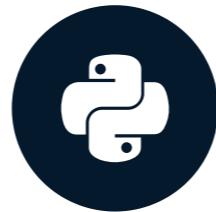


Tracking learning

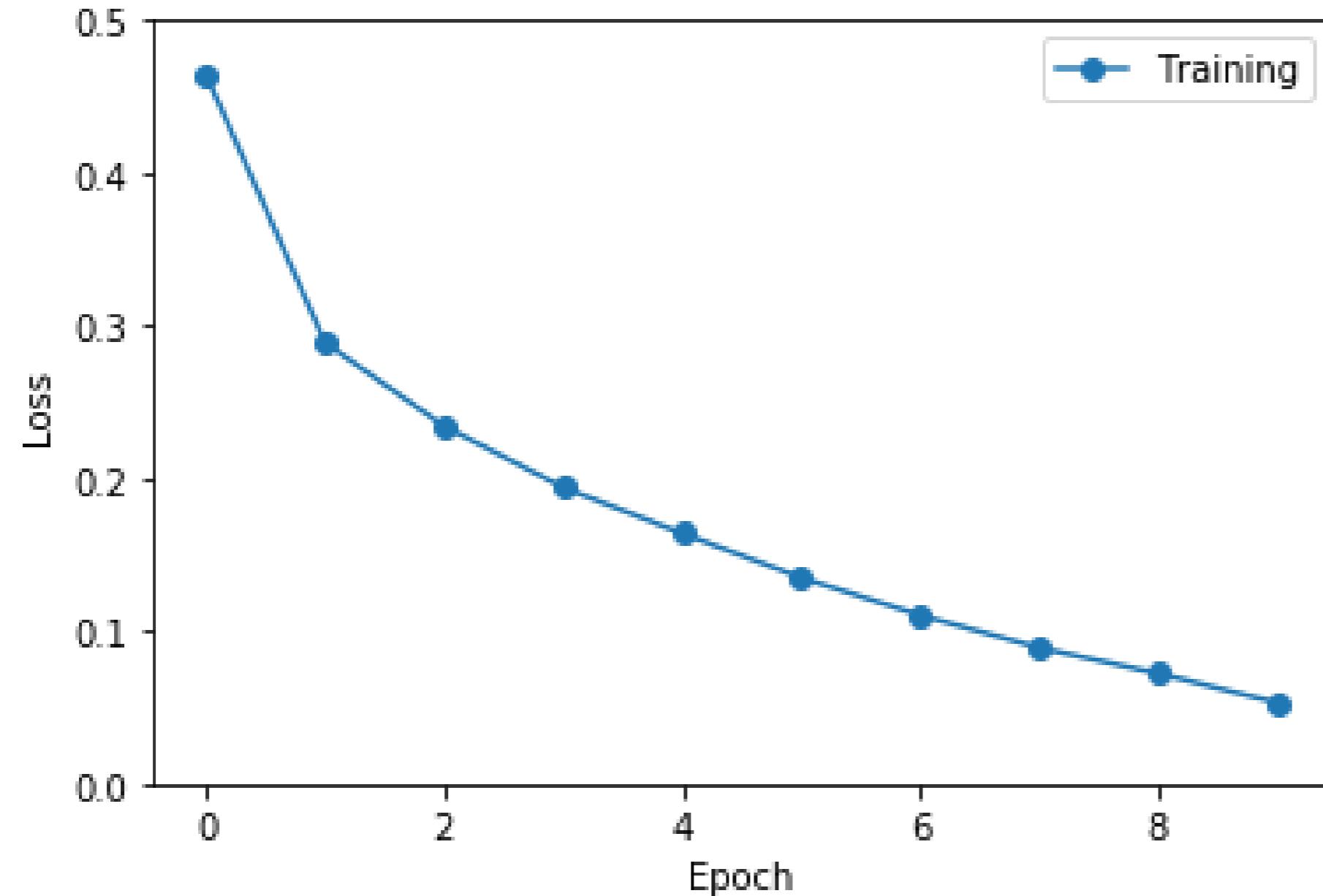
IMAGE PROCESSING WITH KERAS IN PYTHON



Ariel Rokem

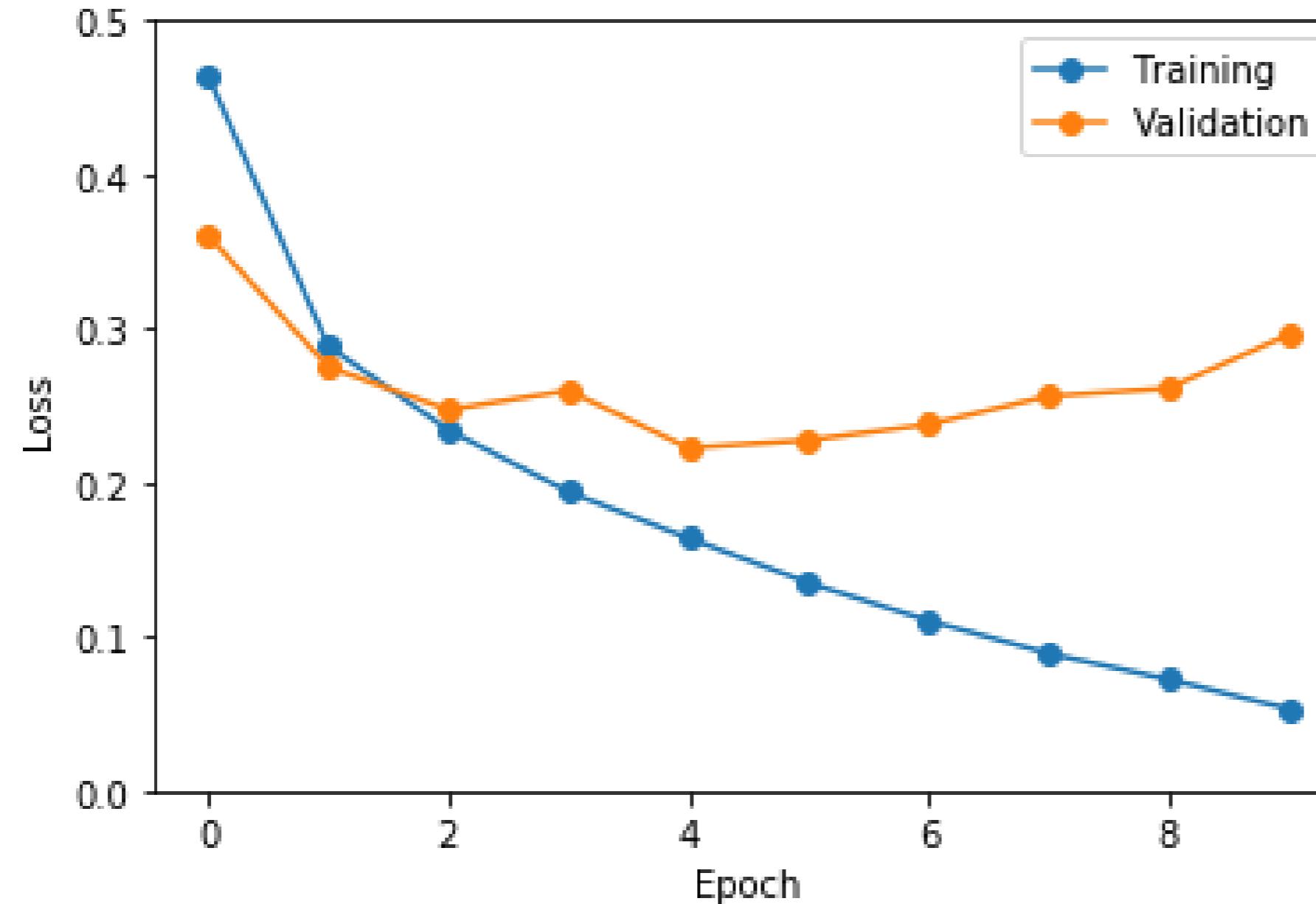
Senior Data Scientist, University of
Washington

Learning curves: training



loss

Learning curves: validation



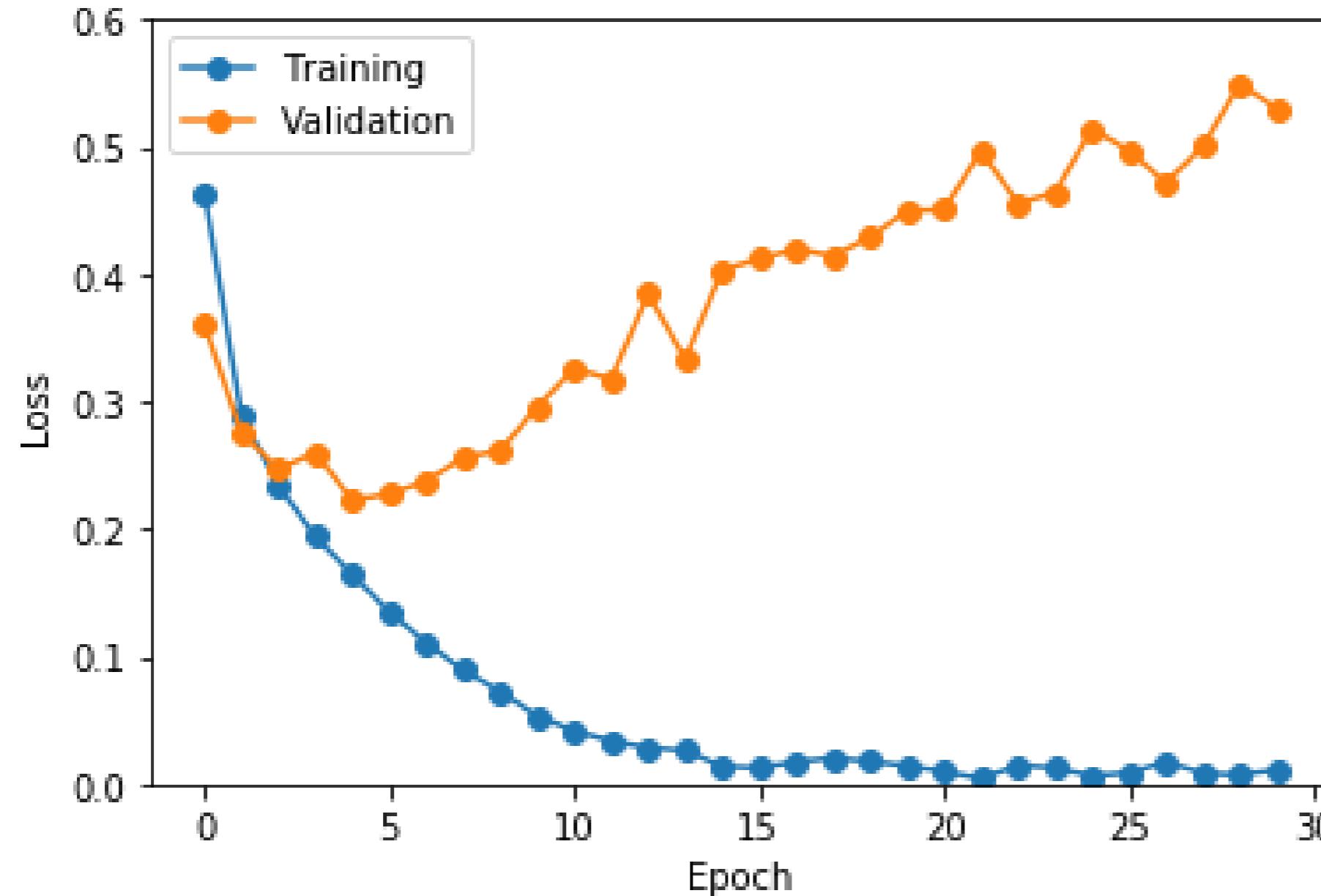
, validation loss

,

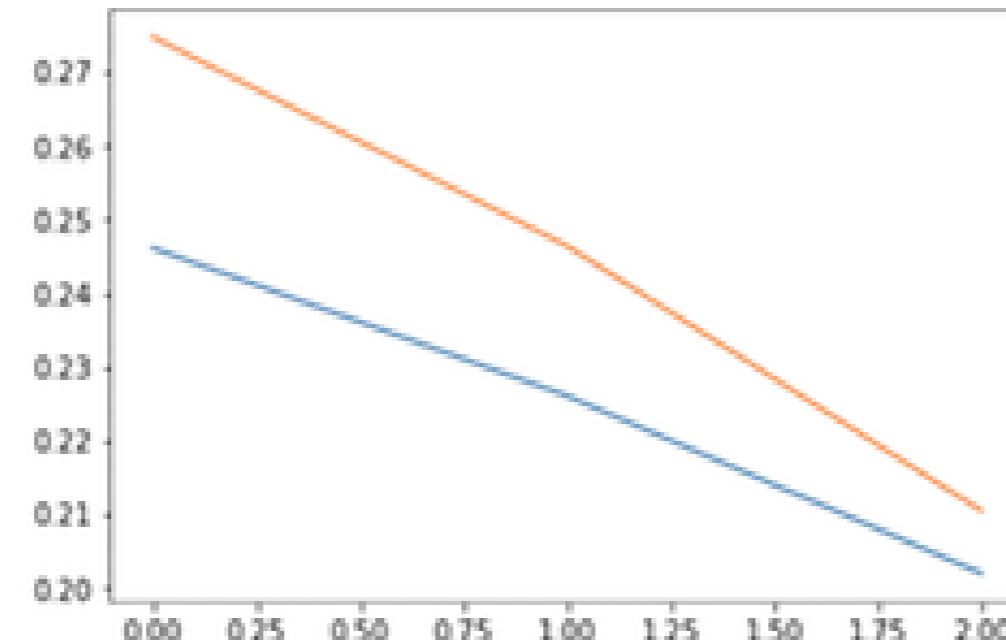
overfitting

overfitting
validation loss 가 가

Learning curves: overfitting



```
training = model.fit(train_data, train_labels,  
                      epochs=3, validation_split=0.2)  
  
import matplotlib.pyplot as plt  
plt.plot(training.history['loss'])  
plt.plot(training.history['val_loss'])  
plt.show()
```



pyplot

loss validation loss

.

Storing the optimal parameters

```
from keras.callbacks import ModelCheckpoint

# This checkpoint object will store the model parameters      callbacks module
# in the file "weights.hdf5"                                modelcheckpoint
#                                                               가
#                                                               hdf5

checkpoint = ModelCheckpoint('weights.hdf5', monitor='val_loss',
                             save_best_only=True),           가
# Store in a list to be used during training
callbacks_list = [checkpoint]
# Fit the model on a training set, using the checkpoint as a
#callback
model.fit(train_data, train_labels, validation_split=0.2,
          epochs=3, callbacks=callbacks_list)

< monitor >
      , . . . , validation set loss가 가
train set loss가 가                               'loss'           'val_loss'
      . . .
```

Loading stored parameters

```
model.load_weights('weights.hdf5')  
model.predict_classes(test_data)  
array([2, 2, 1, 2, 0, 1, 0, 1, 2, 0])
```

load_weights

testset

class

.

Let's practice!

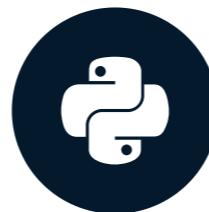
IMAGE PROCESSING WITH KERAS IN PYTHON

Neural network regularization

IMAGE PROCESSING WITH KERAS IN PYTHON

Ariel Rokem

Senior Data Scientist, University of Washington



Dropout

In each learning step:

- Select a subset of the units
- Ignore it in the forward pass
- And in the back-propagation of error

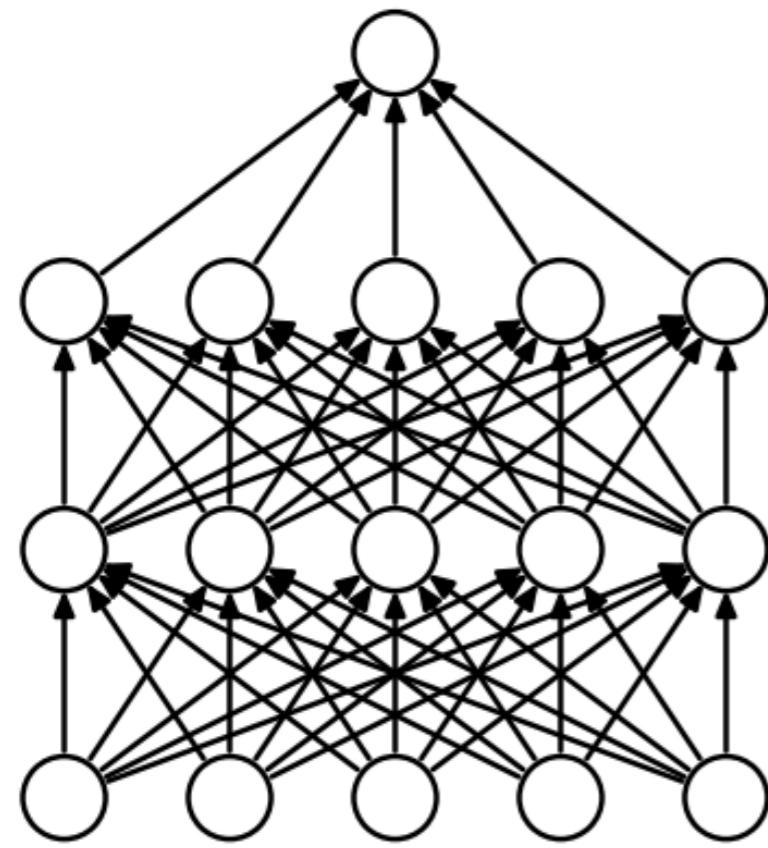
overfitting

dropout

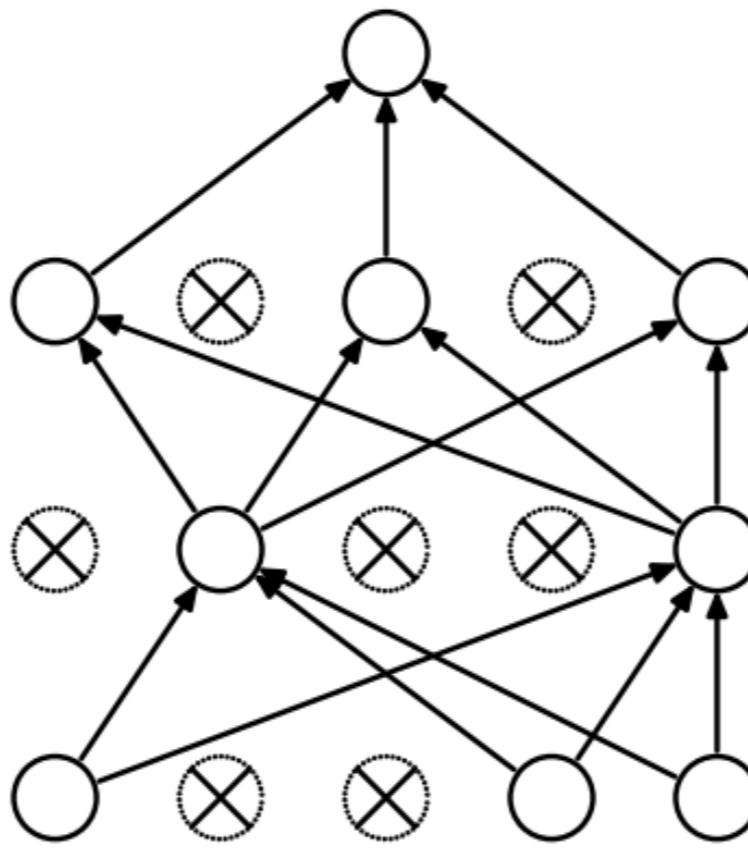
가

,

Dropout



(a) Standard Neural Net



(b) After applying dropout.

dropout
가 .

Dropout in Keras

```
from keras.models import Sequential  
from keras.layers import Dense, Conv2D, Flatten, Dropout  
model = Sequential()  
model.add(Conv2D(5, kernel_size=3, activation='relu',  
                input_shape=(img_rows, img_cols, 1)))  
model.add(Dropout(0.25))  
model.add(Conv2D(15, kernel_size=3, activation='relu'))  
model.add(Flatten())  
model.add(Dense(3, activation='softmax'))
```

dropout

가

0.25

0.25

drop

가

.

Batch normalization

- Rescale the outputs

Batch normalization Mini batch
Backpropagation
Batch normalization

가
.

Batch Normalization in Keras

```
from keras.models import Sequential  
from keras.layers import Dense, Conv2D, Flatten, BatchNormalization  
  
model = Sequential()  
model.add(Conv2D(5, kernel_size=3, activation='relu',  
                input_shape=(img_rows, img_cols, 1)))  
model.add(BatchNormalization())  
model.add(Conv2D(15, kernel_size=3, activation='relu'))  
model.add(Flatten())  
model.add(Dense(3, activation='softmax'))
```

layer batch normalization 가 .

Be careful when using them together!

The disharmony between dropout and batch normalization

dropout
가

batch normalization

,

Let's practice!

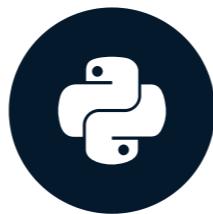
IMAGE PROCESSING WITH KERAS IN PYTHON

Interpreting the model

IMAGE PROCESSING WITH KERAS IN PYTHON

Ariel Rokem

Senior Data Scientist, University of Washington



Selecting layers

```
model.layers
```

```
[<keras.layers.convolutional.Conv2D at 0x109f10c18>,
<keras.layers.convolutional.Conv2D at 0x109ec5ba8>,
<keras.layers.core.Flatten at 0x1221ffcc0>,
<keras.layers.core.Dense at 0x1221ffff0>]
```

```
layer      layer      .      2      layer      flatten, dencse      .
```

Getting model weights

```
conv1 = model.layers[0]  
weights1 = conv1.get_weights()  
len(weights1)
```

2

layer

get_weights

```
kernels1 = weights1[0]  
kernels1.shape kernel shape
```

3X3X1X5 shape 가

: 5

(3, 3, 1, 5)

```
kernel1_1 = kernels1[:, :,  
                      ,  
                      0, 0]  
kernel1_1.shape
```

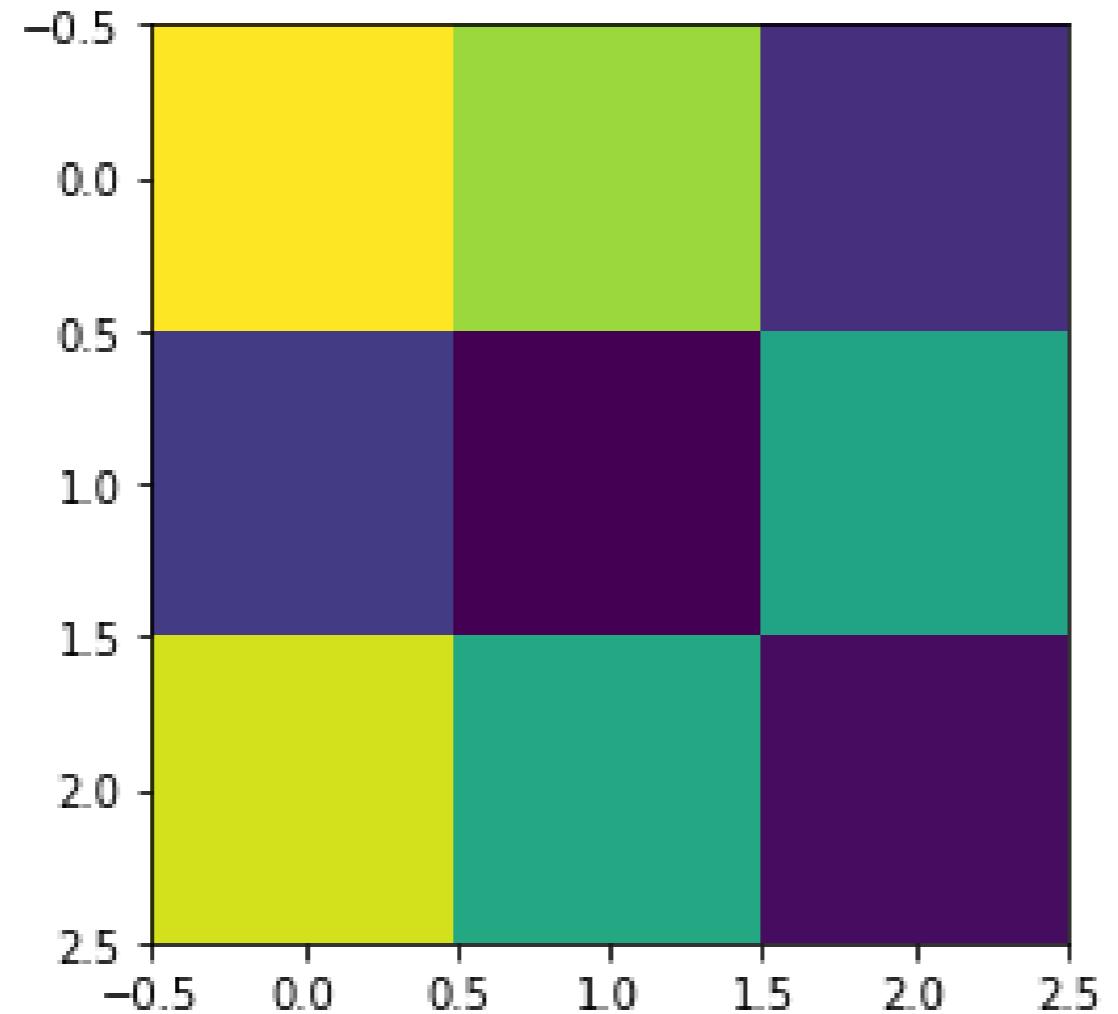
0

(3, 3)

3x3

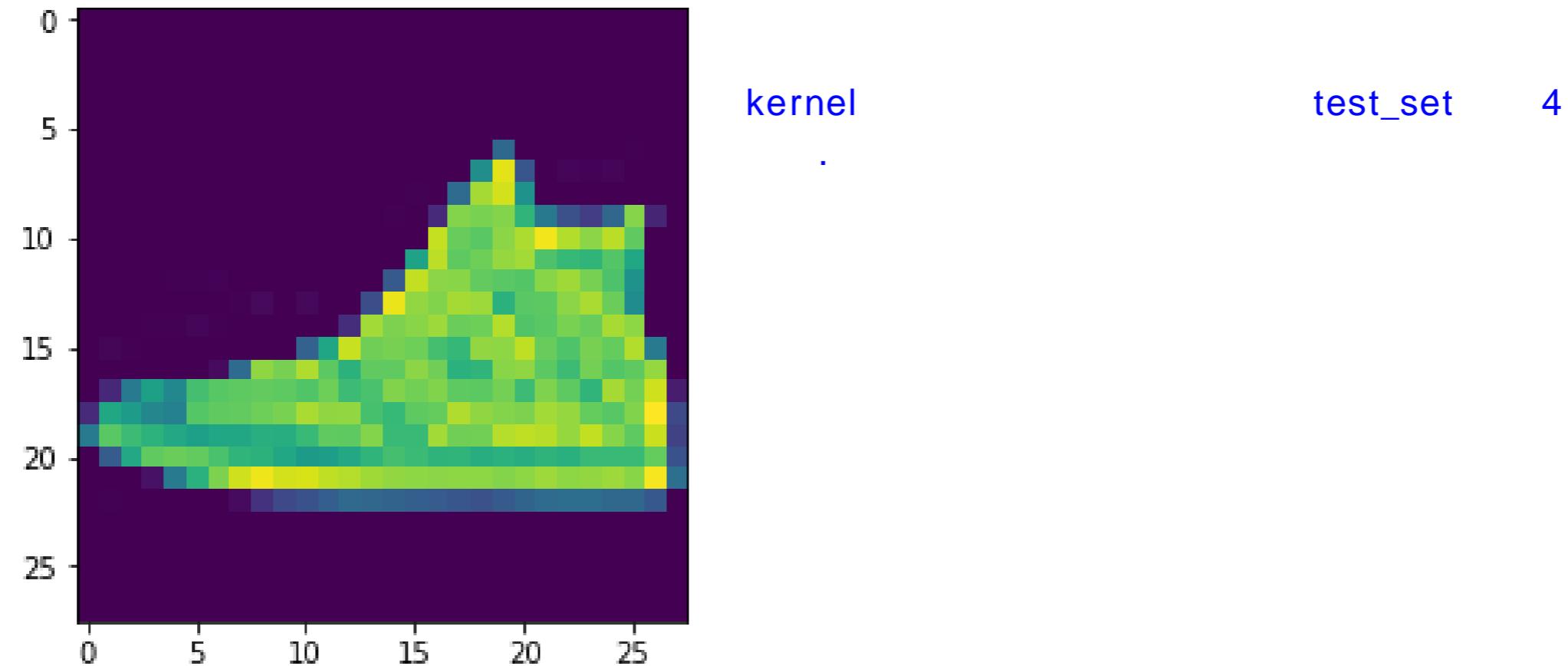
Visualizing the kernel

```
plt.imshow(kernel1_1)
```



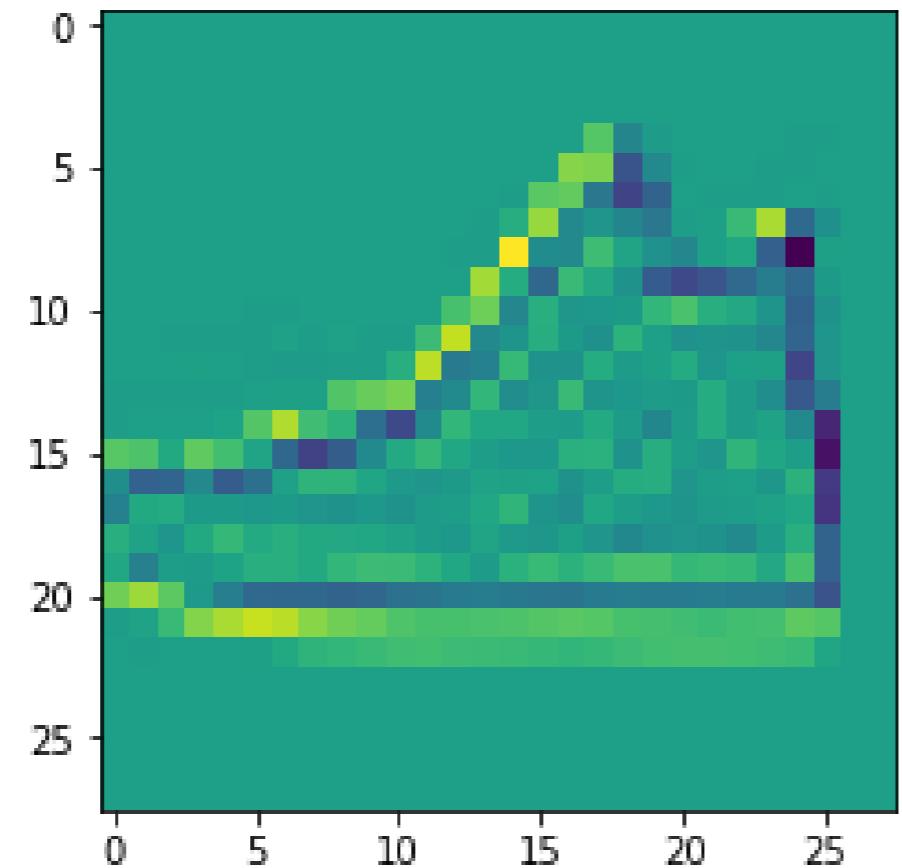
Visualizing the kernel responses

```
test_image = test_data[3, :, :, 0]  
plt.imshow(test_image)
```



Visualizing the kernel responses

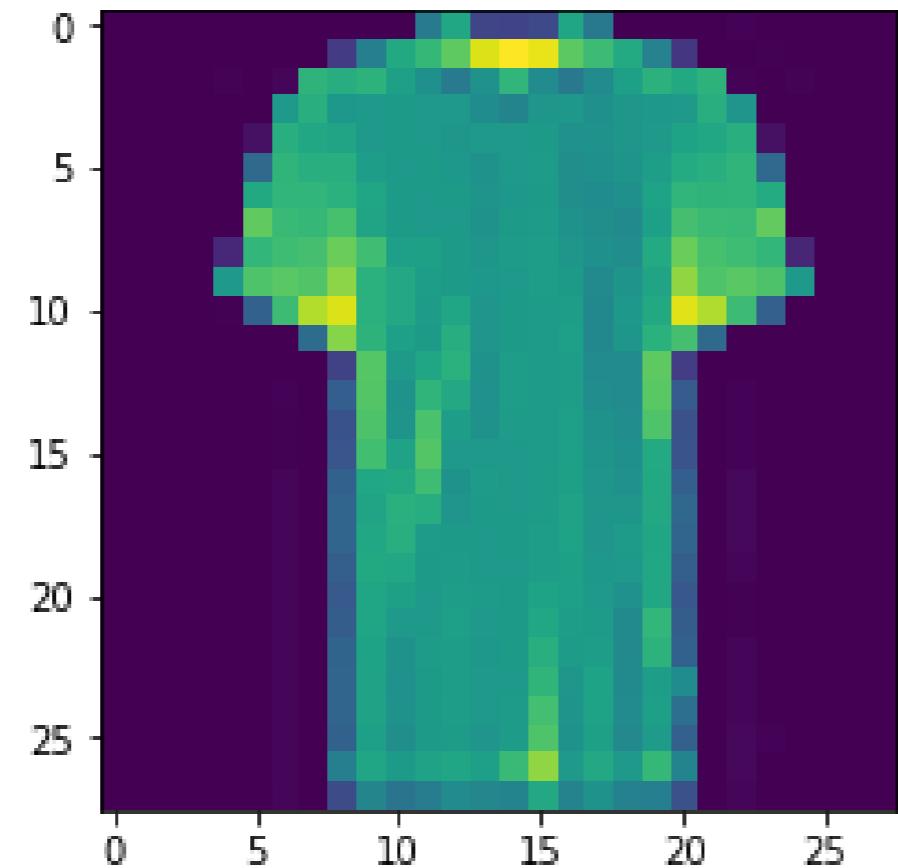
```
filtered_image = convolution(test_image, kernel1_1)  
plt.imshow(filtered_image)
```



가

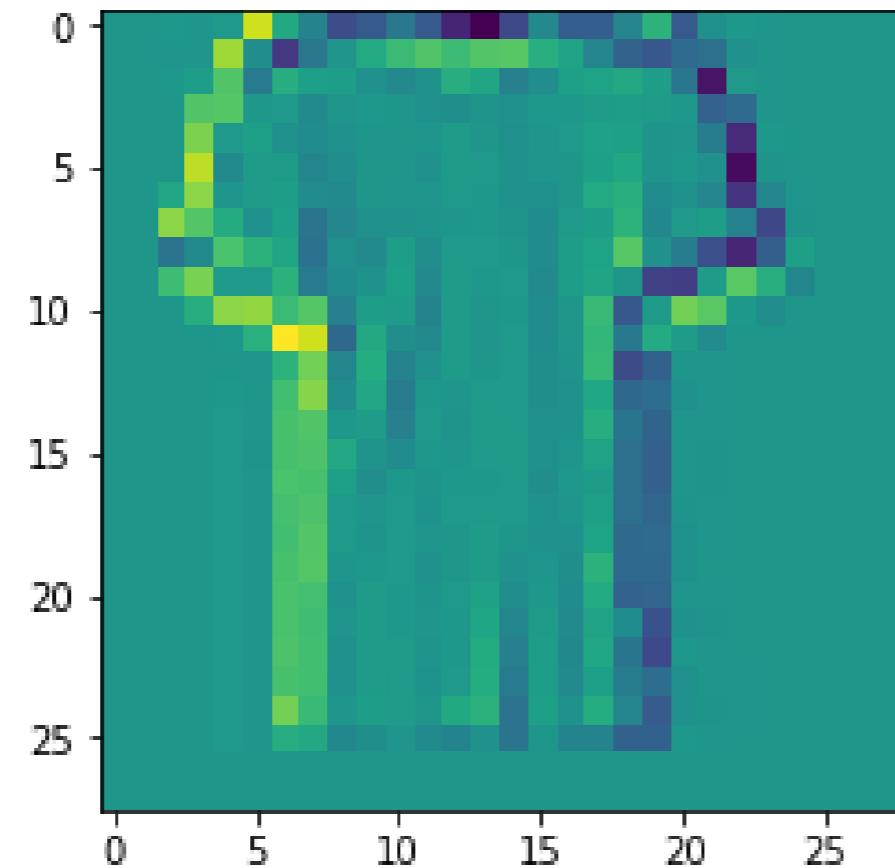
Visualizing the kernel responses

```
test_image = test_data[4, :, :, 1]  
plt.imshow(test_image)
```



Visualizing the kernel responses

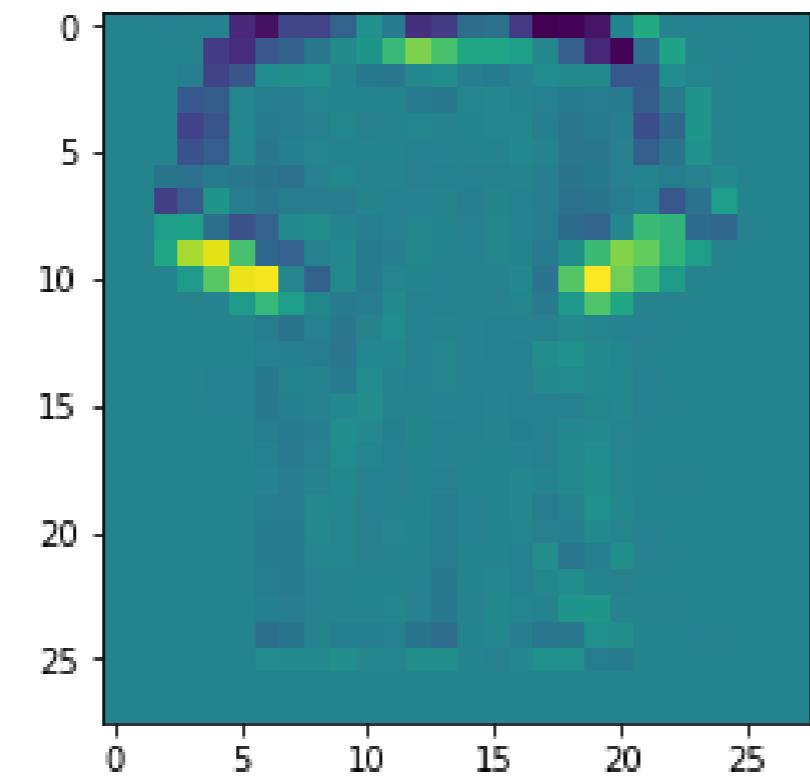
```
filtered_image = convolution(test_image, kernel1_1)  
plt.imshow(filtered_img)
```



가 가

Visualizing the kernel responses

```
kernel1_2 = kernels[:, :, 0, 1]
filtered_image = convolution(test_image, kernel1_2)
plt.imshow(filtered_img)
```



Let's practice!

IMAGE PROCESSING WITH KERAS IN PYTHON

Wrapping up

IMAGE PROCESSING WITH KERAS IN PYTHON



Ariel Rokem

Senior Data Scientist, University of
Washington

What did we learn?

- Image classification
 - Convolutions
 - Reducing the number of parameters
 - Tweaking your convolutions . CNN
 - Adding pooling layers
 - Improving your network
 - Regularization
 - Understanding your network
 - Monitoring learning
 - Interpreting the parameters

가 가

가

가 가

가

가

가

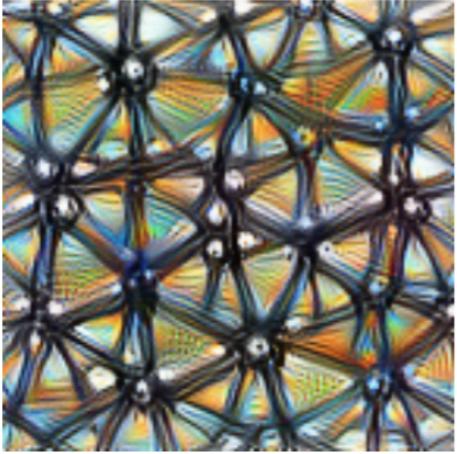
가

Model interpretation

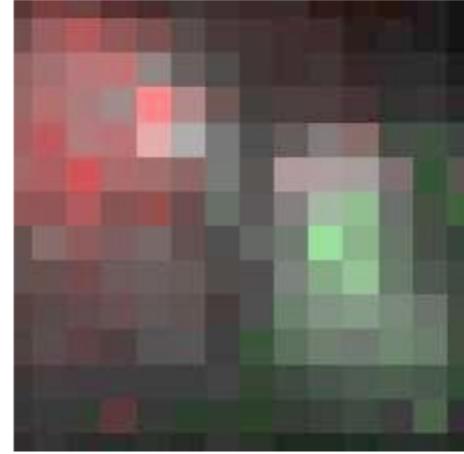
<https://distill.pub/2017/feature-visualization/>



Feature visualization answers questions about what a network—or parts of a network—are looking for by generating examples.



Attribution¹ studies what part of an example is responsible for the network activating a particular way.





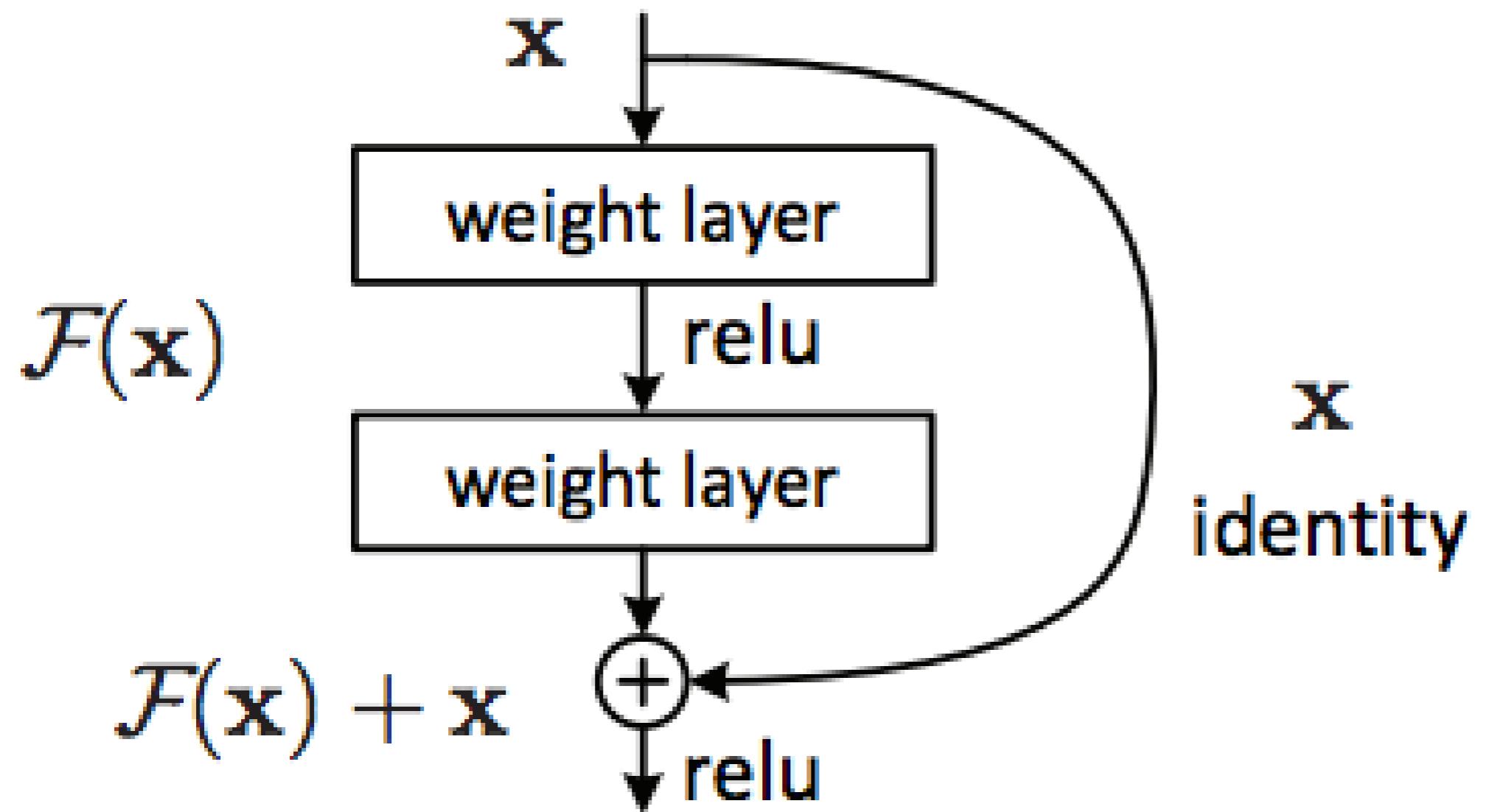
What next?

- Even deeper networks
- Residual networks

가

가

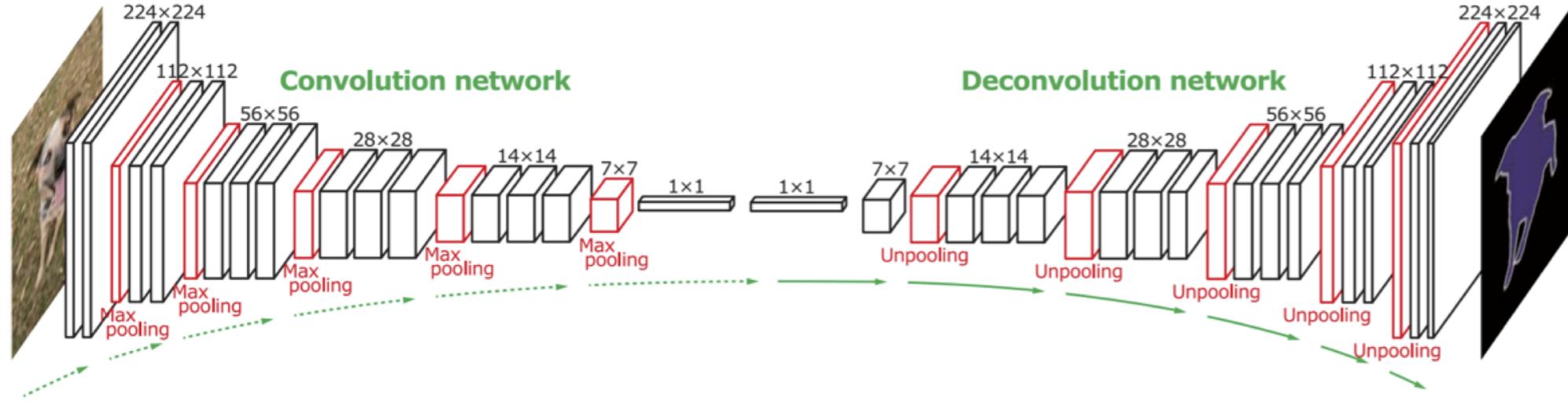
Residual networks



What next?

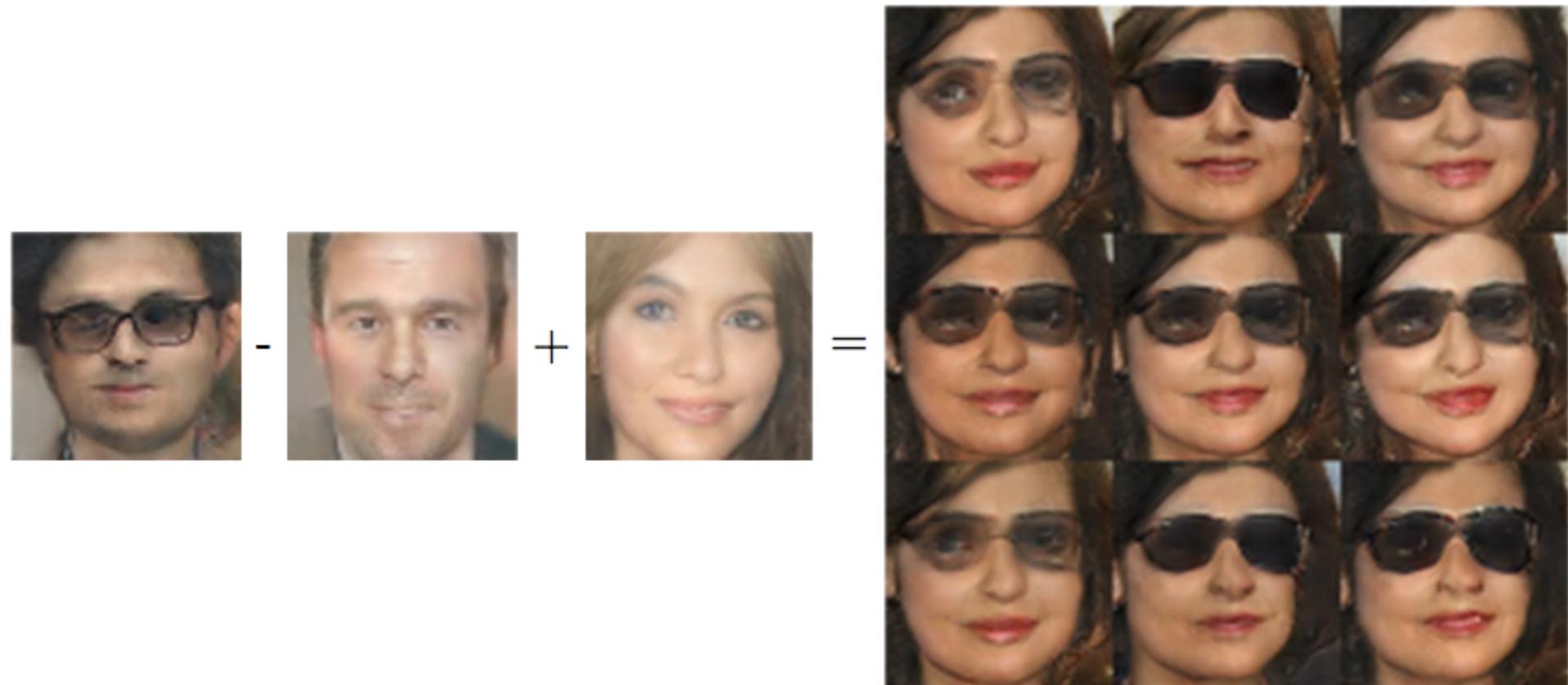
- Even deeper networks
- Residual networks
- Transfer learning
- Fully convolutional networks

Fully convolutional networks



가

Generative adversarial networks



What next?

- Even deeper networks
- Residual networks
- Transfer learning
- Fully convolutional networks
- Generative adversarial networks
- ...

가

Good luck!

IMAGE PROCESSING WITH KERAS IN PYTHON