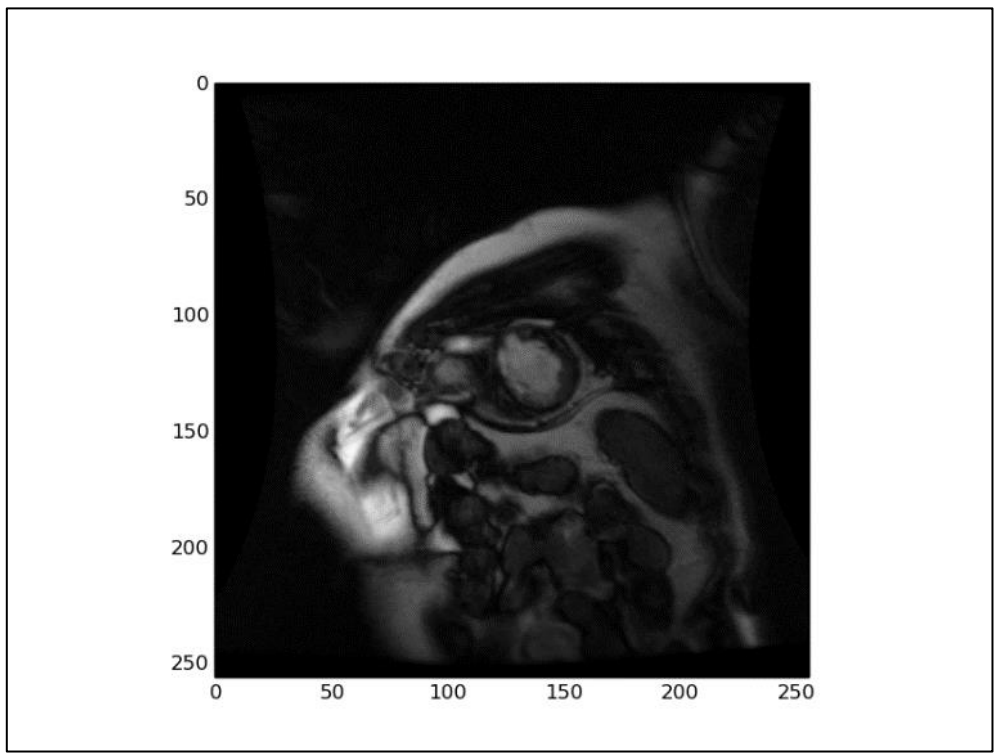
1. **Prepare the Data**
   TFRecords
2. **Build the Keras Model**
3. **Train the Model**
4. **Evaluate the Model**
   DICE, TensorBoard


THE LAB

Dataset

- Cardiac MRI short-axis (SAX) scans
  - Sunnybrook cardiac images from earlier competition
    http://smial.sri.utoronto.ca/LV_Challenge/Data.html
  - "Sunnybrook Cardiac MR Database" is made available under the CC0 1.0 Universal license described above, and with more detail here:
    http://creativecommons.org/publicdomain/zero/1.0/
  - Attribution:
    - Radau P, Lu Y, Connelly K, Paul G, Dick AJ, Wright GA. "Evaluation Framework for Algorithms Segmenting Short Axis Cardiac MRI." The MIDAS Journal -Cardiac MR Left Ventricle Segmentation Challenge, http://hdl.handle.net/10380/3070

Image Example



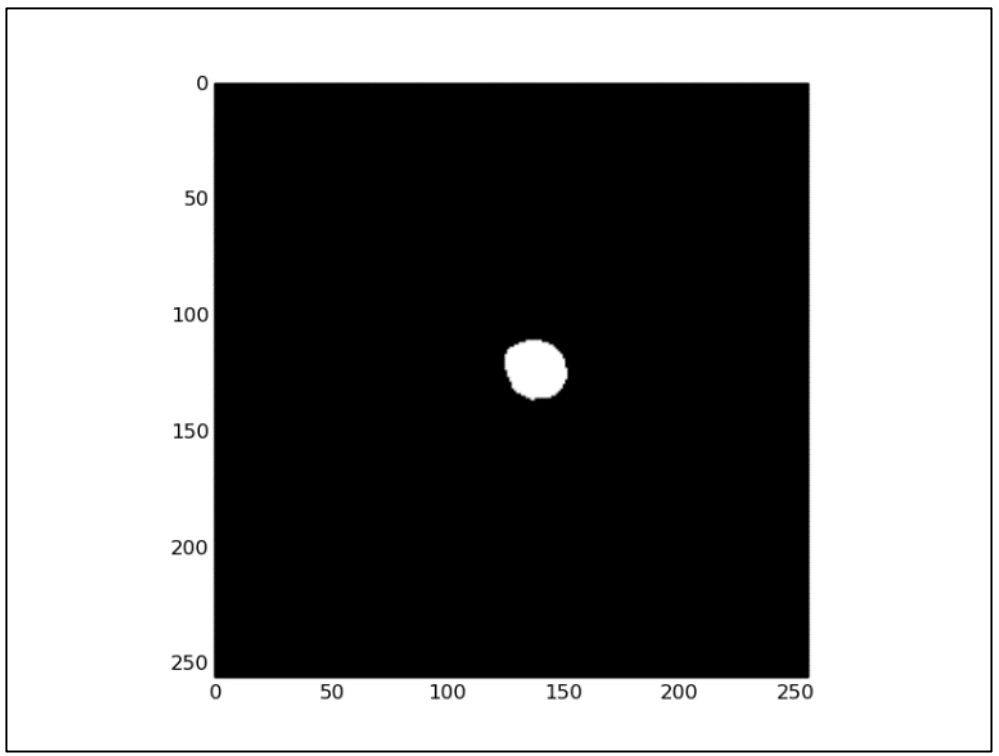
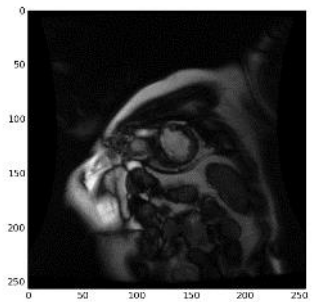
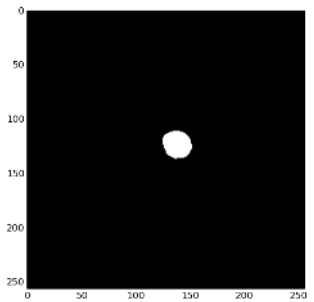
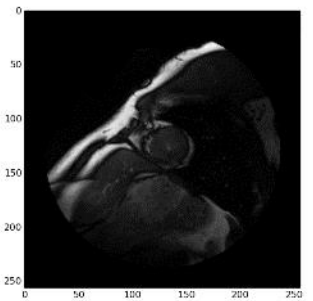
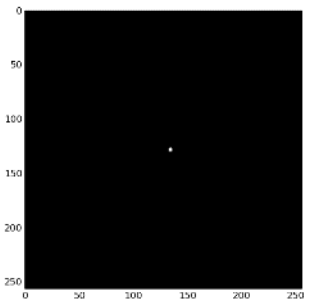
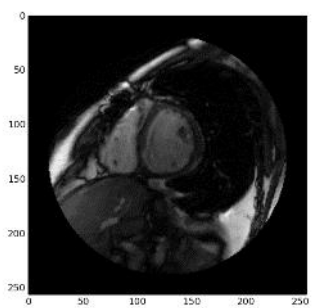
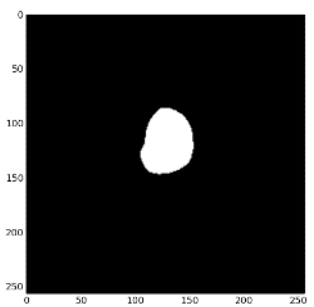
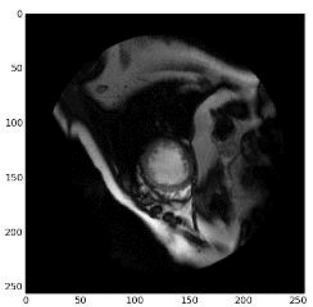
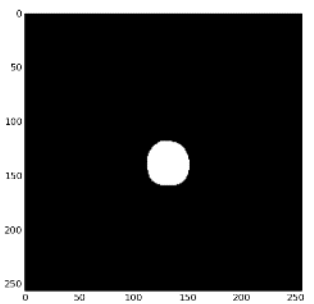


Image Examples

Complete images and expertly labeled contours of LV

Data Details

- Original images are 256 x 256 grayscale DICOM format
- Output is a tensor of size 256 x 256 x 2
  - Each pixel belongs to one of two classes
- Training set consist of 234 images
- Validation set consist of 26 images

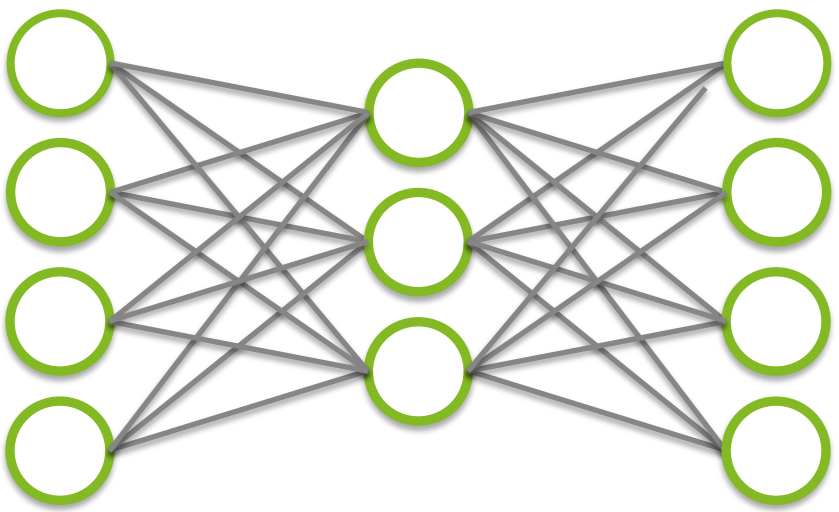
Background Data Setup

- Lots of guidance and code for how to setup/extract data taken from here:
  - https://www.kaggle.com/c/second-annual-data-science-bowl/details/deep-learning-tutorial
- Images and contours have been extracted from the raw data and packaged up for ingest into TensorFlow
- TensorFlow data records provided but raw data is NOT provided for this lab
  - If interested you can download yourself

Lab Tasks

TASK 1

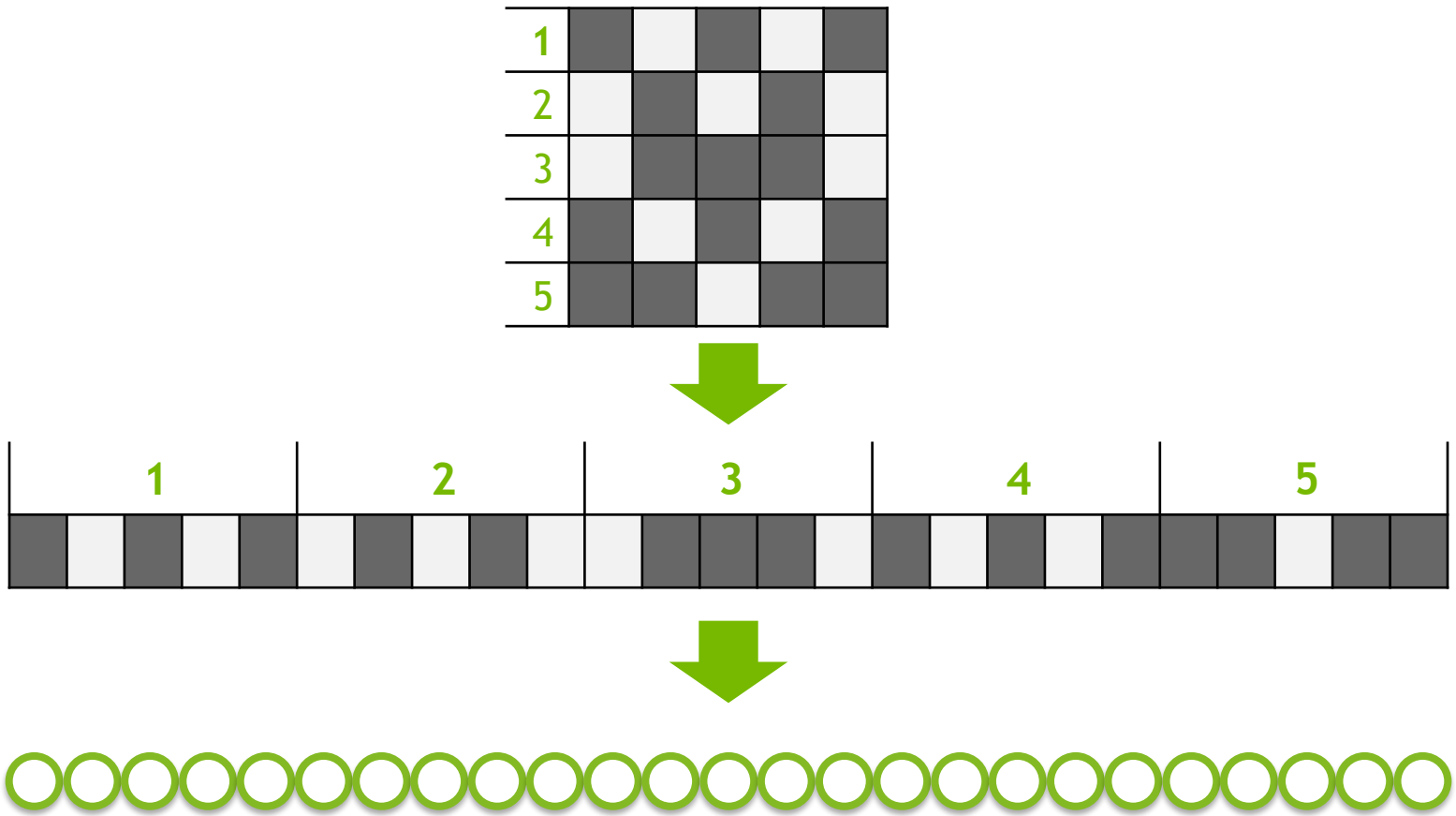
Ensure things are working properly!



- Train and test a fully-connected neural network with one hidden layer
- For the loss computation we'll use TF built-in sparse_softmax_cross_entropy_with_logits
  - Computes softmax of the inference output then cross entropy against the correct labels

TASK 1

Ensure things are working properly!



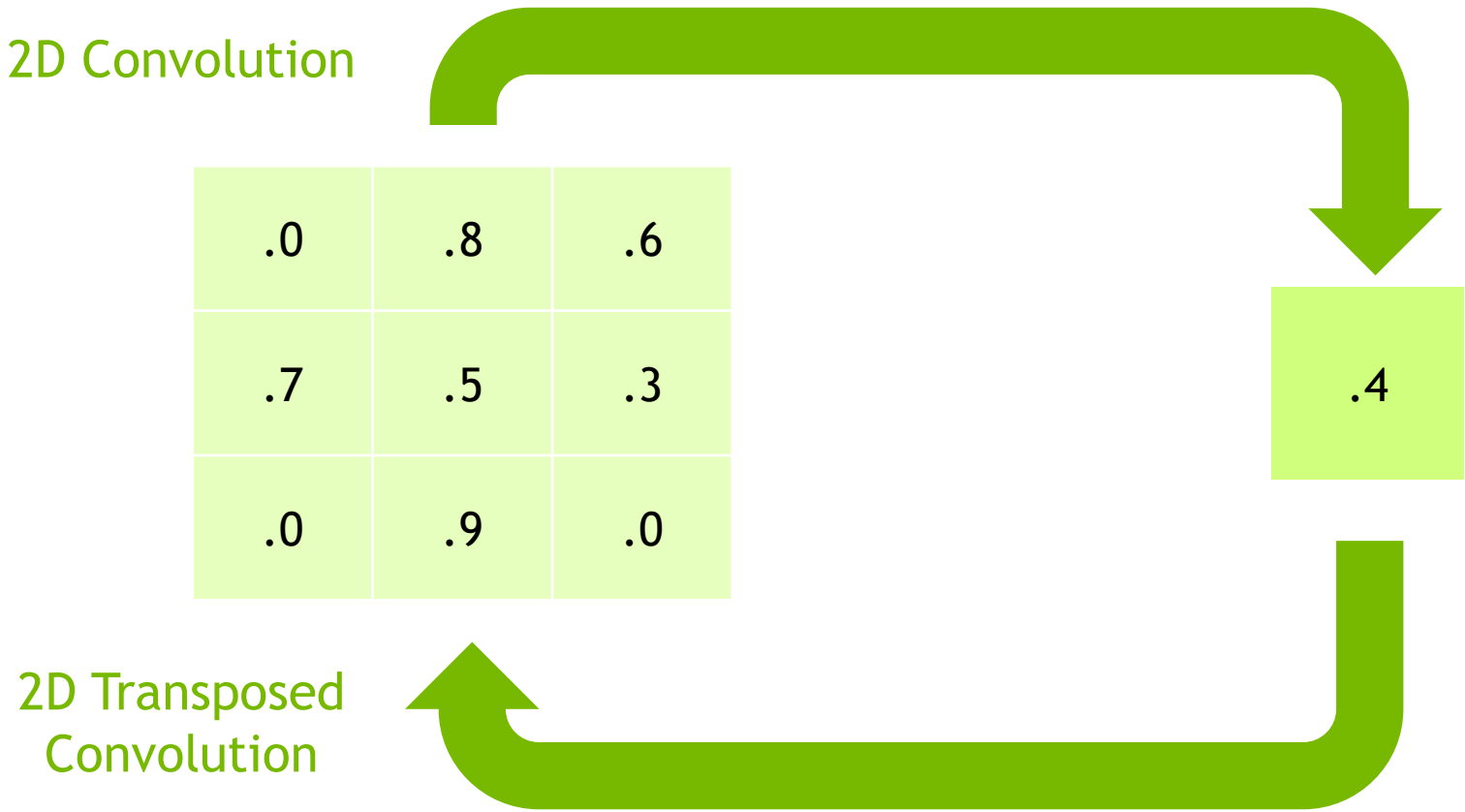
TASK 2

Convolutional Layers

- Convolution layers
  - Previous example focused on each input pixel
  - What if features encompass multiple input pixels
  - Can use convolutions to capture larger receptive fields
- Pooling layers
  - Essentially a down-sampling method retaining information while eliminating some computational complexity

TASK 2

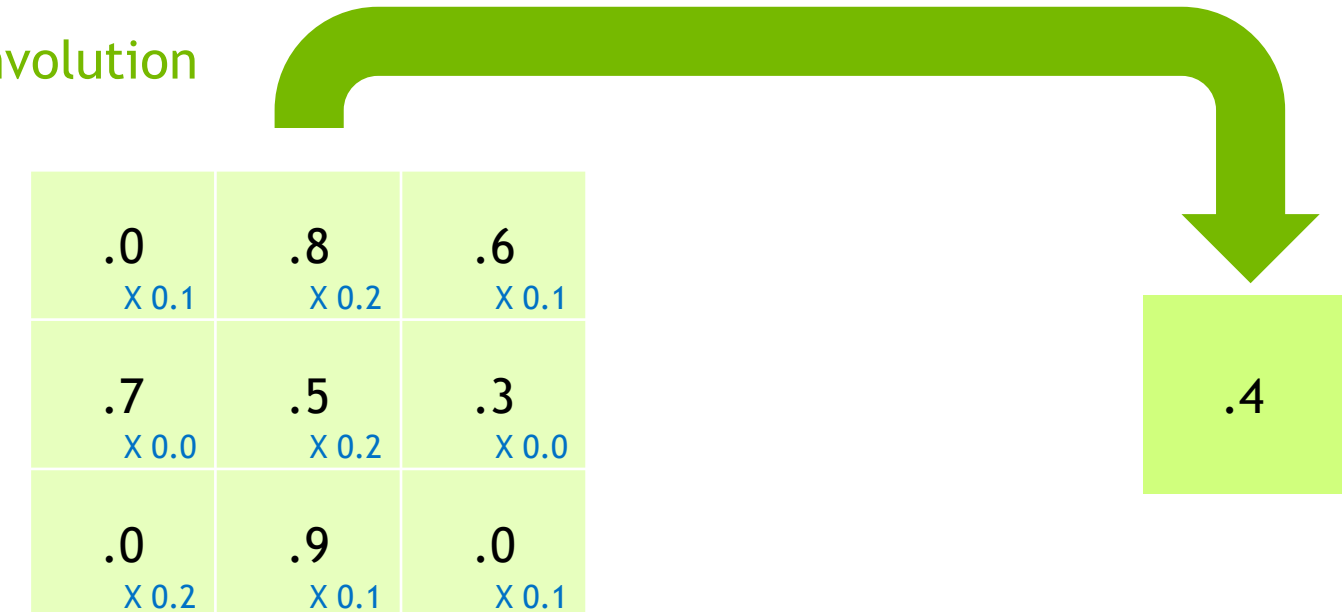
Convolutional Layers



TASK 2

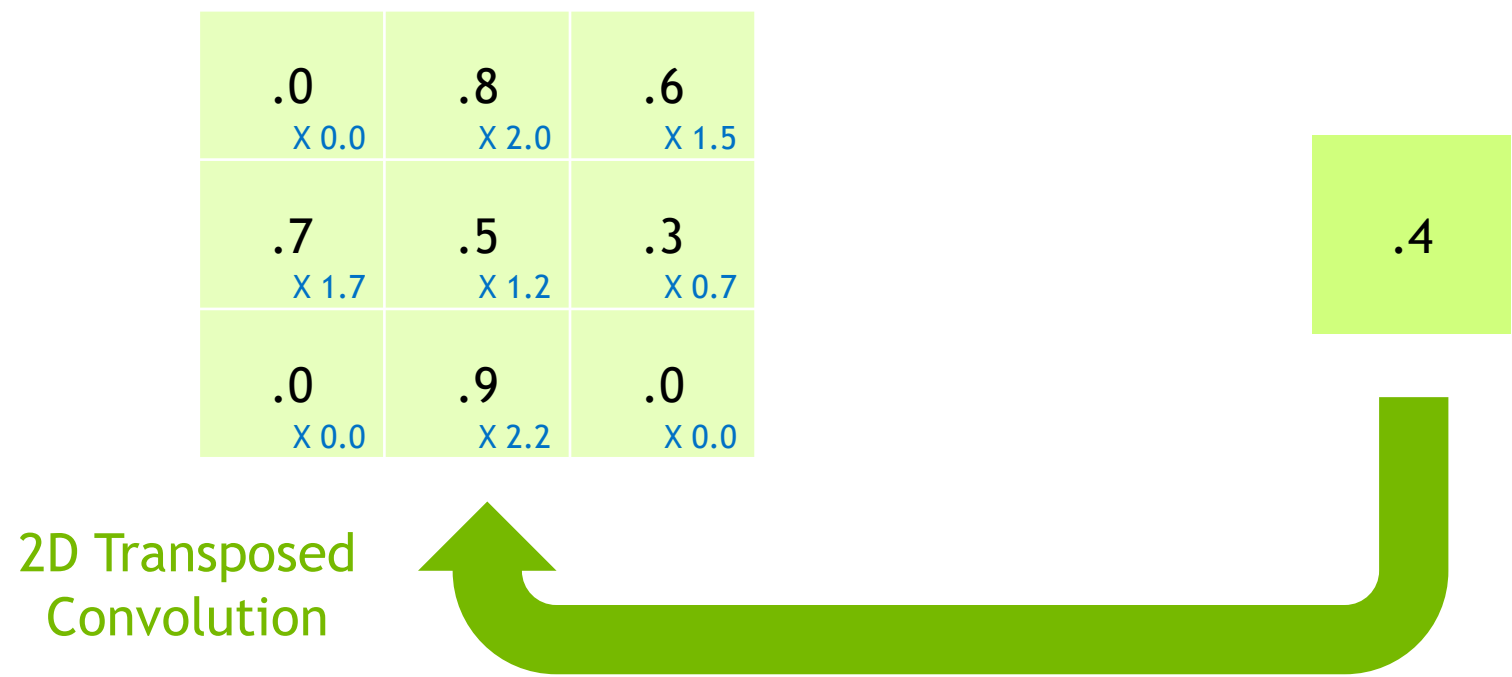
Convolutional Layers

2D Convolution



TASK 2

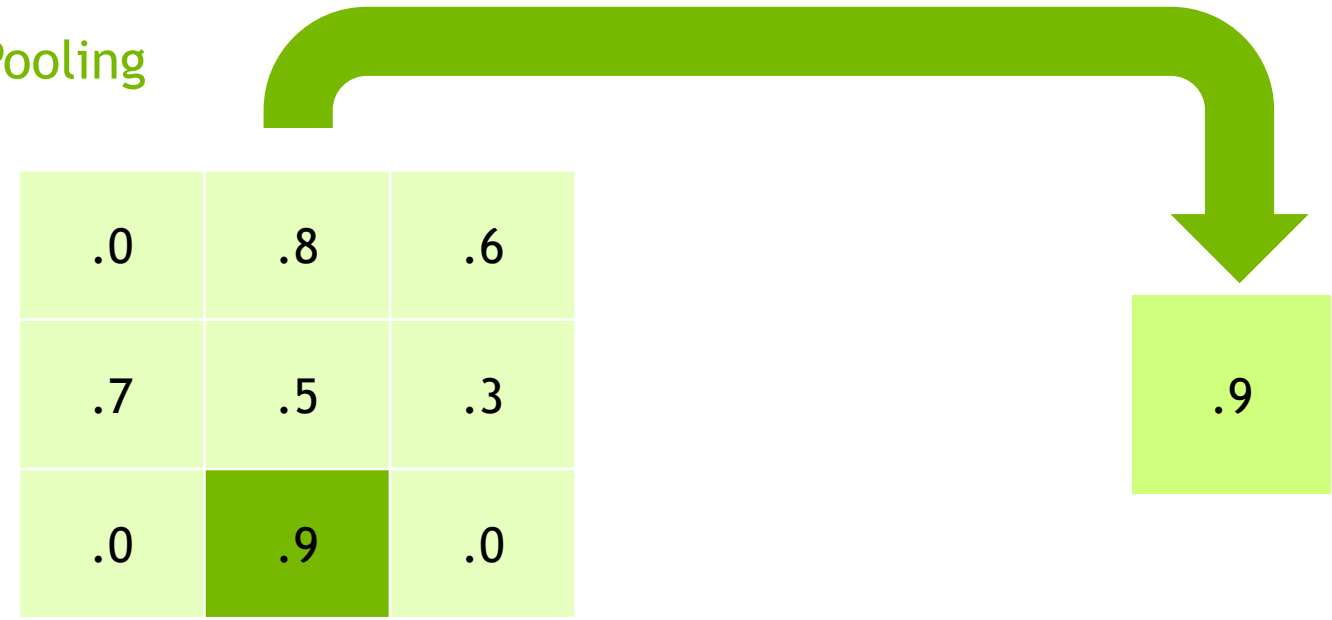
Convolutional Layers



TASK 2

Convolutional Layers

Max Pooling



TASK 2

Convolutional Layers

- Finish the CNN, replace “FIXME”
  - You need to figure out the dimensions
- Convolution1, 5x5 kernel, stride 2; Maxpooling1, 2x2 window, stride 2
- Convolution2, 5x5 kernel, stride 2; Maxpooling2, 2x2 window, stride 2
- Convolution3, 3x3 kernel, stride 1; Convolution4, 3x3 kernel, stride 1
- Score_classes, 1x1 kernel, stride 1; Upscore (DeConv), 31x31 kernel, stride 16
- Optional / Time Permitting: Experiment with num_epochs

TASK 3

The DICE Metric

- Metric to compare the similarity of two samples:

$$\frac{2A_{nl}}{A_n + A_l}$$

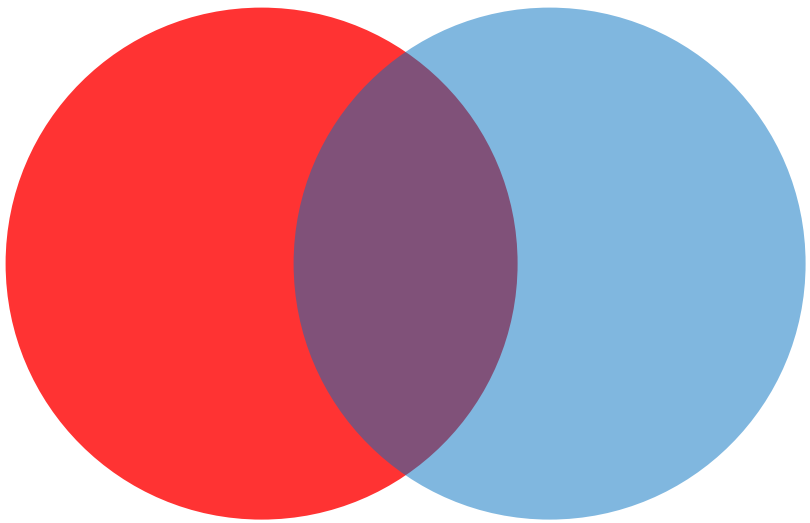
  - Where:
    - A_n is the area of the contour predicted by the network
    - A_l is the area of the contour from the label
    - A_{nl} is the intersection of the two
      - The area of the contour that is predicted correctly by the network
      - 1.0 means perfect score.

- More accurately compute how well we're predicting the contour against the label

- We can just count pixels to give us the respective areas

TASK 3

The DICE Metric



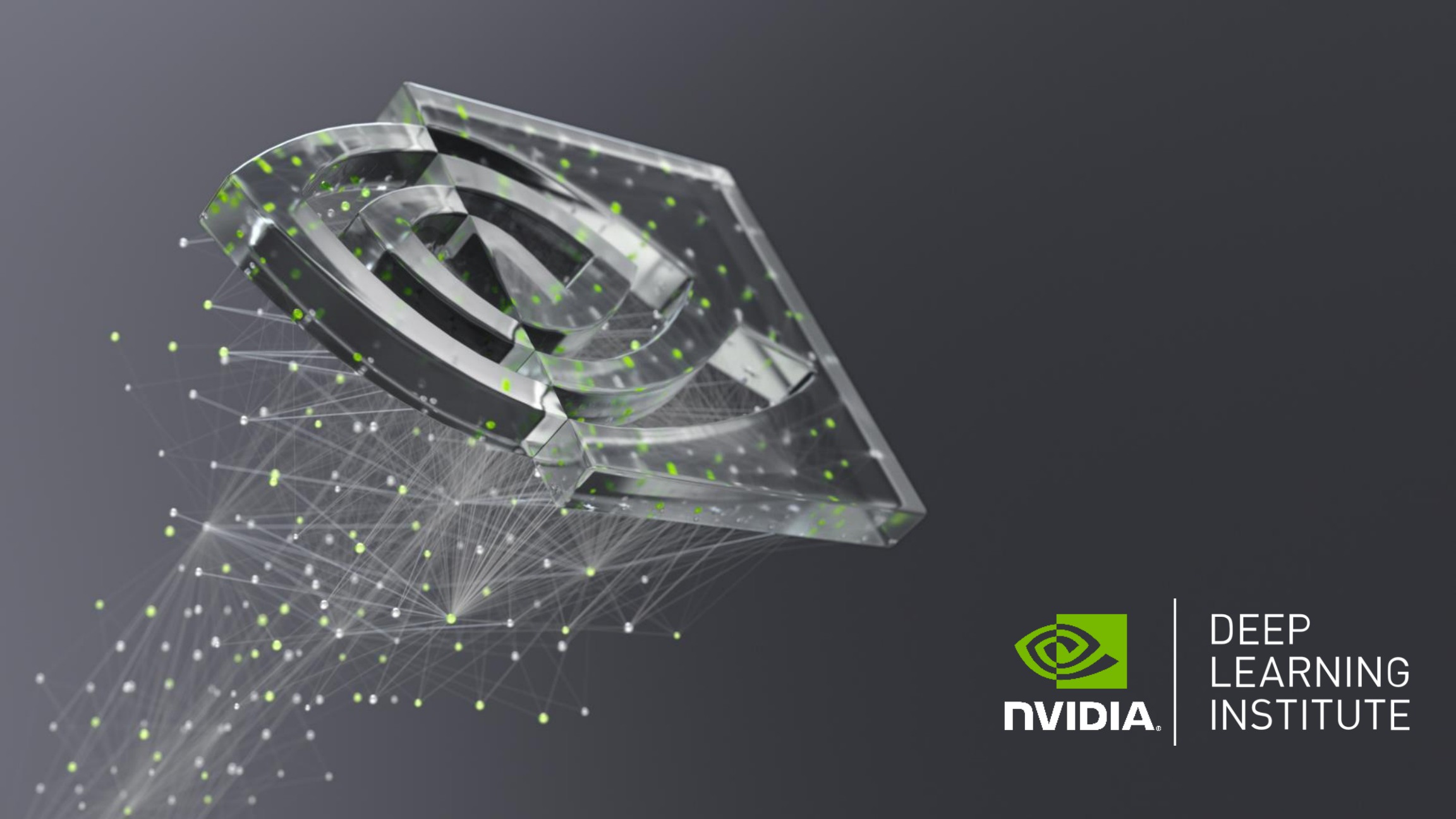
$$\frac{2 \times \text{The Purple Area}}{\text{Area of the Red Circle} + \text{Area of the Blue Circle}}$$

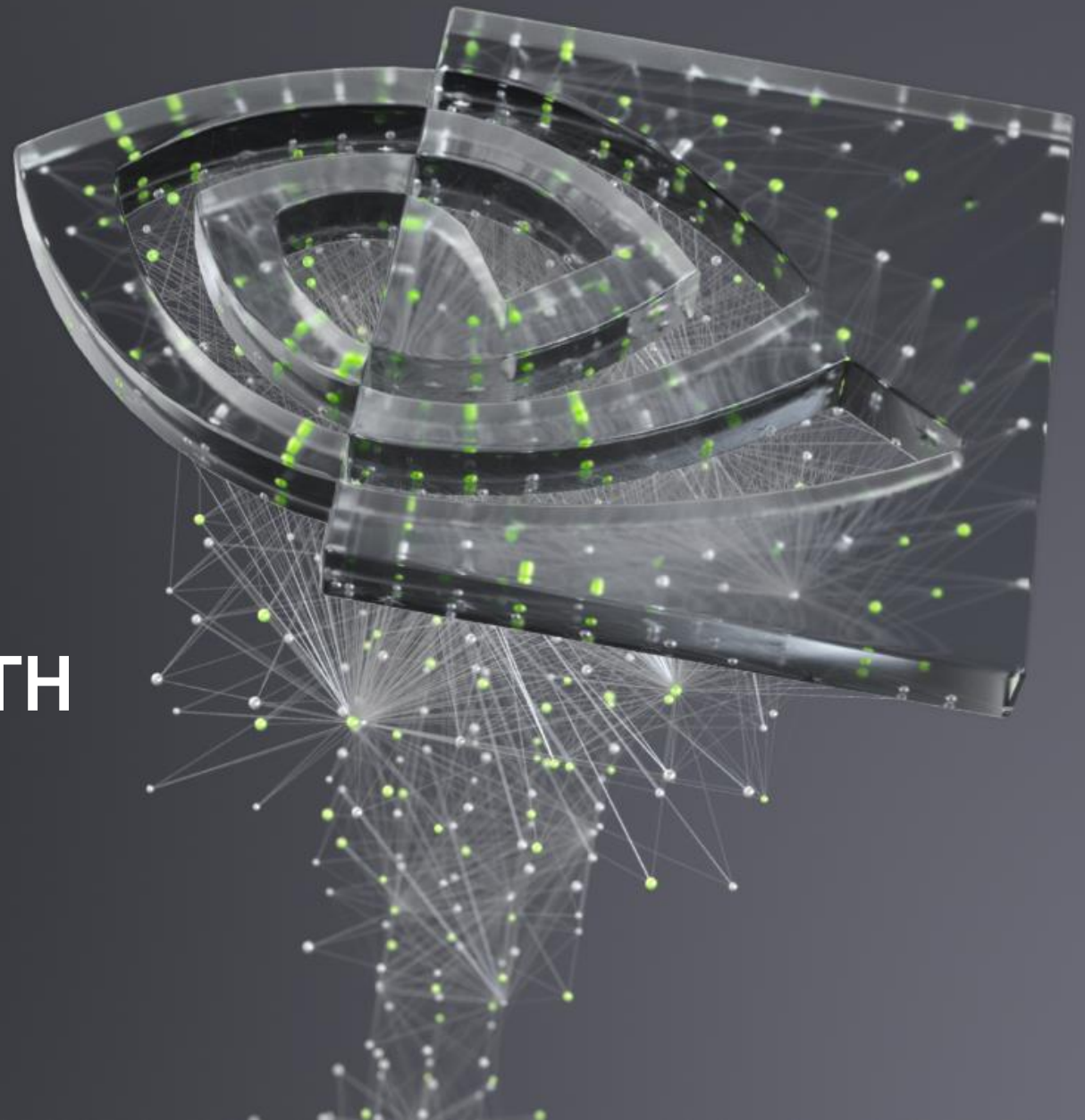
TASK 3

Parameter Search

- learning_rate: initial learning rate
- decay_rate: the rate that the initial learning rate decays
  - e.g., 1.0 is no decay, 0.5 means cut the decay rate in half each number of (decay) steps
- decay_steps: number of steps to execute before changing learning rate
- num_epochs: number of times to cycle through the input data
- batch_size: keep at 1 for now
- Experiment with **learning_rate, decay_rate, decay_steps, num_epoch**
- Record the parameters that give you the best Dice score

Start the Lab!


NVIDIA
DEEP
LEARNING
INSTITUTE



NVIDIA | DEEP LEARNING INSTITUTE

GETTING STARTED WITH IMAGE SEGMENTATION SOLUTIONS

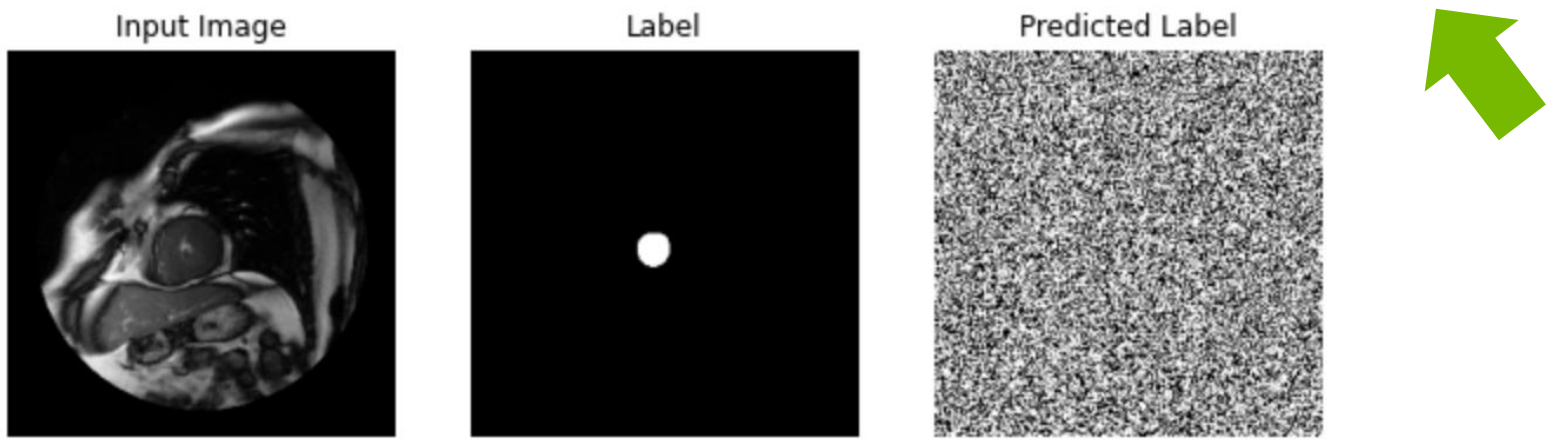
Task 1 - Solution

```
# set up the model architecture
model = tf.keras.models.Sequential([
    Flatten(input_shape=[256, 256, 1]),
    Dense(64, activation='relu'),
    Dense(256*256*2, activation='softmax'),
    Reshape((256, 256, 2))
])

# specify how to train the model with algorithm, the loss function and metrics
model.compile(
    optimizer='adam',
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
    metrics=['accuracy'])
```

Task 1 Training Output – 1 Epoch

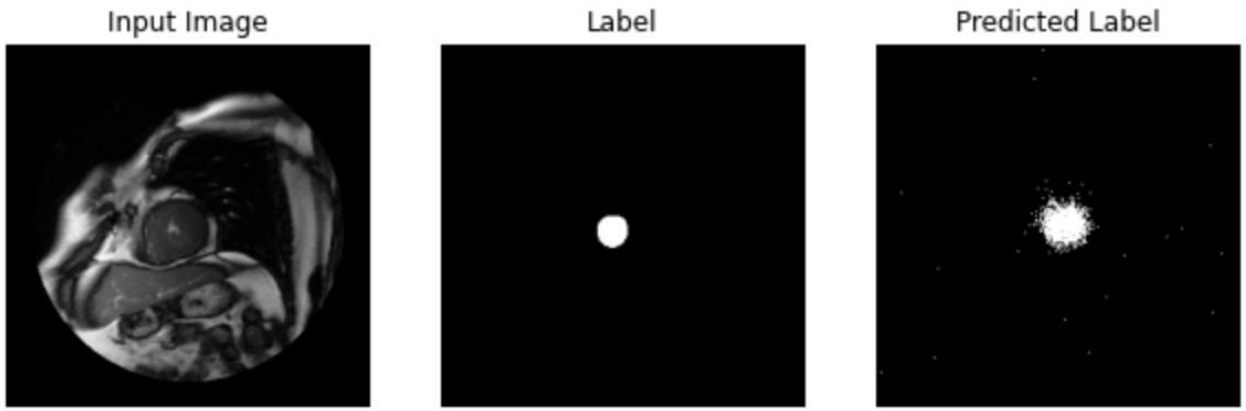
```
Train for 234 steps, validate for 26 steps
Epoch 1/20
WARNING:tensorflow:Method (on_train_batch_end) is slow compared to the batch update (0.270887). Check your
callbacks.
  1/234 [..............................] - ETA: 2:12 - loss: 0.6931 - accuracy: 0.5012WARNING:tensorflow:M
ethod (on_train_batch_end) is slow compared to the batch update (0.191189). Check your callbacks.
  2/234 [..............................] - ETA: 1:19 - loss: 0.6931 - accuracy: 0.4996WARNING:tensorflow:M
ethod (on_train_batch_end) is slow compared to the batch update (0.111491). Check your callbacks.
220/234 [===========================>..] - ETA: 0s - loss: 0.6931 - accuracy: 0.5048
```



Sample Prediction after epoch 1

Task 1 Training Output – 20 Epochs

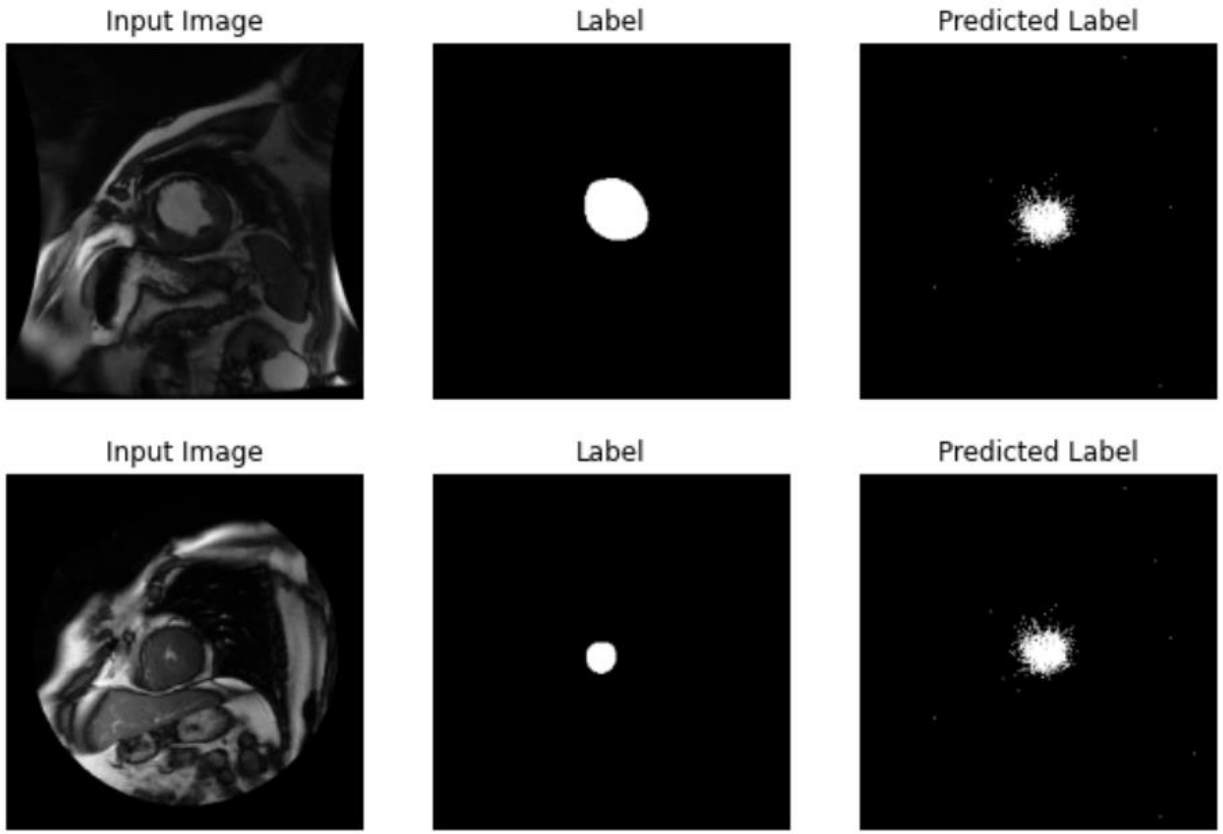
```
Epoch 20/20
226/234 [============================>..] - ETA: 0s - loss: 0.6931 - accuracy: 0.9864
```




```
Sample Prediction after epoch 20

234/234 [==============================] - 2s 9ms/step - loss: 0.6931 - accuracy: 0.9865 - val_loss: 0.6931 - val_accu
racy: 0.9865
```

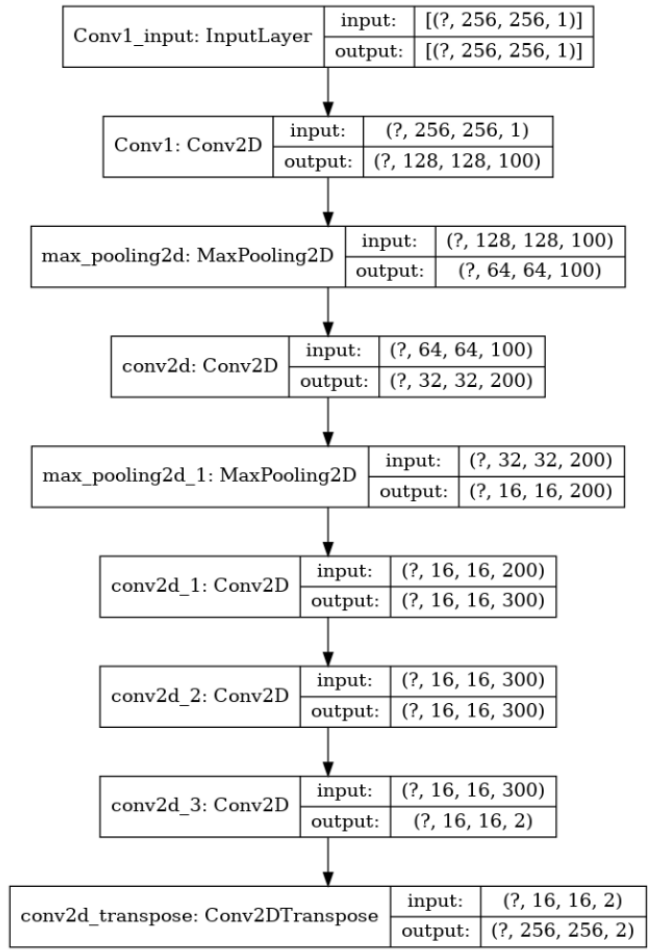
Task 1 Training Output – Good Result?



Task 2 - Solution

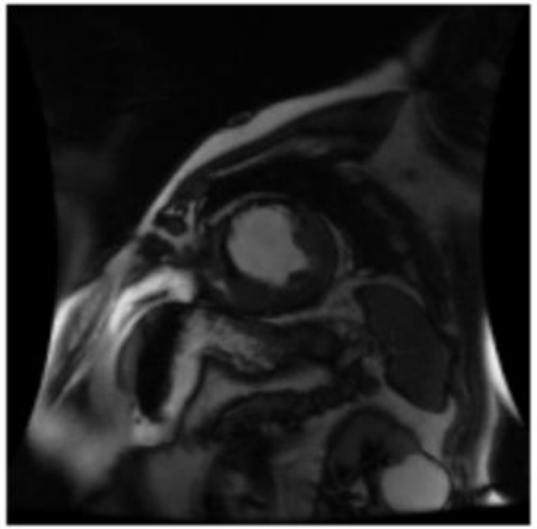
```
layers = [
    Conv2D(input_shape=[256, 256, 1],
           filters=100,
           kernel_size=5,
           strides=2,
           padding="same",
           activation=tf.nn.relu,
           name="Conv1"),
    MaxPool2D(pool_size=2, strides=2, padding="same"),
    Conv2D(filters=200,
           kernel_size=5,
           strides=2,
           padding="same",
           activation=tf.nn.relu),
    MaxPool2D(pool_size=2, strides=2, padding="same"),
    Conv2D(filters=300,
           kernel_size=3,
           strides=1
```

… and repeat …

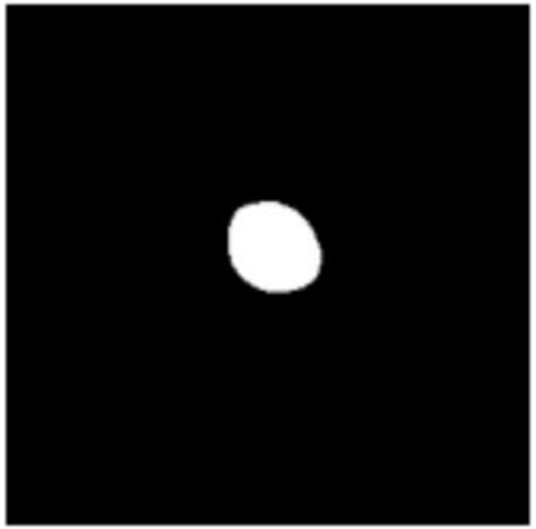


Task 2 Training Output – No Training

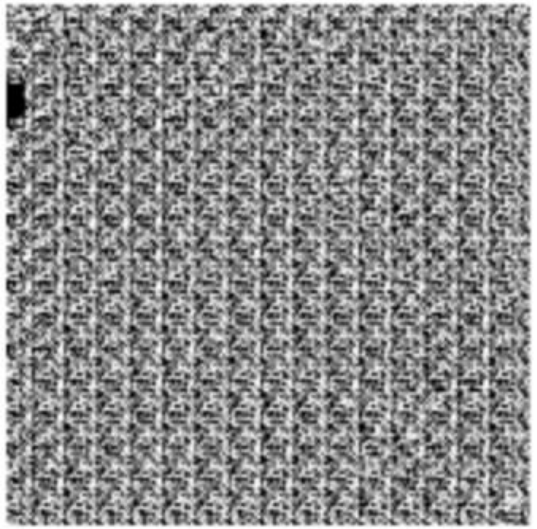
Input Image



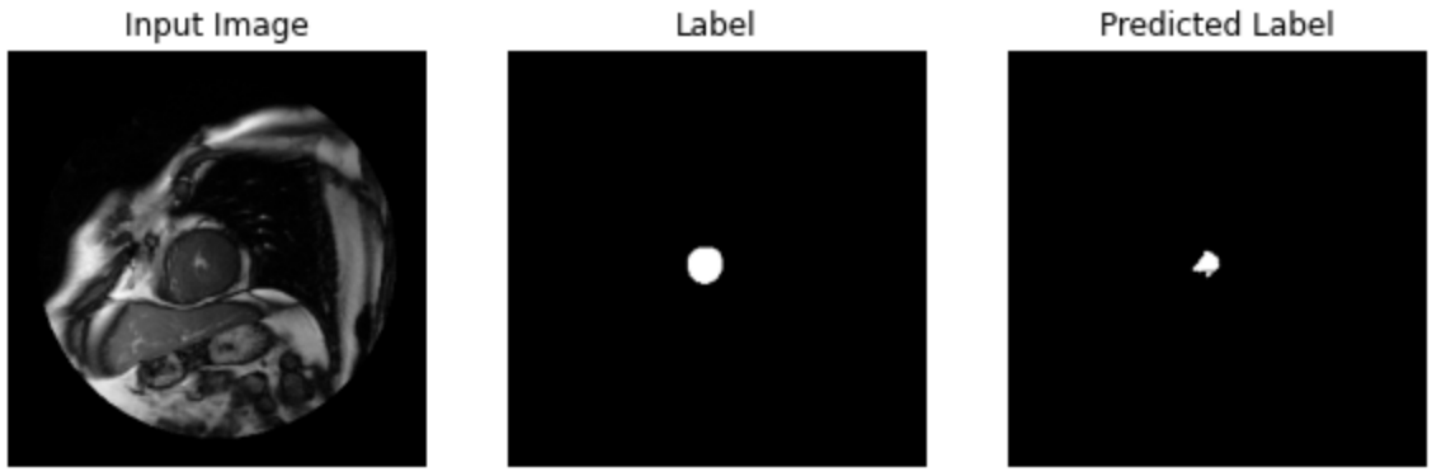
Label



Predicted Label



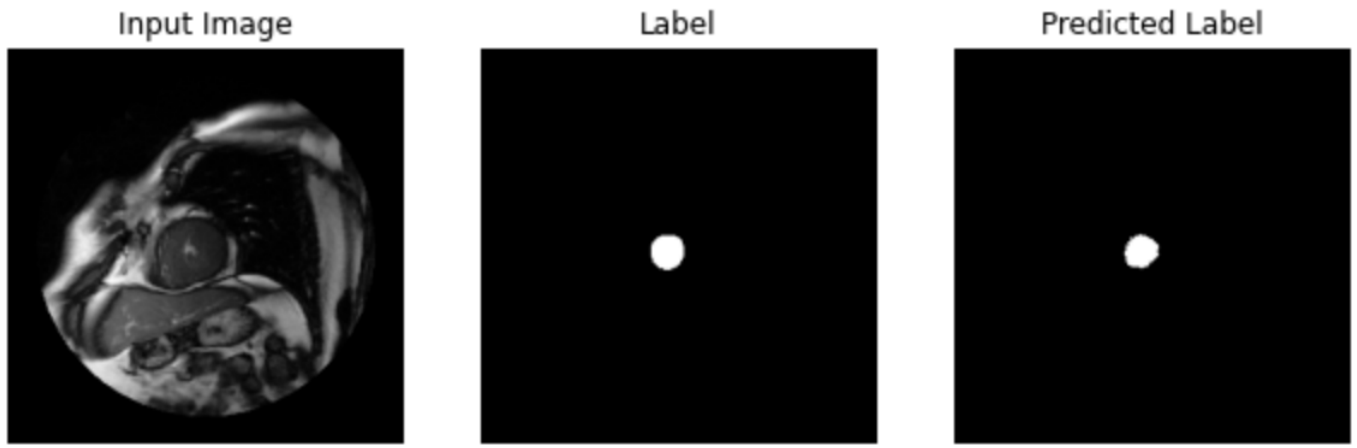
Task 2 Training Output – 20 Epochs




```
Sample Prediction after epoch 20

234/234 [==============================] - 2s 7ms/step - loss: 0.0040 - accuracy: 0.9983 - val_loss: 0.0100 -
val_accuracy: 0.9969
```

Task 3 – 30 Epochs

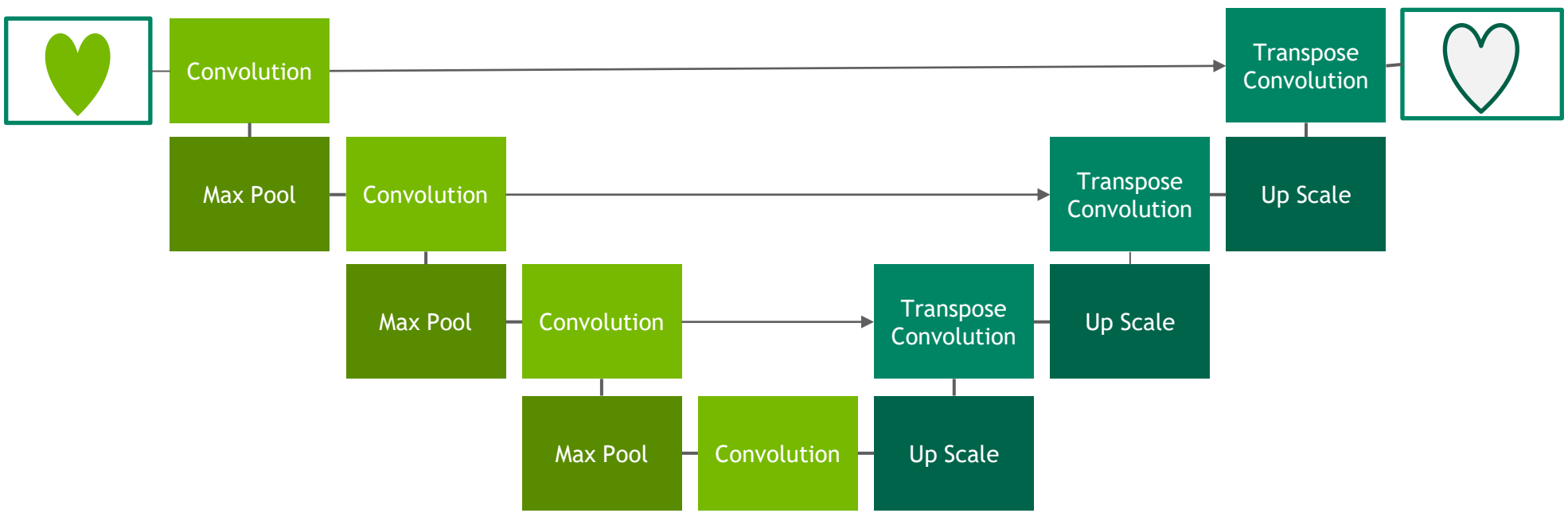



```
Sample Prediction after epoch 30

234/234 [==============================] - 1s 6ms/step - loss: 0.0019 - dice_coef: 0.9583 - accuracy: 0.9992
- val_loss: 0.0104 - val_dice_coef: 0.8519 - val_accuracy: 0.9975
```

Task 3 Parameter Search - Solution

You tell us!



U-Net

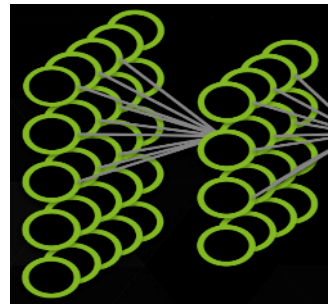
NVIDIA DEEP LEARNING INSTITUTE

Online self-paced labs and instructor-led workshops on deep learning and accelerated computing

Take self-paced labs at **www.nvidia.com/dlilabs**

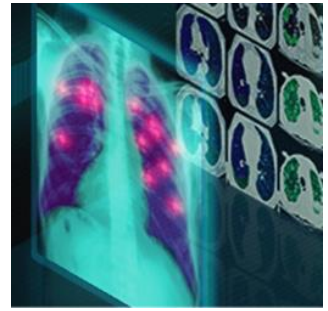
View upcoming workshops and request a workshop onsite at **www.nvidia.com/dli**

Educators can join the University Ambassador Program to teach DLI courses on campus and access resources. Learn more at **www.nvidia.com/dli**

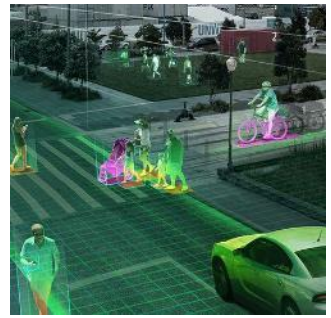


Fundamentals

Autonomous Vehicles



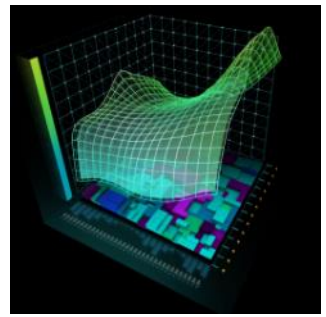
Healthcare



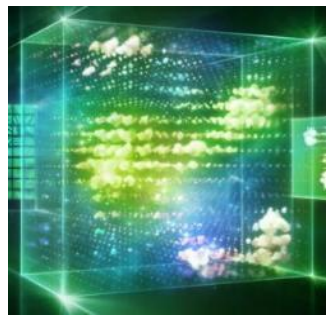
Intelligent Video Analytics

Robotics

Game Development & Digital Content

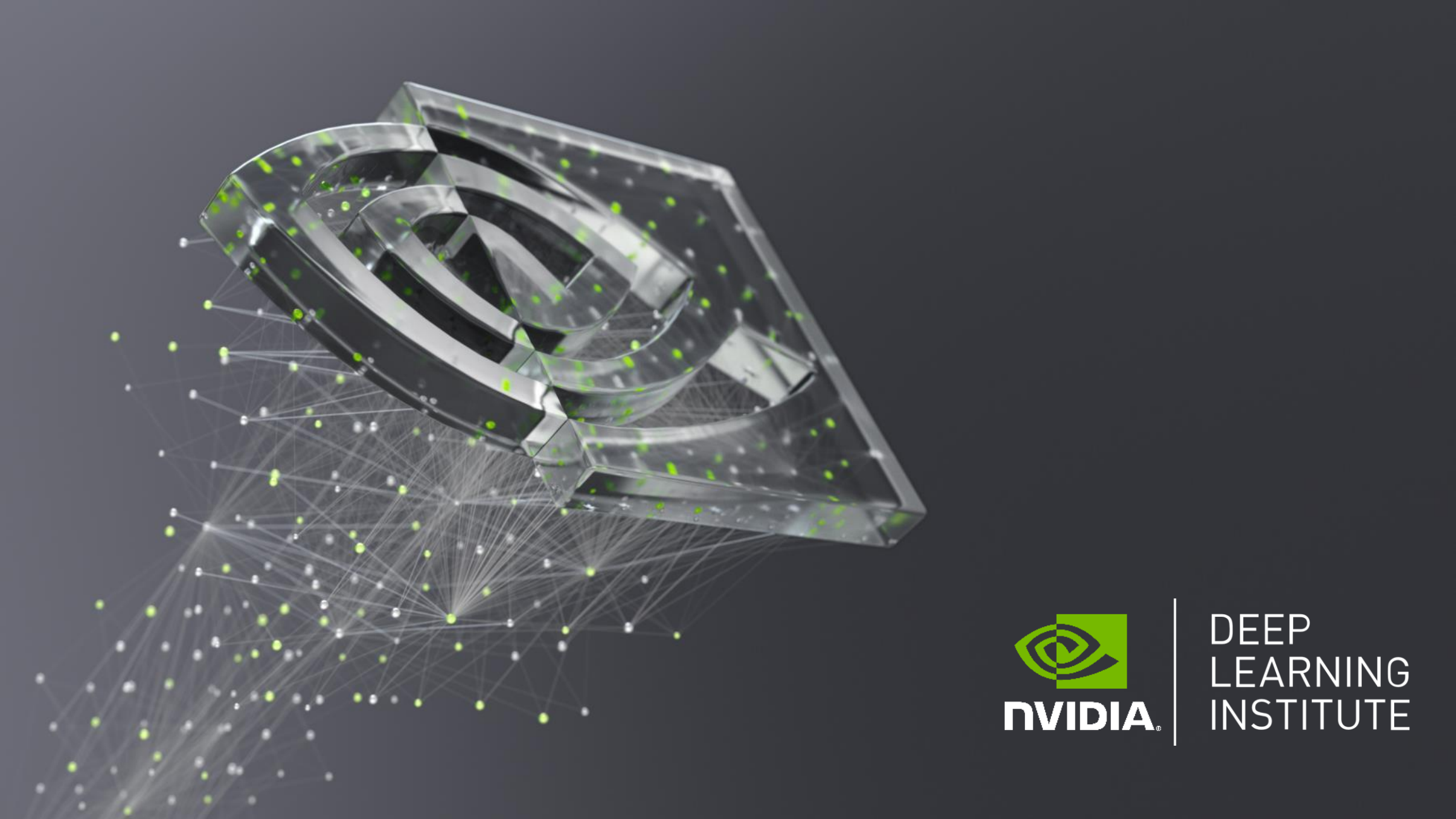


Finance



Accelerated Computing

Virtual Reality


NVIDIA
DEEP
LEARNING
INSTITUTE