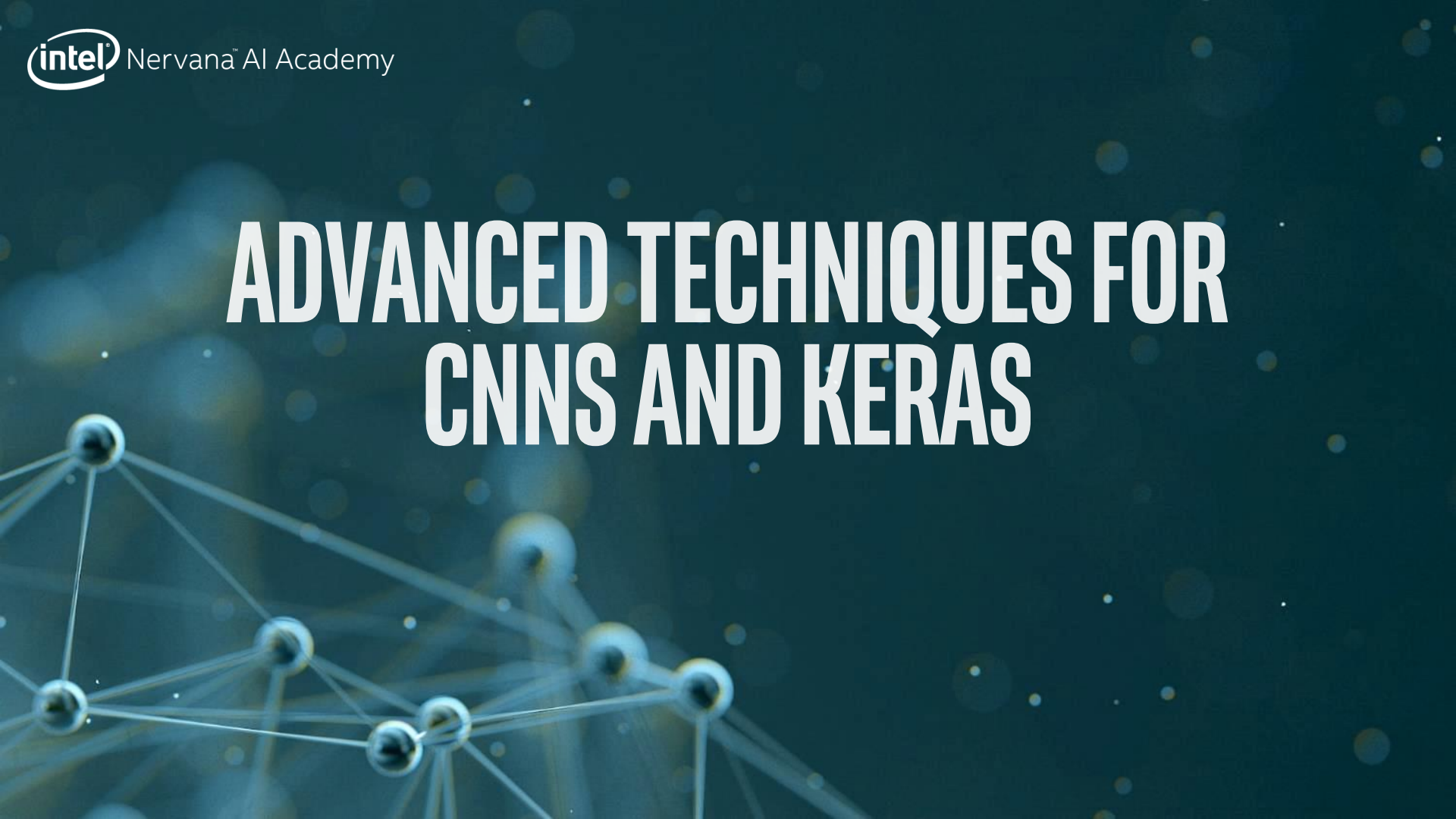

intel Nervana AI Academy
ADVANCED TECHNIQUES FOR CNNS AND KERAS

DATA AUGMENTATION

- One practical obstacle to building image classifiers is obtaining labeled training data.
- Finding images is difficult.
- Labeling images is time consuming and costly.
- How can me make the most out of the labelled data we have?

DATA AUGMENTATION

If this is a chair:

DATA AUGMENTATION

If this is a chair...

Then so is this!

DATA AUGMENTATION

If this is a chair...

Also this:

DATA AUGMENTATION

If this is a chair...

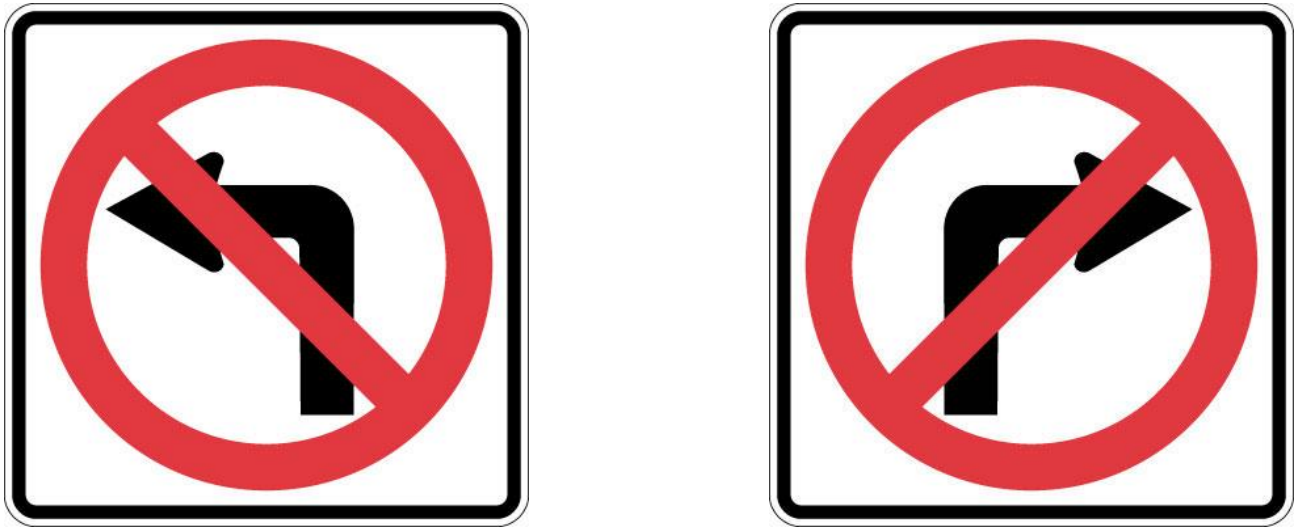
Also this:

DATA AUGMENTATION

- By slightly altering images, we can increase our effective data size.
- Also allows the neural network to learn invariance to certain transformations.
- But we need to be careful—this can have unintended consequences.

DATA AUGMENTATION

Would not want a self-driving car to think these mean the same thing!



DATA FLOWS IN KERAS

- Keras has a convenient mechanism for Data Augmentation.
- Requires use of “Data Generators”
- To date, we have used the standard `model.fit` mechanism
- Holds entire dataset in memory
- Reads the batches one by one out of memory

DATA FLOWS IN KERAS

- Alternative mechanism is to use a “data generator”
- Idea: define a generator object which “serves” the batches of data.
- Then use `model.fit_generator` instead of `model.fit`
- Generators can be used to serve images from disk to conserve working memory

IMAGEDATAGENERATOR

- Keras has an `ImageDataGenerator` class which permits “real-time” data-augmentation.
- When a batch of images is served, you can specify random changes to be made to the image.
- These include shifting, rotating, flipping, and various normalizations of the pixel values.

IMAGEDATAGENERATOR

```
keras.preprocessing.image.ImageDataGenerator(
featurewise_center=False, samplewise_center=False,
featurewise_std_normalization=False, samplewise_std_normalization=False,
zca_whitening=False,
rotation_range=0.,
width_shift_range=0.,
height_shift_range=0.,
shear_range=0., zoom_range=0., channel_shift_range=0., fill_mode='nearest',
cval=0.,
horizontal_flip=False, vertical_flip=False,
 rescale=None, preprocessing_function=None,
data_format=K.image_data_format())
```

Lots of options! We'll discuss a few.

SHIFTING IMAGES

```
keras.preprocessing.image.ImageDataGenerator(
width_shift_range=0.,
height_shift_range=0.,
...)
```

- These determine the range of possible horizontal or vertical shifts to make to the image.
- Measured as a percentage of the image size.
- So if an image is 200 x 200, and `width_shift_range=0.1`, then it will shift up to 20 pixels to the left or right.

SHIFTING IMAGES (HOW TO FILL IN)

```
keras.preprocessing.image.ImageDataGenerator(
...,
fill_mode='nearest', cval=0.,
...)
```

- When shifting, we don't wish to change the proportions of the image.
- We need to "fill in" the pixels on the other side.
- Options are "`constant`", "`nearest`", "`reflect`", "`wrap`"
- The cval is the value when "constant" is specified.

ROTATING IMAGES

```
keras.preprocessing.image.ImageDataGenerator(
...,
rotation_range=0.,
...)
```

- This allows us to specify a range of possible rotations
- Measured in degrees
- So rotation_range=30 means up to a 30 degree rotation (in either direction)

FLIPPING IMAGES

```
keras.preprocessing.image.ImageDataGenerator(
...,
horizontal_flip=False, vertical_flip=False,
...)
```

Whether or not to randomly flip in a horizontal or vertical direction.

KERAS FUNCTIONAL API

- So far we have primarily used the Keras Sequential method.
- Very convenient when each additional layer takes the results of the previous layer.
- However, suppose we have more than one input to a particular layer.
- Suppose we have multiple outputs, and want different loss functions for these outputs.
- More complicated graph structures require the Functional API.

KERAS FUNCTIONAL API—PRINCIPLES

- Every layer is “callable” on a tensor, and returns a tensor.
- Specifically designate inputs using the Input layer.
- Merge outputs into a single output using `keras.layers.concatentate`
- Stack layers in a similar fashion to the Sequential model.
- Use Model to specify inputs and outputs of your model.
- When compiling, can specify different losses for different outputs, and specify how they should be weighted.

intel
Nervana™ AI Academy