MACHINE LEARNING WITH TENSORFLOW

Ben Hendrickson
Physics Dept.
Portland State University
4.19.2019

TODAY

- Brief(!) overview of neural network components & machine learning
- Run through an example
- Real life research example

AN EXAMPLE

- What’s the number in my head?

AN EXAMPLE

- What’s the number in my head?

- It’s 5!

AN EXAMPLE

- What’s the number in my head?

AN EXAMPLE

- What's the number in my head?

- It's 5!

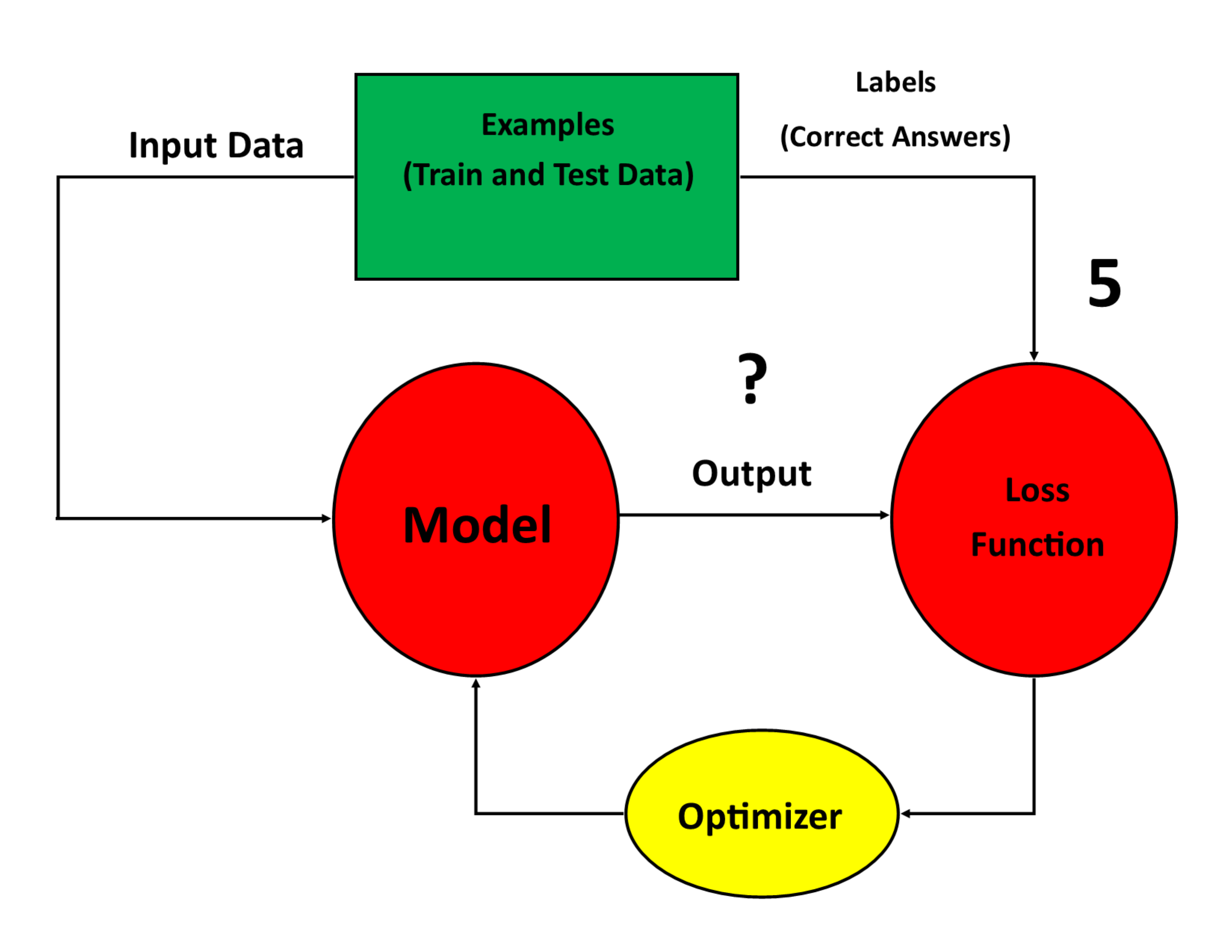
AN EXAMPLE

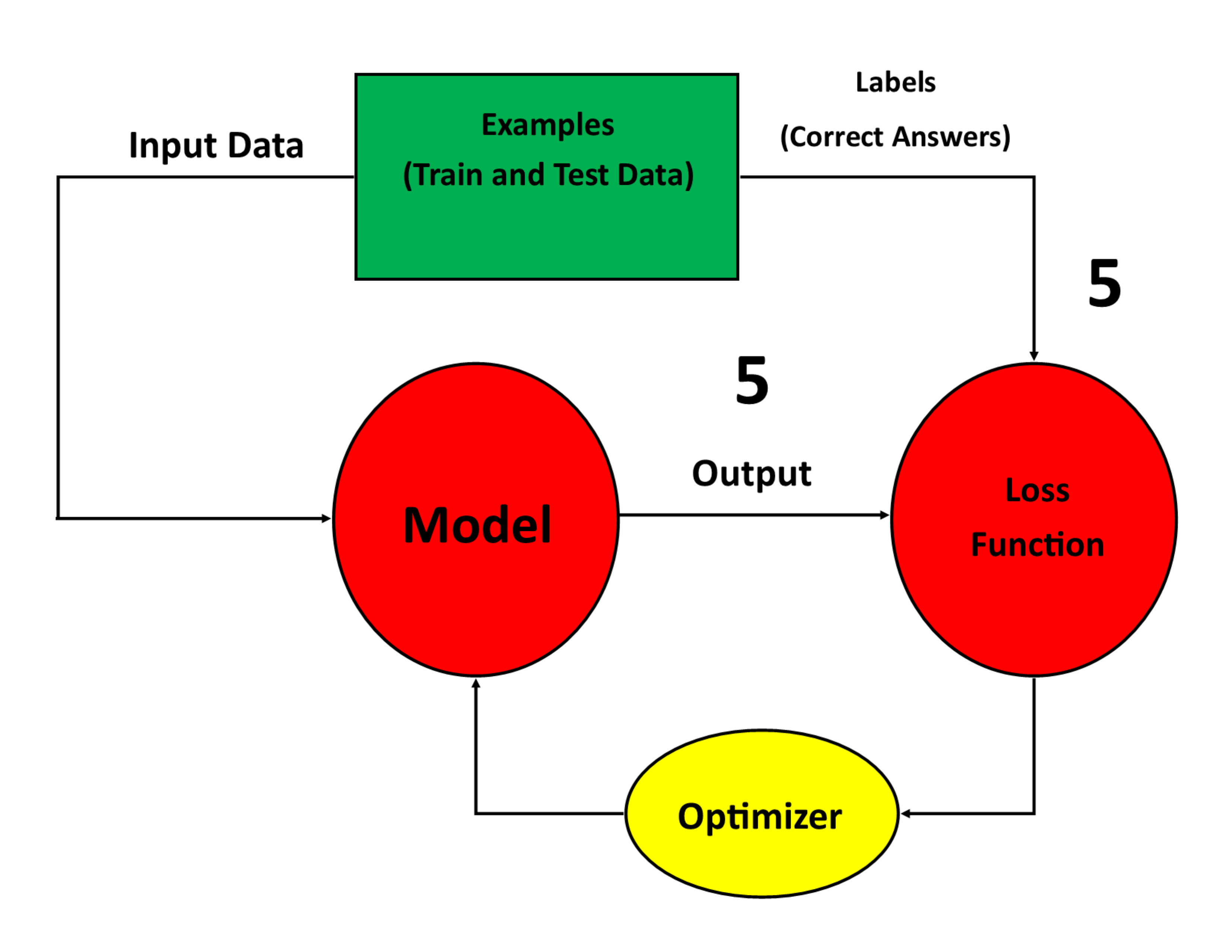
- What’s the number in my head?

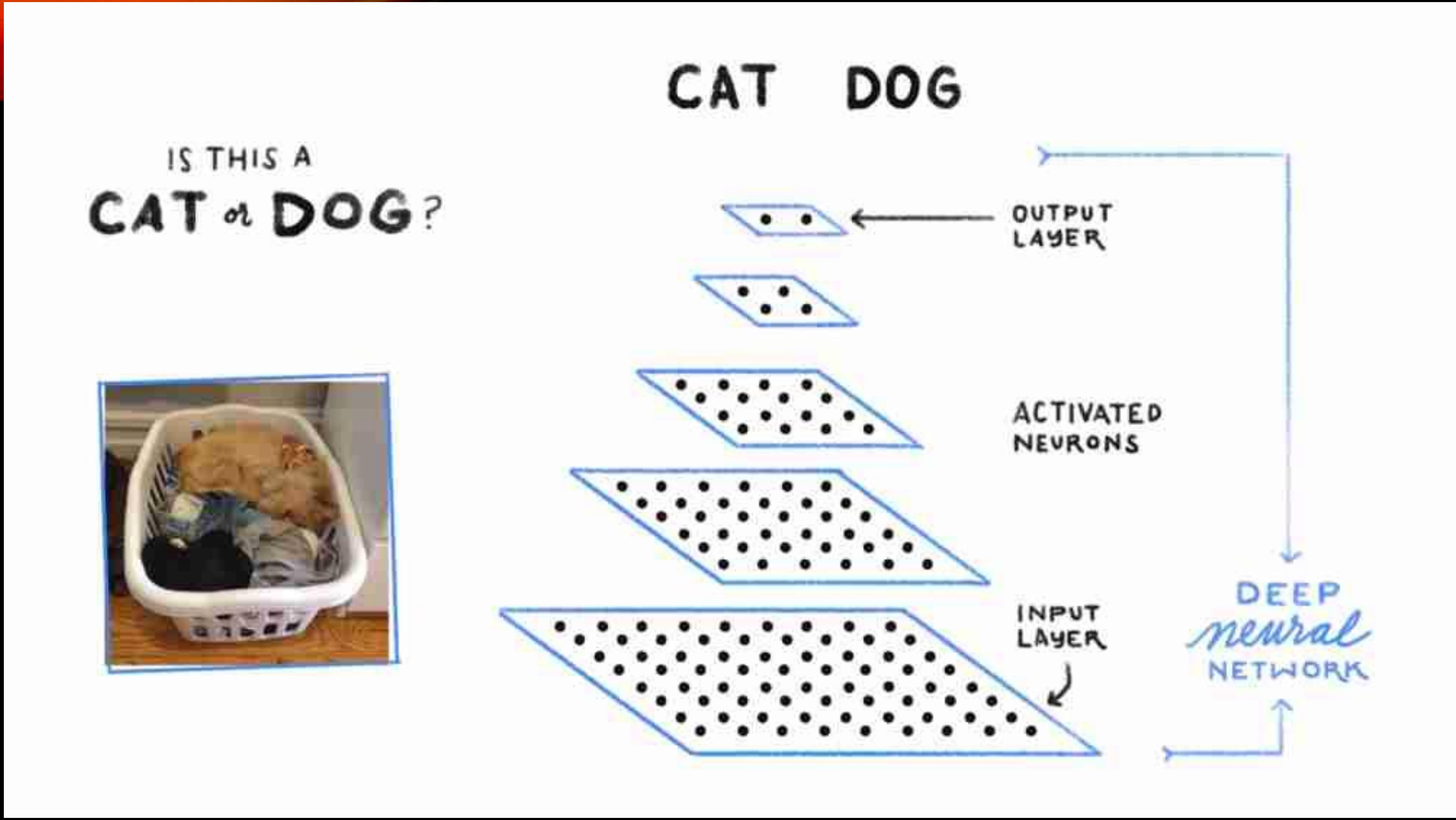
AN EXAMPLE

- What’s the number in my head?

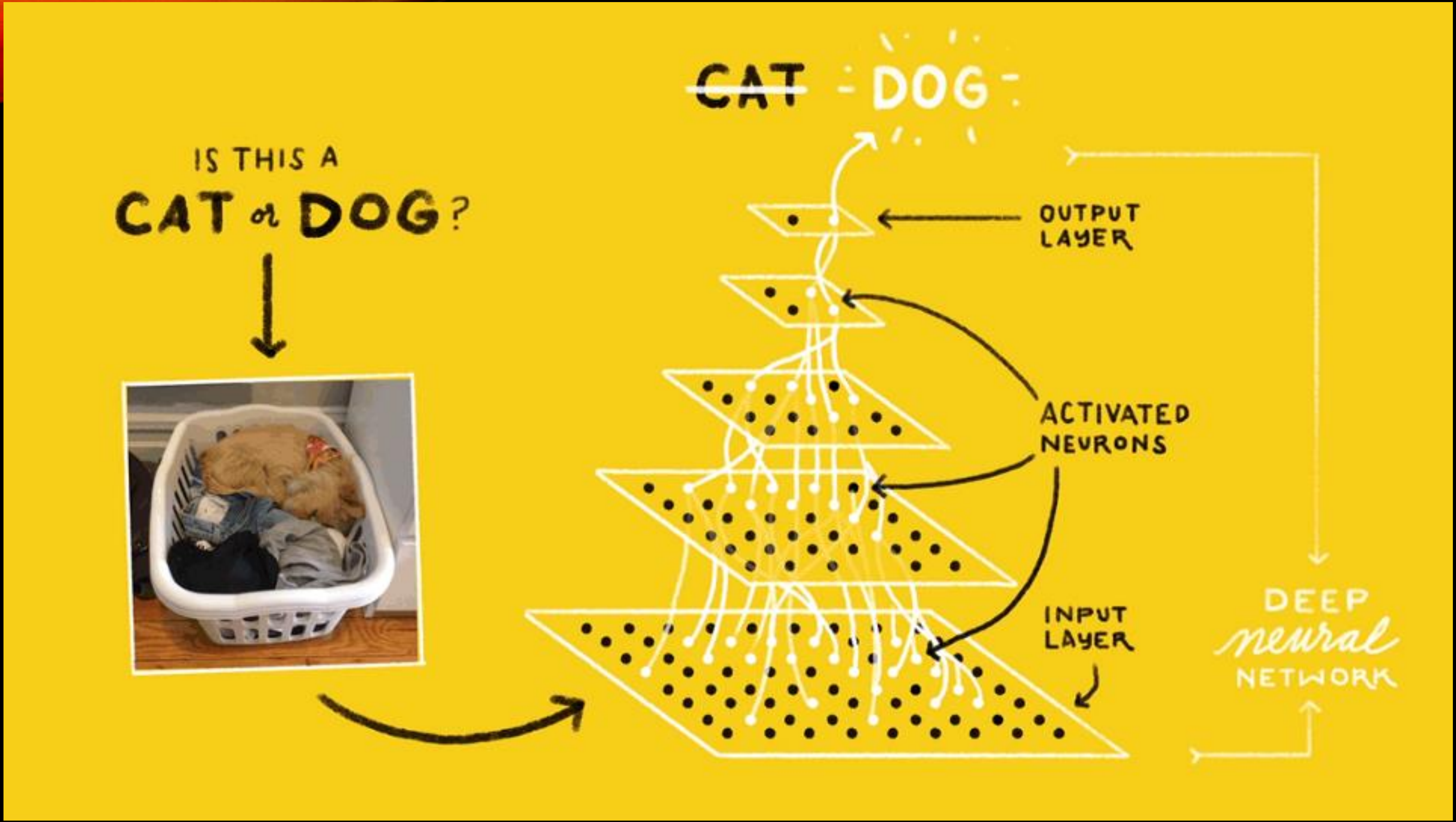
- It’s 5!


Examples
(Train and Test Data)
Input Data
Labels
(Correct Answers)
5
?
Output
Model
Loss
Function
Optimizer


Examples
(Train and Test Data)
Input Data
Labels
(Correct Answers)
5
5
Output
Model
Loss
Function
Optimizer

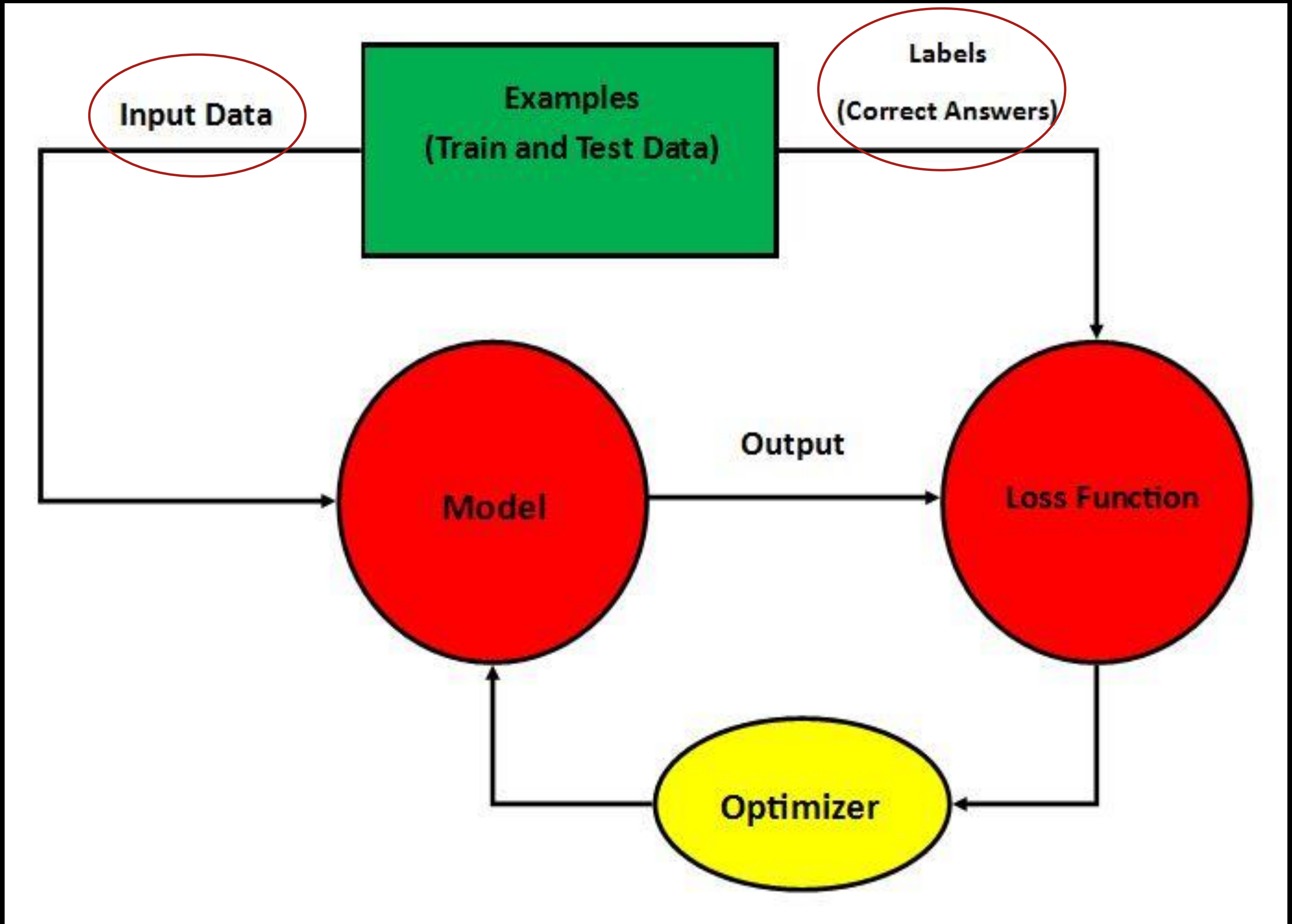


CAT DOG
IS THIS A
CAT or DOG?
OUTPUT LAYER
ACTIVATED NEURONS
INPUT LAYER
DEEP neural NETWORK


IS THIS A
CAT or DOG?
~~CAT~~ DOG
OUTPUT LAYER
ACTIVATED NEURONS
INPUT LAYER
DEEP neural NETWORK

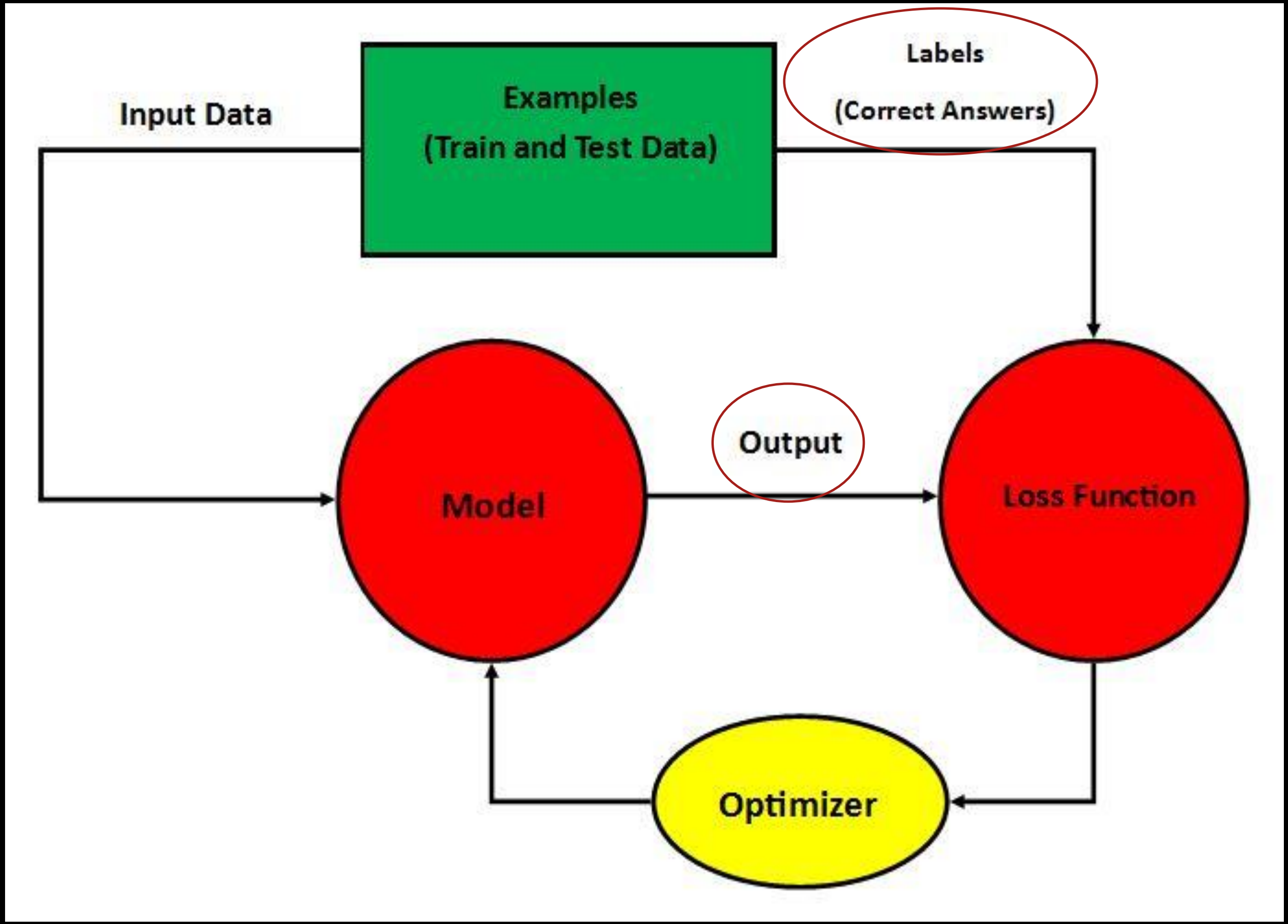
HOW DO WE MAKE A GOOD CLASSIFICATION MODEL?

- Create a good set of training data
  - Can be real, or a good simulated representation
- Assign labels to each piece of data



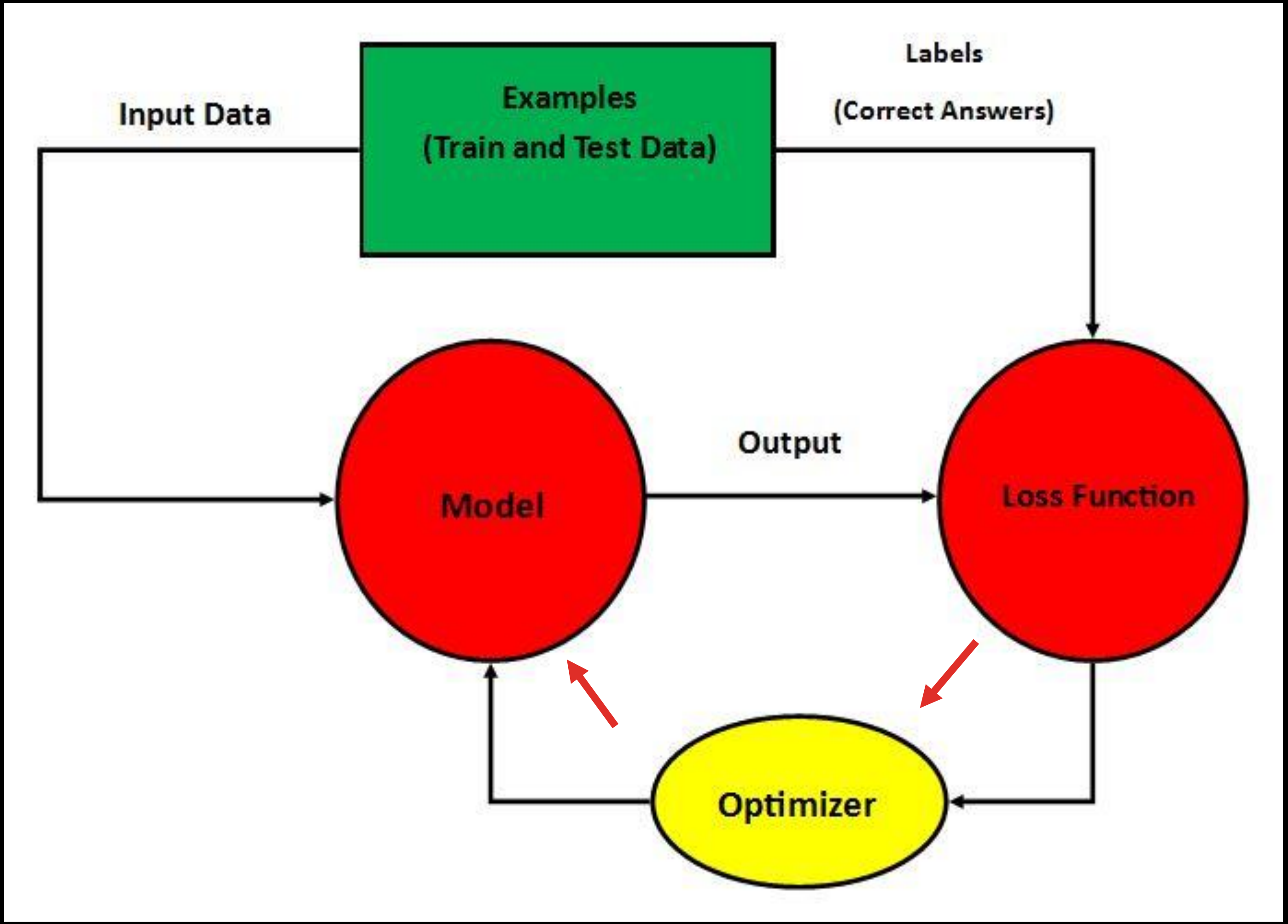
HOW DO WE MAKE A GOOD CLASSIFICATION MODEL?

- During training, the model takes each piece of data, and guesses its class.
- The error is measured by the so-called loss function



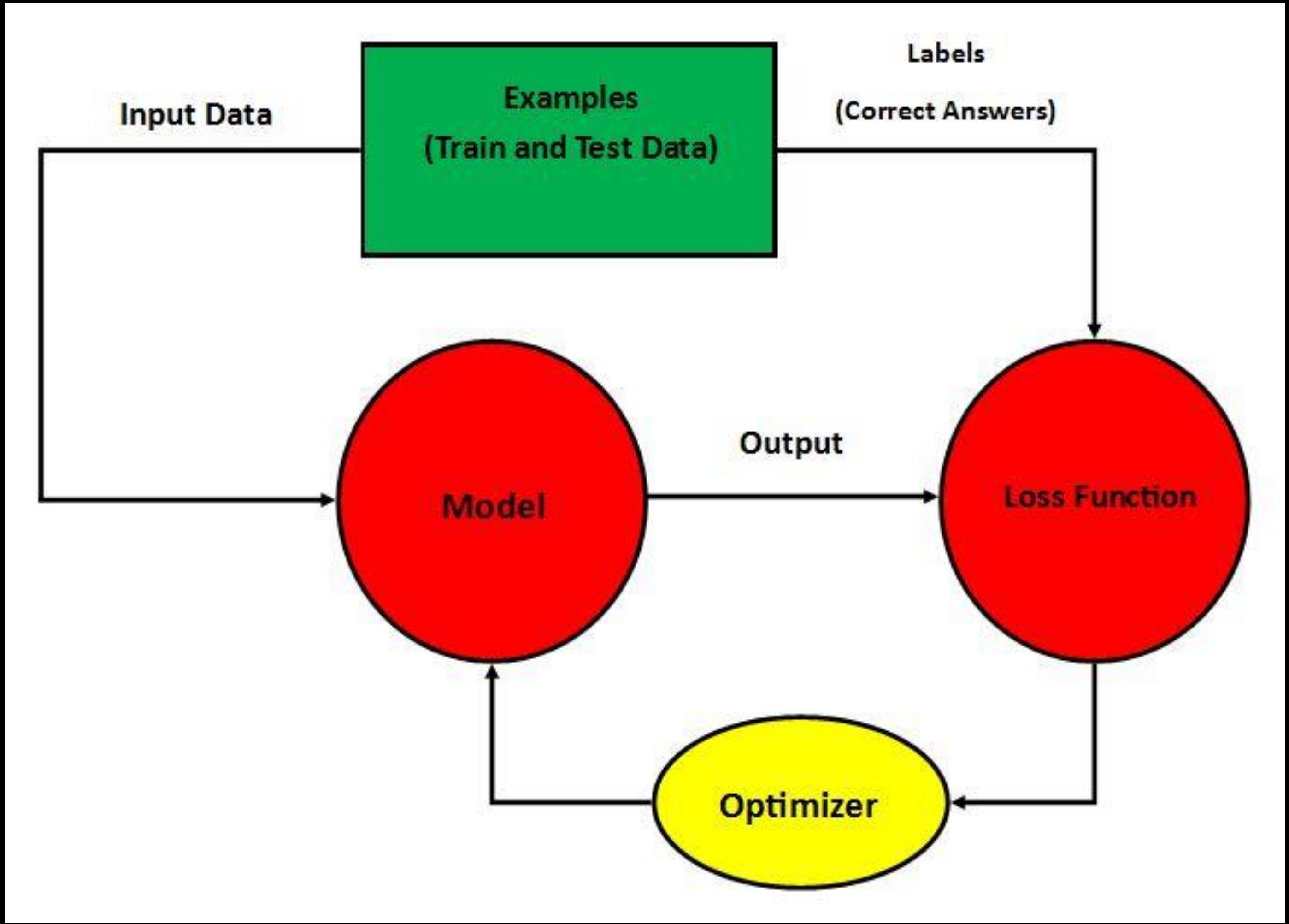
HOW DO WE MAKE A GOOD CLASSIFICATION MODEL?

- The amount of error is fed into the optimizer, which performs some gradient descent algorithm to tweak the model.



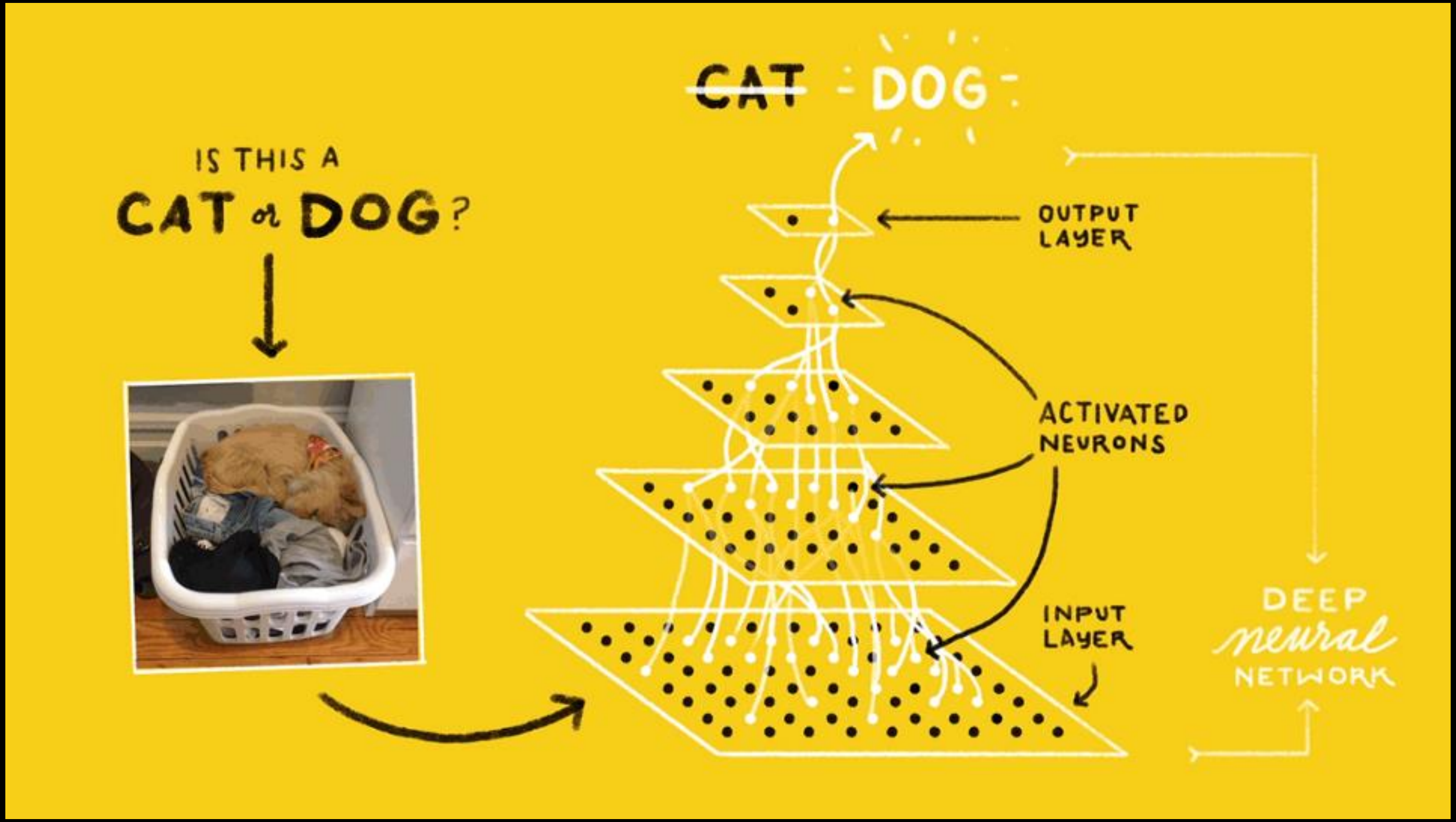
HOW DO WE MAKE A GOOD CLASSIFICATION MODEL?

THAT'S IT!!



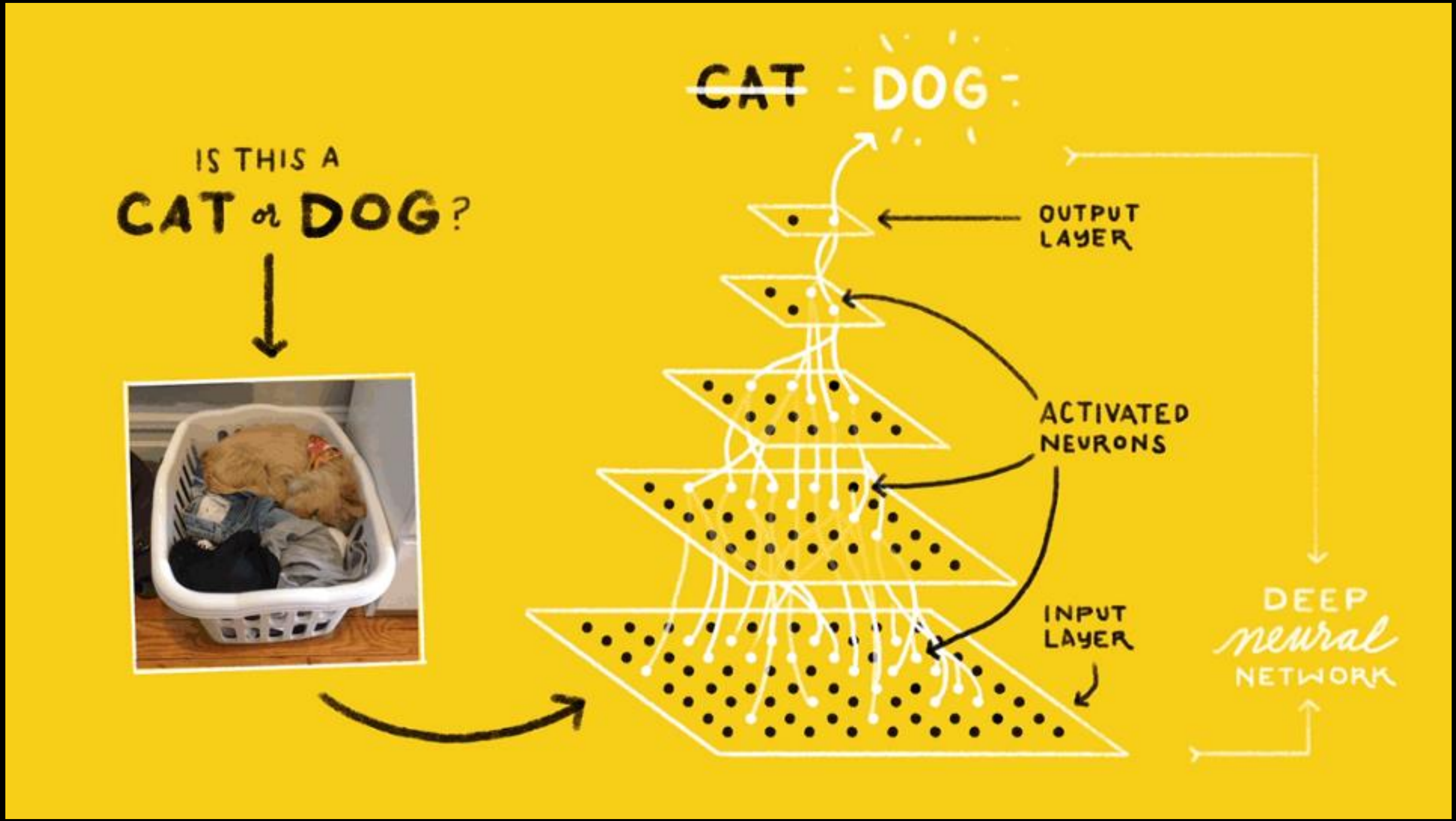
NOW, SOME LOOSE DEFINITIONS

- This is a dense, or fully connected model. Only one kind of model!!
- Components include:
  - The input layer
  - Neurons (activations)
  - Connections (with various weights)
  - An output layer
- The training process changes weights and activations so a dog picture activates some neurons, while cat pictures activate others.



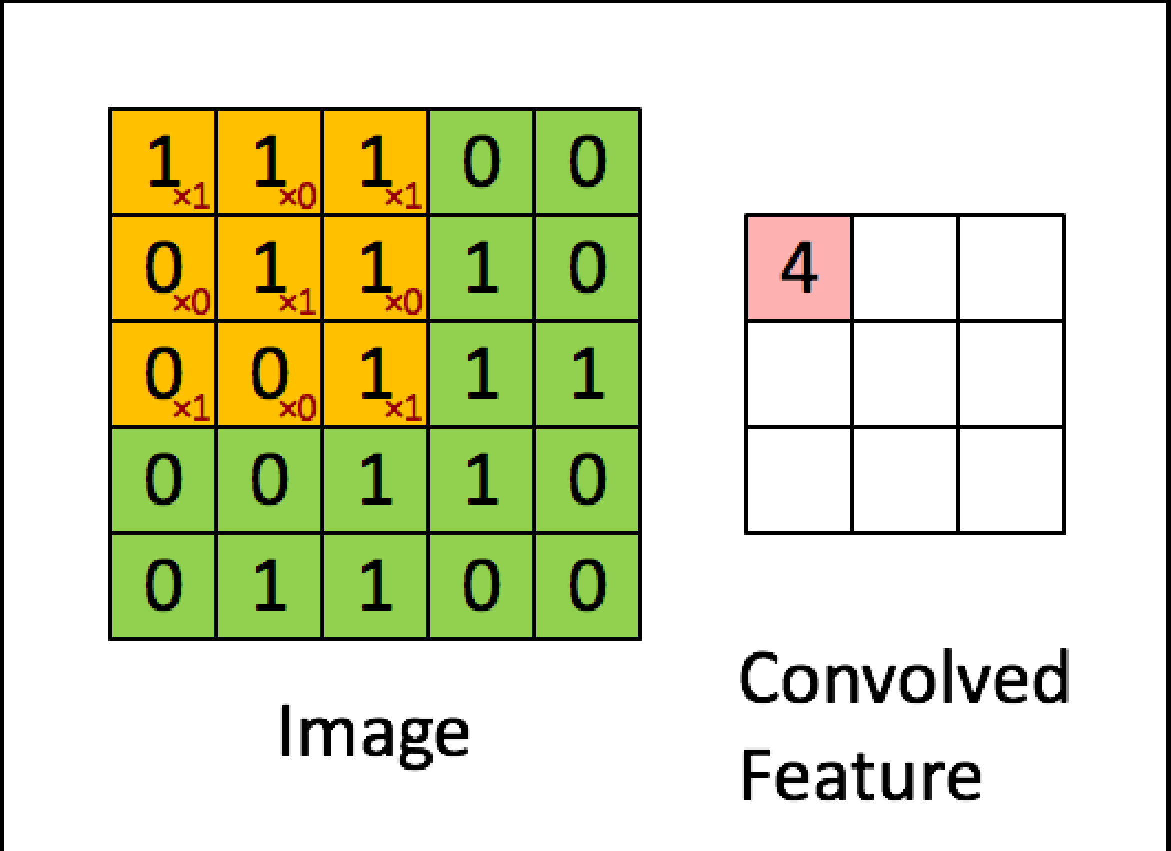
NOW, SOME LOOSE DEFINITIONS

- There are different kinds of layers!



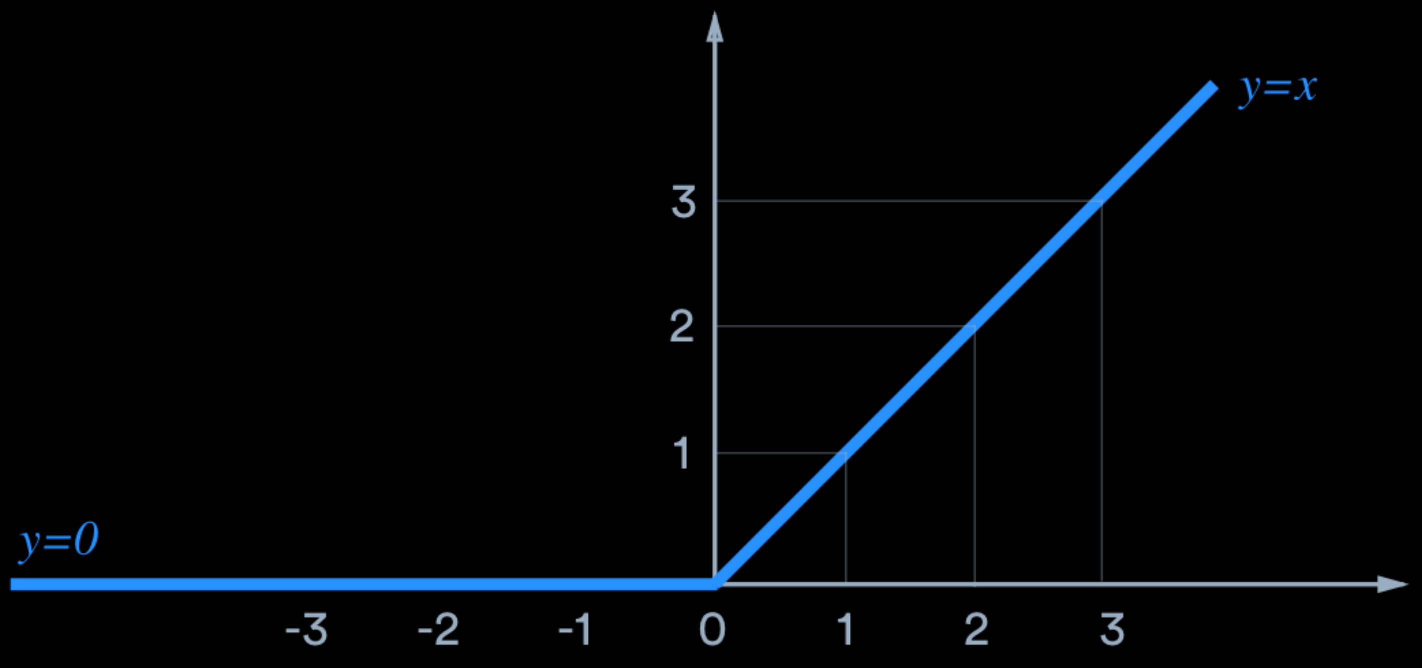
NOW, SOME LOOSE DEFINITIONS

- A fully connected can become unwieldy since each neuron has to touch every output from the previous layer.
- Convolutional layers can simplify the model by sliding a filter across a layer, and creating feature maps.
- Became very popular a few years ago (2015?), and are now standard.
- Can improve performance by focusing on feature extraction.



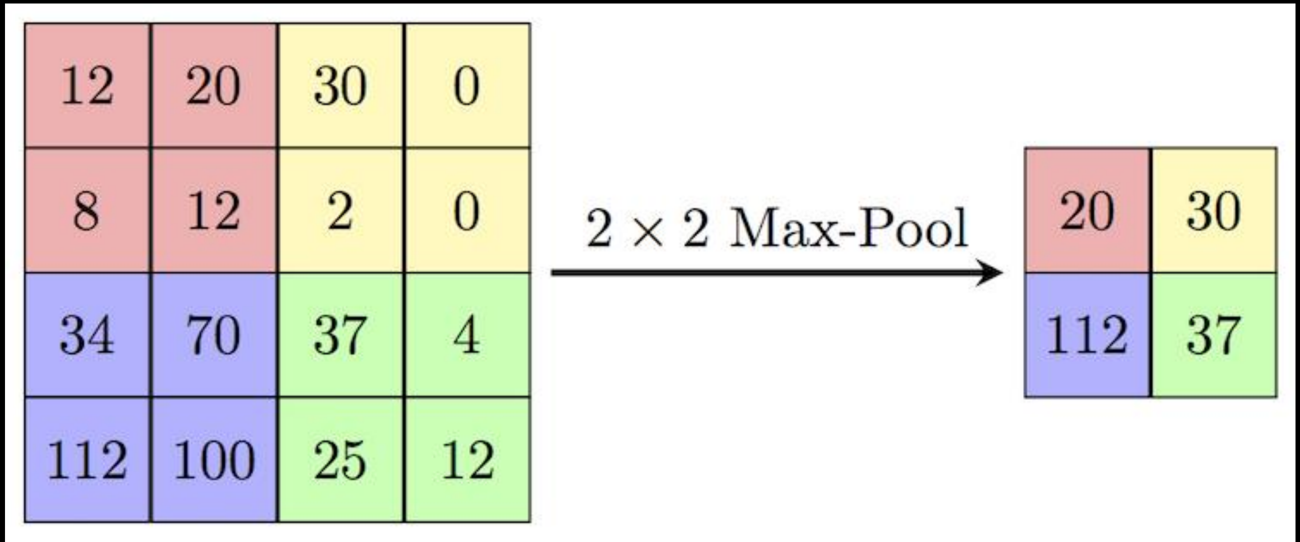
NOW, SOME LOOSE DEFINITIONS

- Activation gives the model its non-linear kick
- Occurs after convolution
- Operates on the feature map
- Non-linearity allows the model to learn more complex features
- Without it, the entire model becomes a reversible transformation



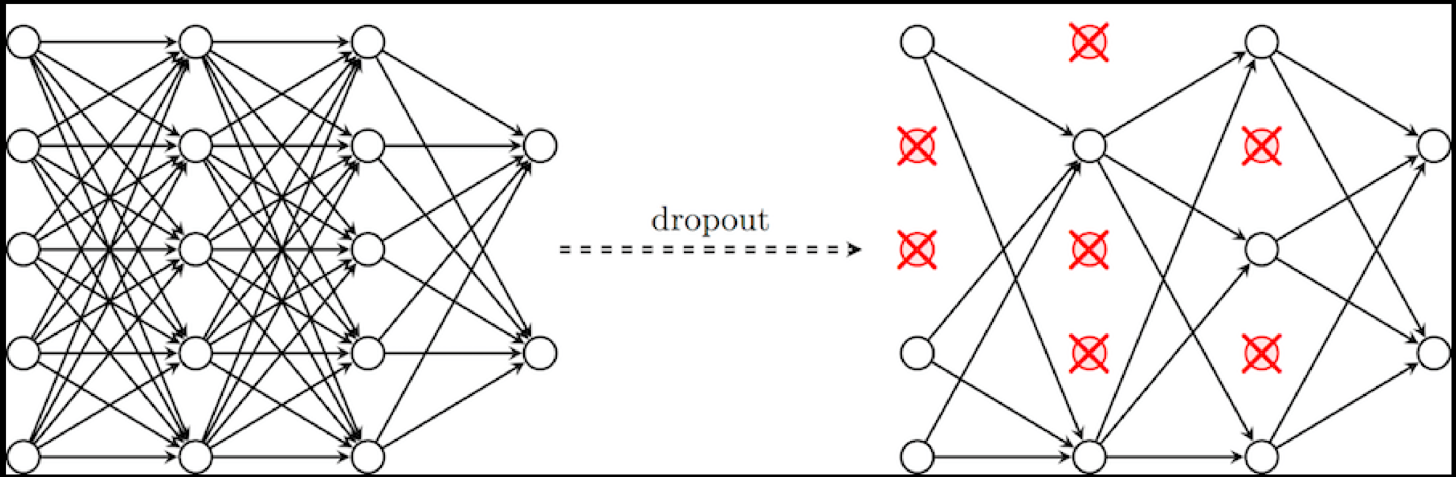
NOW, SOME LOOSE DEFINITIONS

- Pooling layers reduce the dimensionality of a feature map by down sampling.
- Eases computational load, fewer numbers to crunch.
- Typically happens directly following a convolutional layer to throw out less important features.



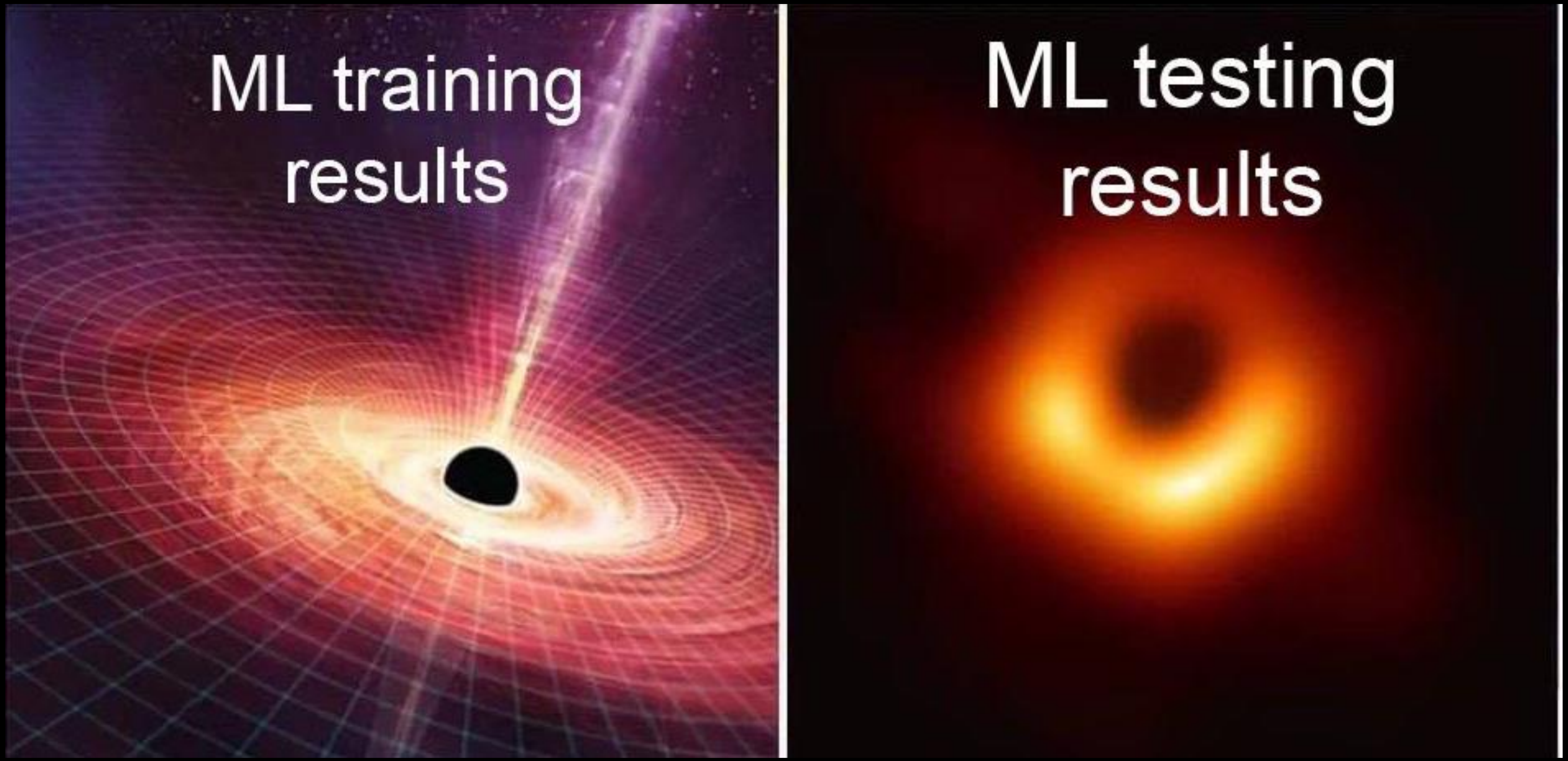
NOW, SOME LOOSE DEFINITIONS

- Dropout randomly turns off connections between layers.
- Useful to help prevent overfitting



NOW, SOME LOOSE DEFINITIONS

- Overfitting: The model is trained to work VERY well on the training data, but isn't generalized well enough.
- Tests poorly.



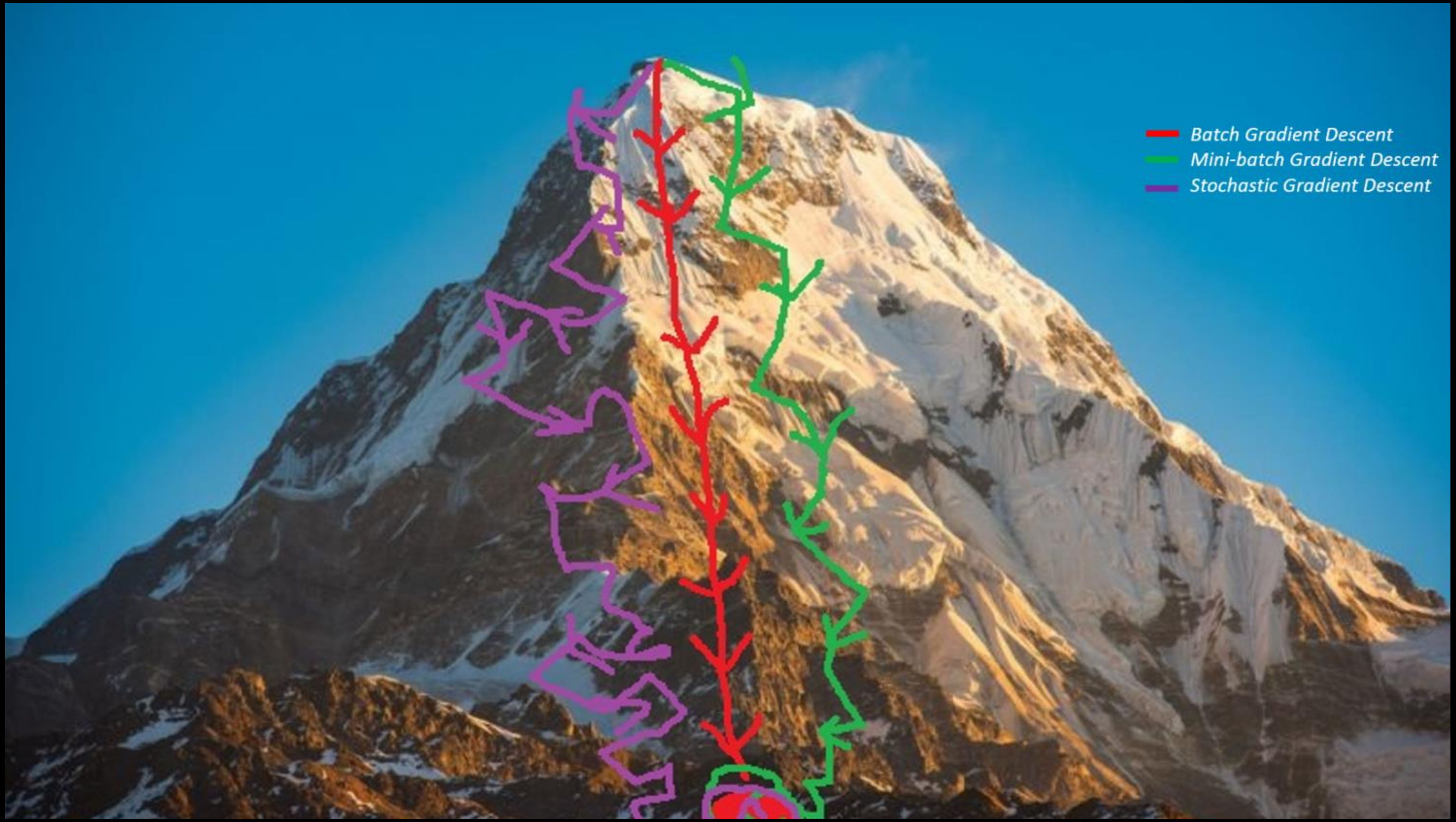
NOW, SOME LOOSE DEFINITIONS

- The loss function measures the quality of the prediction.
- The function below is called ‘binary cross-entropy’.
- t is the target, or label.
- y is the prediction from the model.
- If t_i is zero, and y_i is small, E will be small.

$$E = -\sum_{i=1}^{n}(t_i \log(y_i) + (1 - t_i)\log(1 - y_i)$$

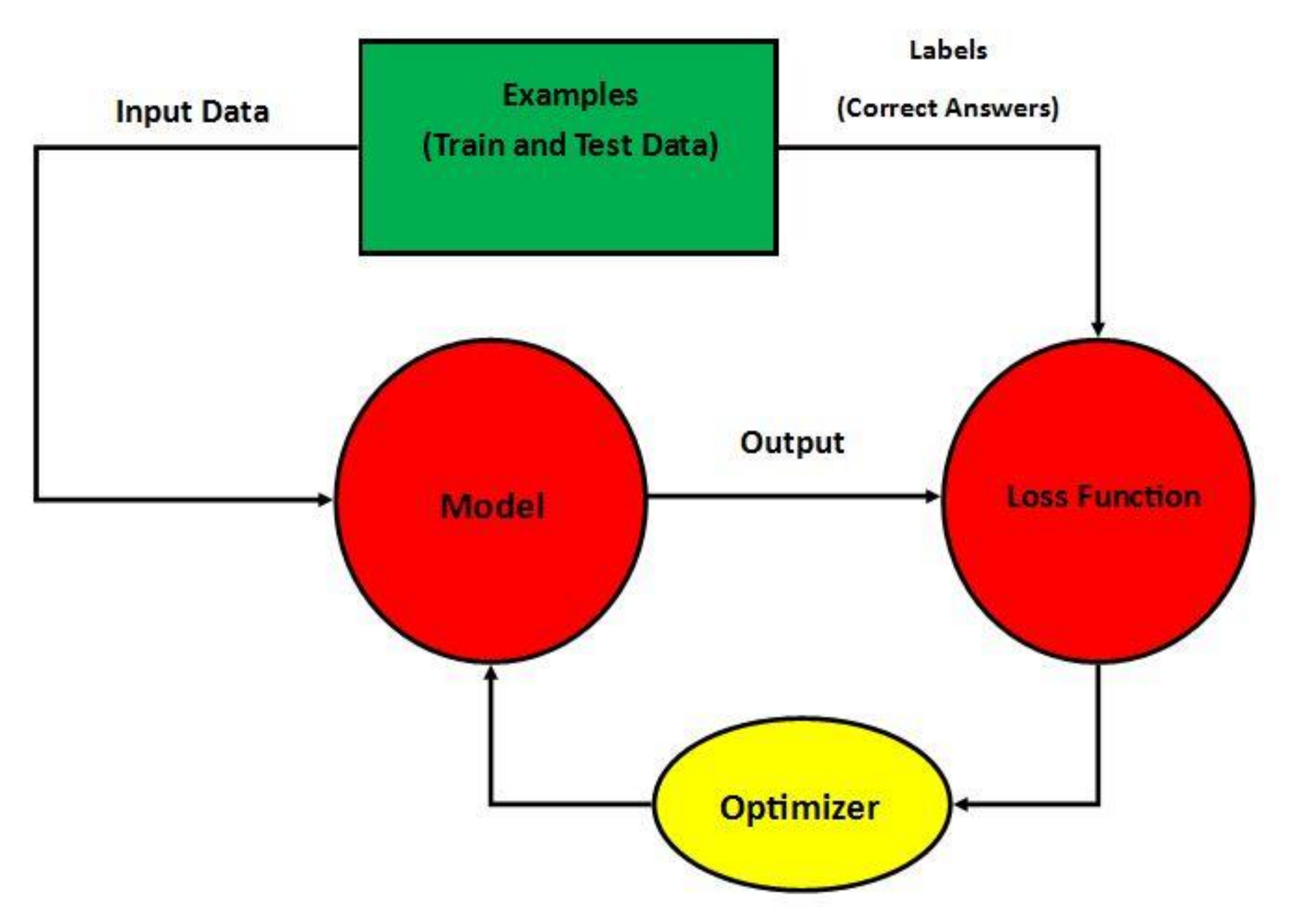
NOW, SOME LOOSE DEFINITIONS

- Optimization tunes the knobs of the model by taking the gradient of the loss function with respect to the weights of the model
- Basically a fancy minimization



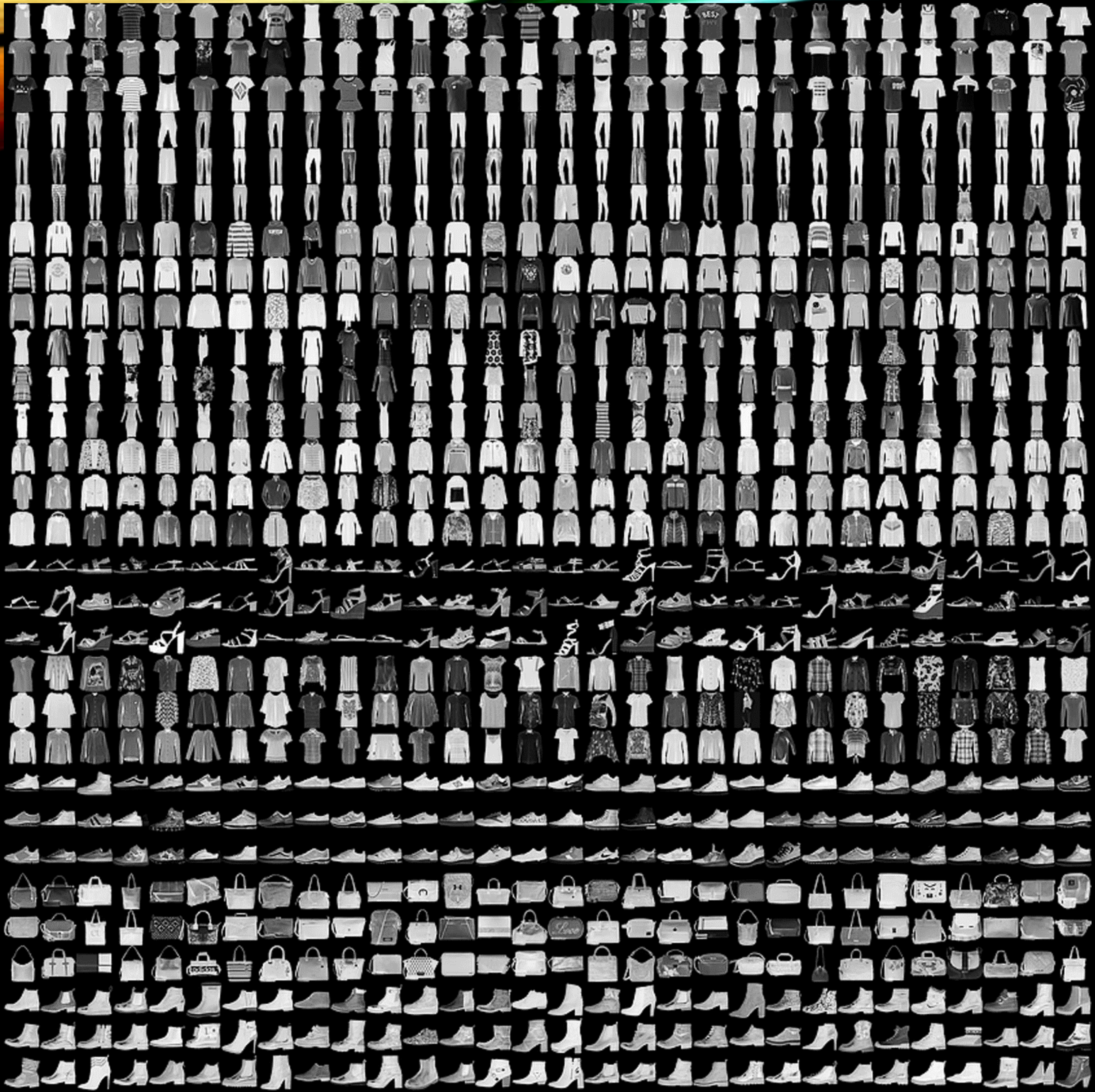
RINSE AND REPEAT!

- Run data through model
- Compare against the label
- Compute the loss function
- Tune the model in order to minimize the loss function
- Run the next piece of data
- Run several epochs!

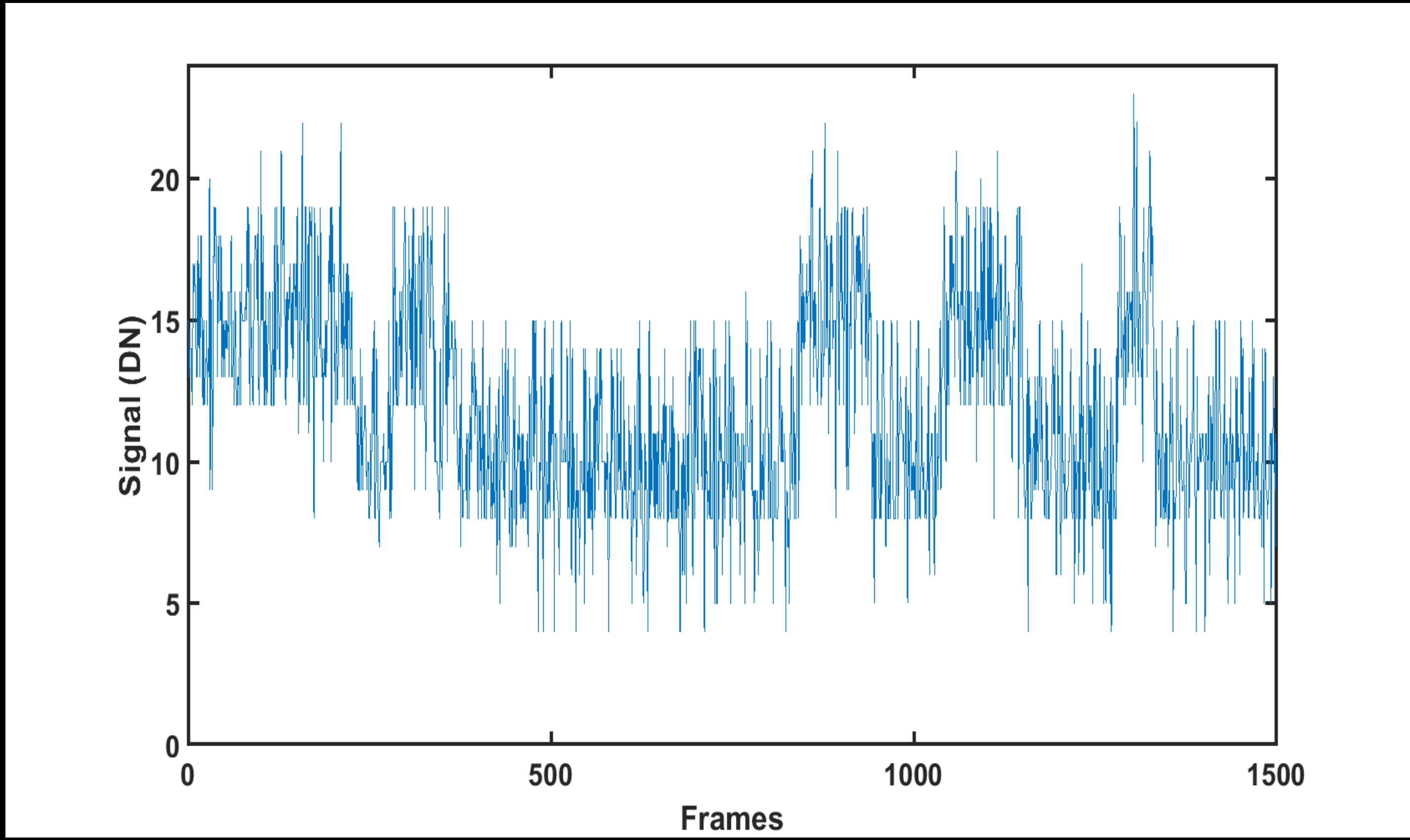


LET'S DO THE DANG THING!

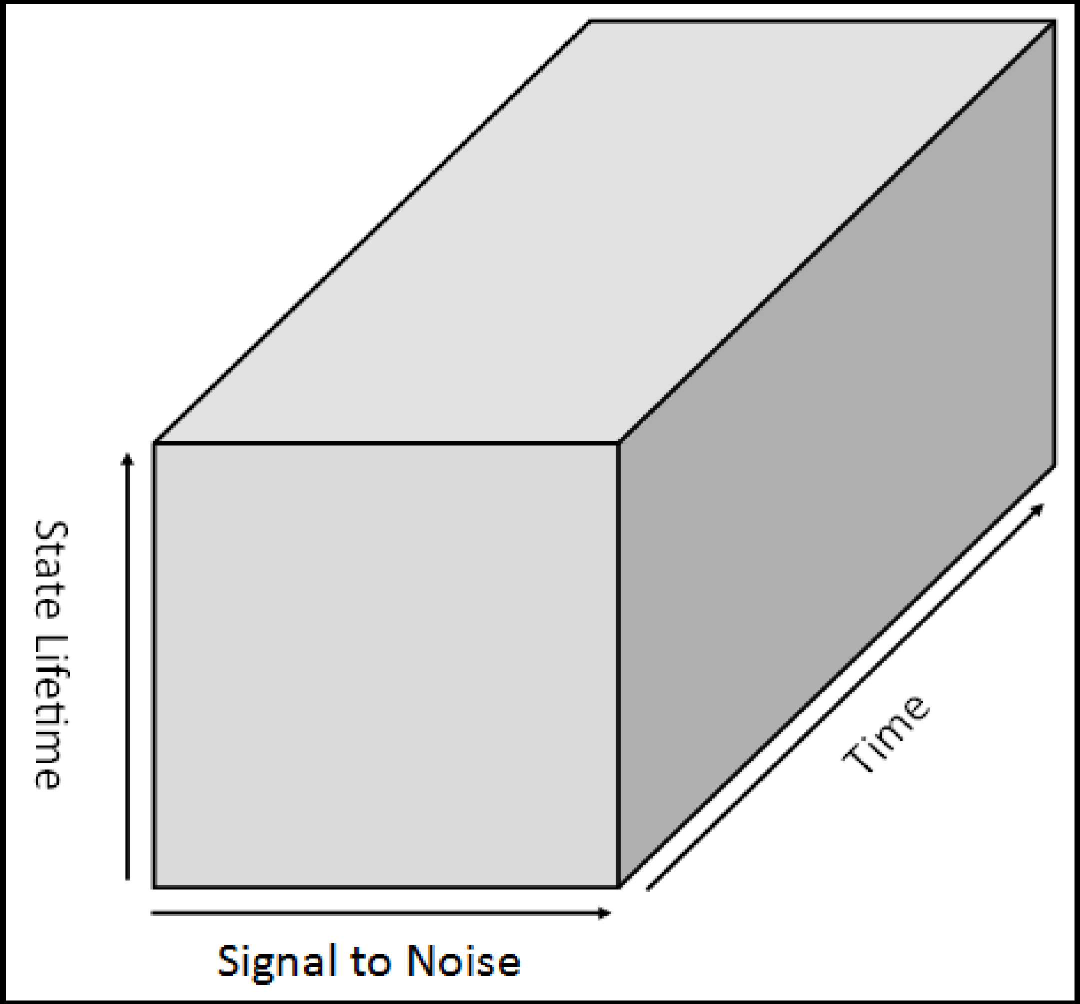
- Go to www.tensorflow.org
  - Learn -> TensorFlow
    - Tutorials -> Learn and Use ML -> Basic Classification
      - Run in Google CoLab (Uses a Jupyter Notebook environment)

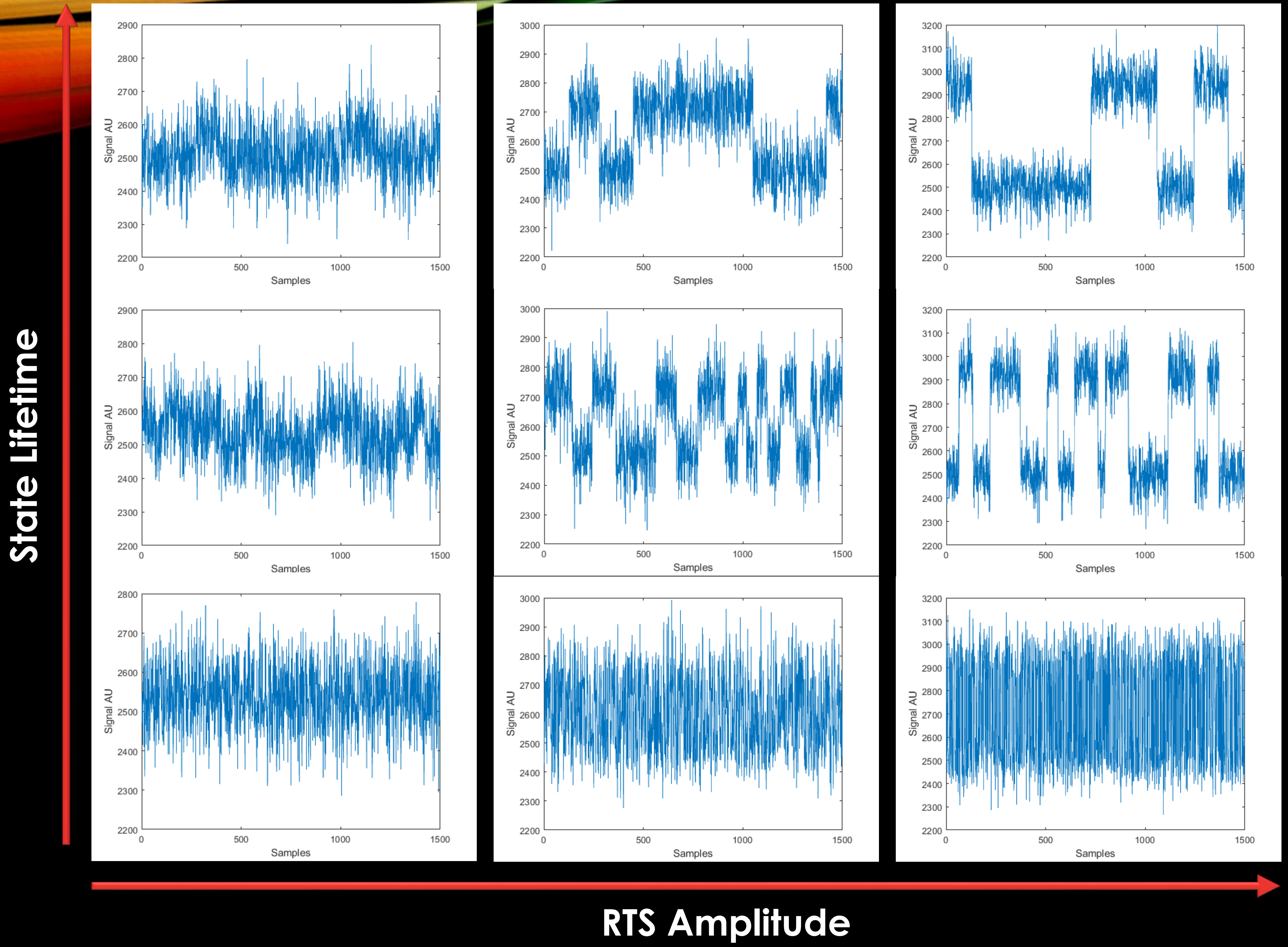


RANDOM TELEGRAPH SIGNAL DETECTION

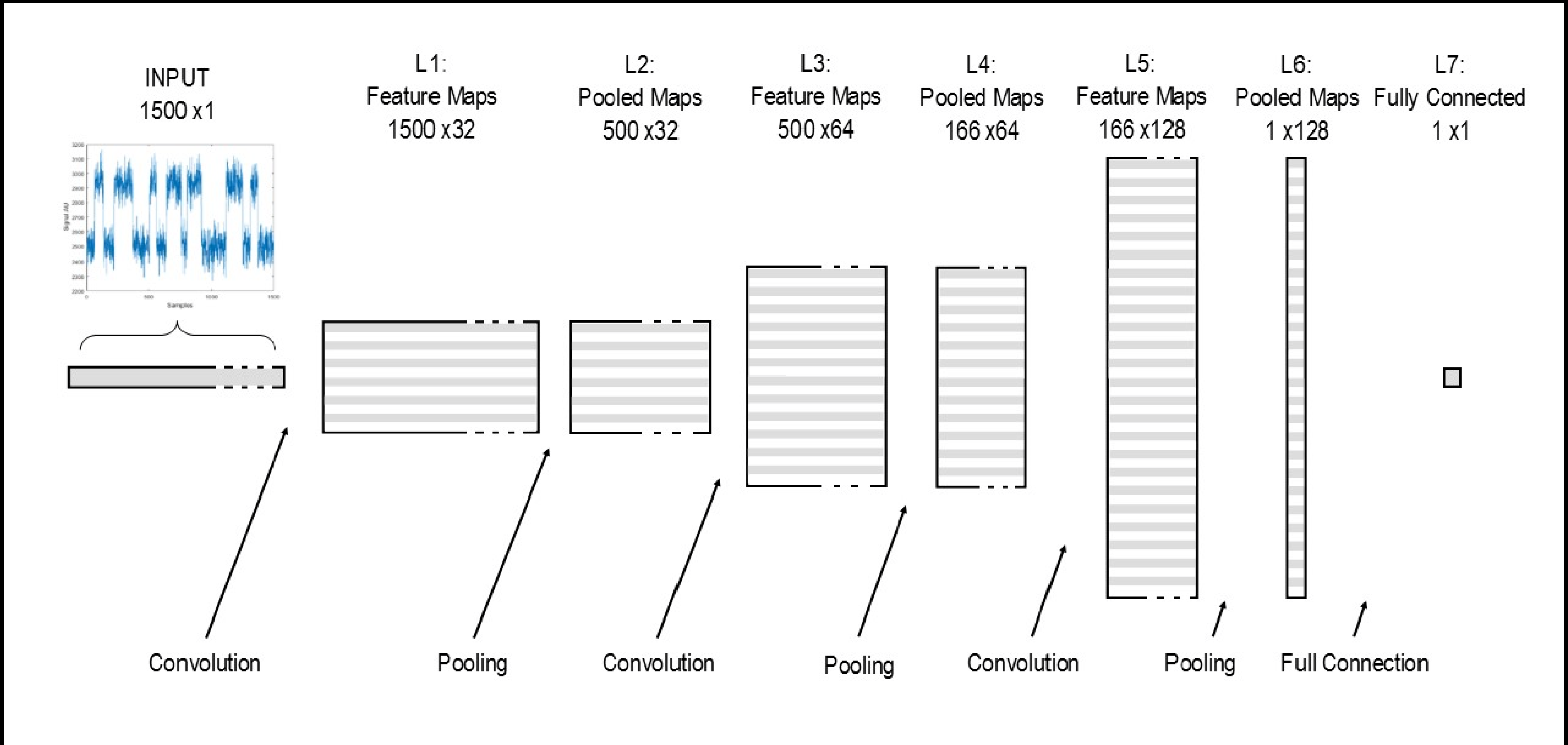


RANDOM TELEGRAPH SIGNAL DETECTION

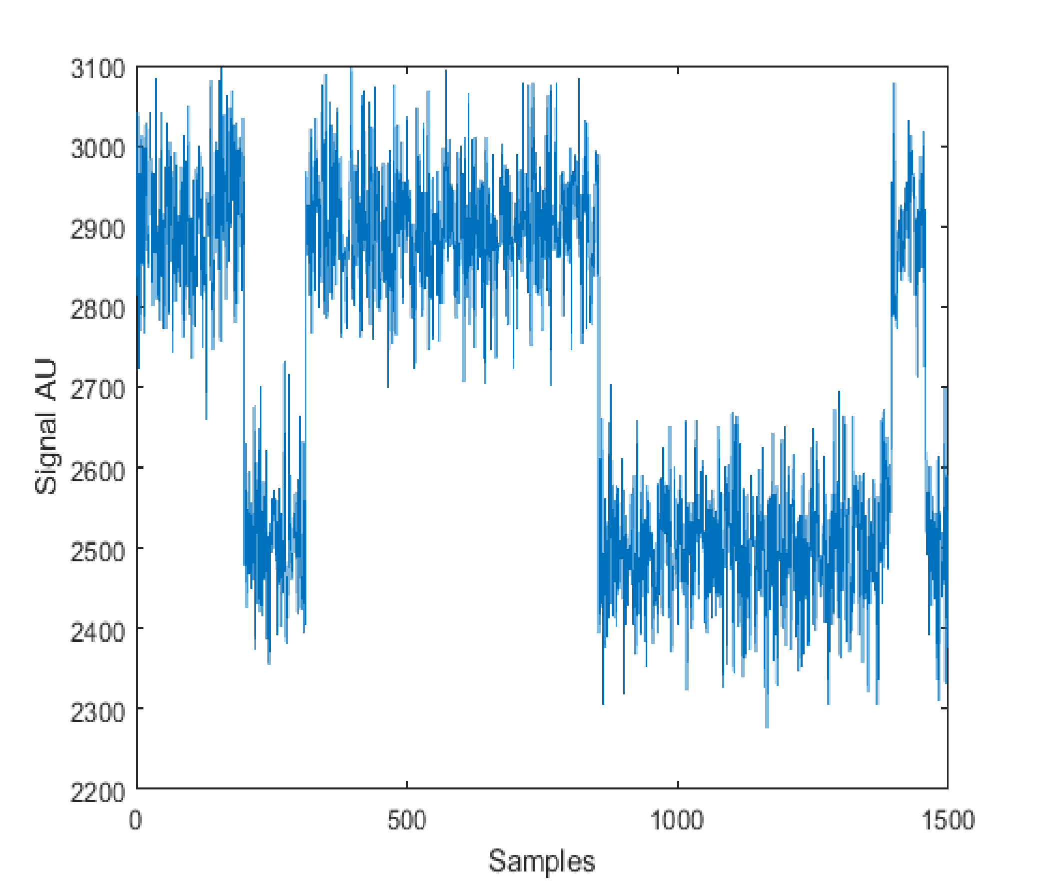


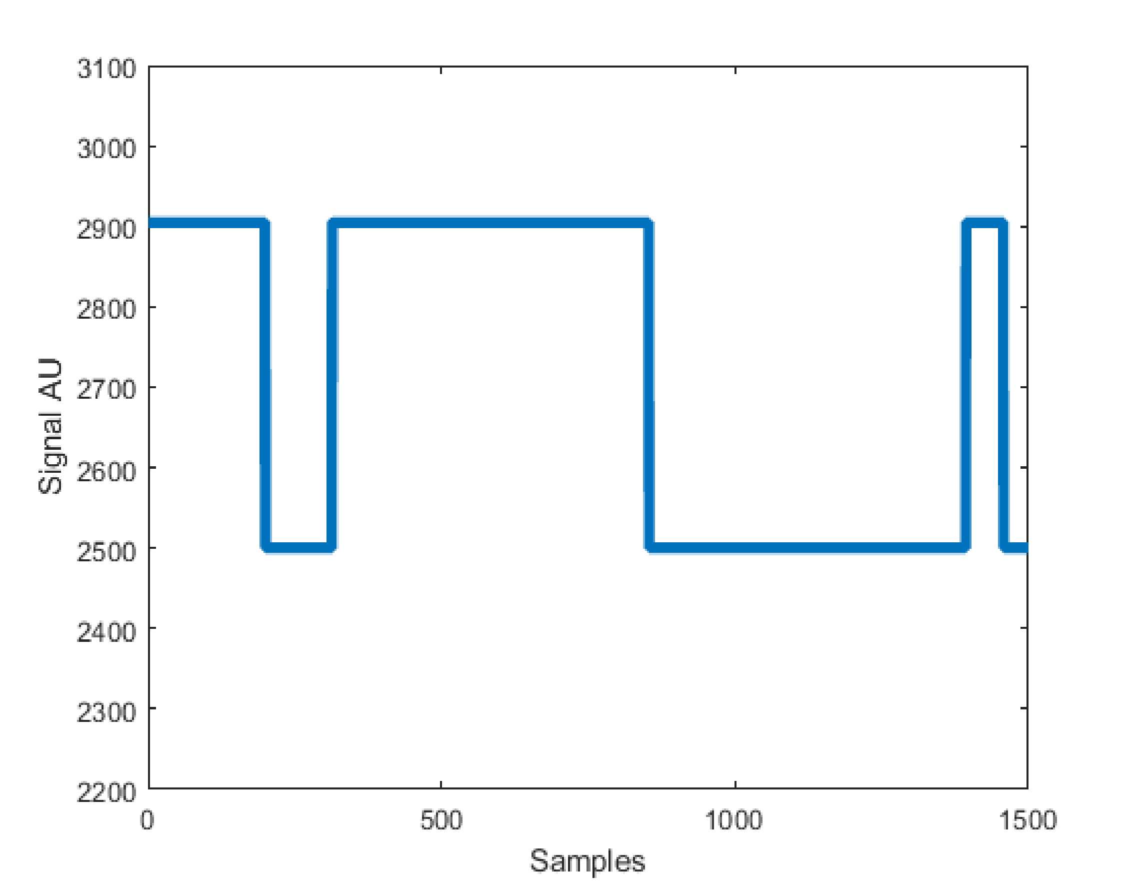

State Lifetime
RTS Amplitude
Signal AU
Samples

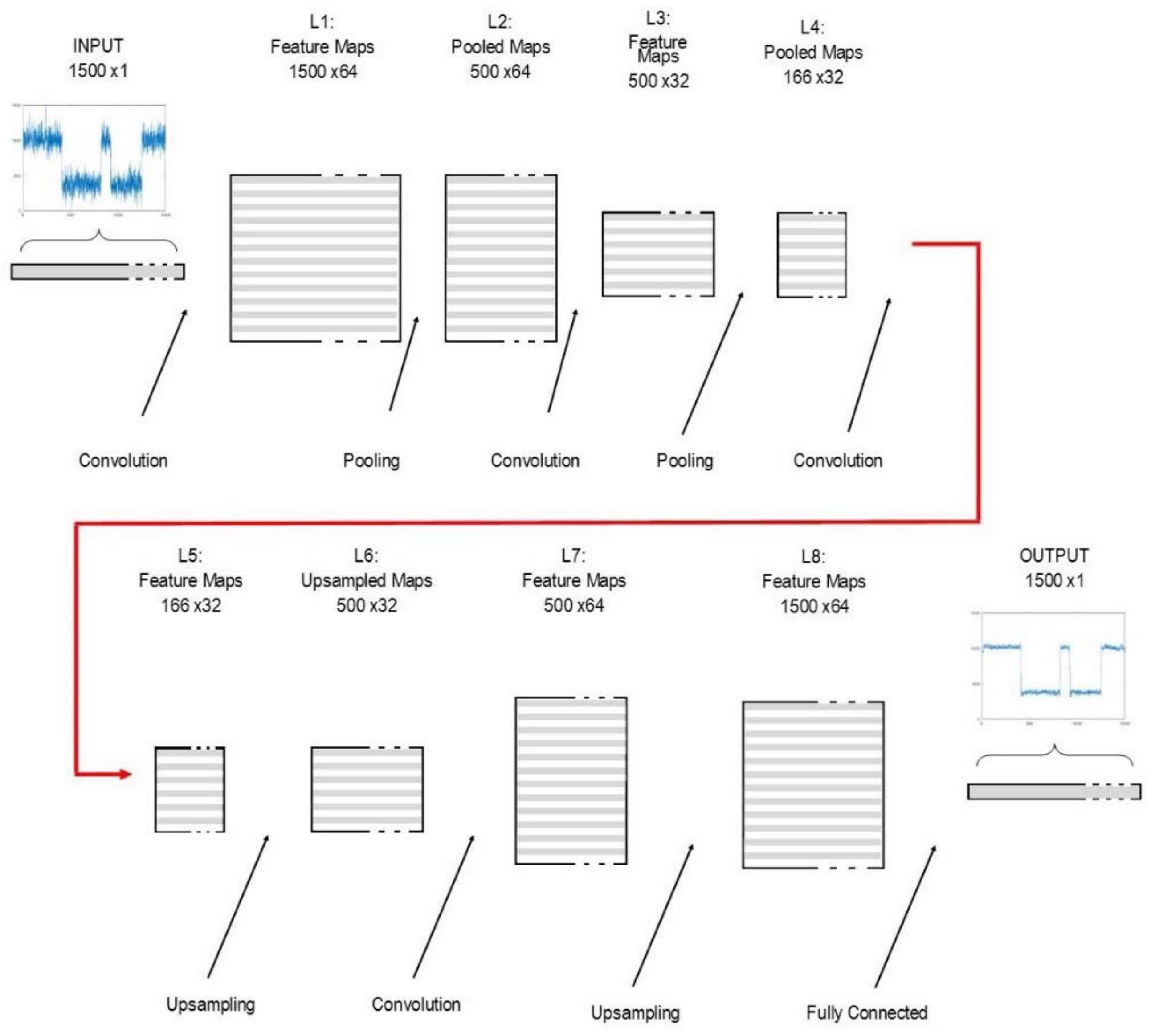
RANDOM TELEGRAPH SIGNAL DETECTION



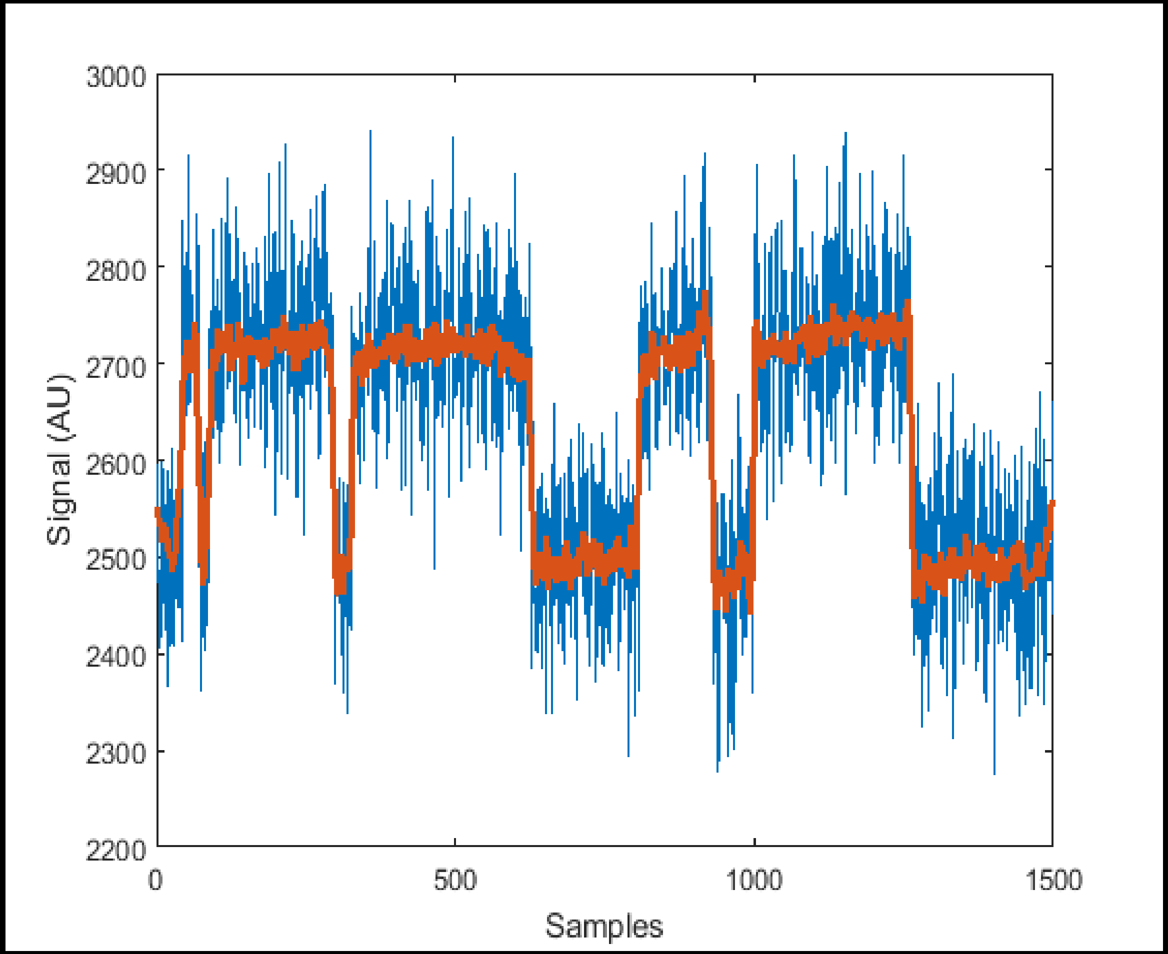
SIGNAL RECONSTRUCTION



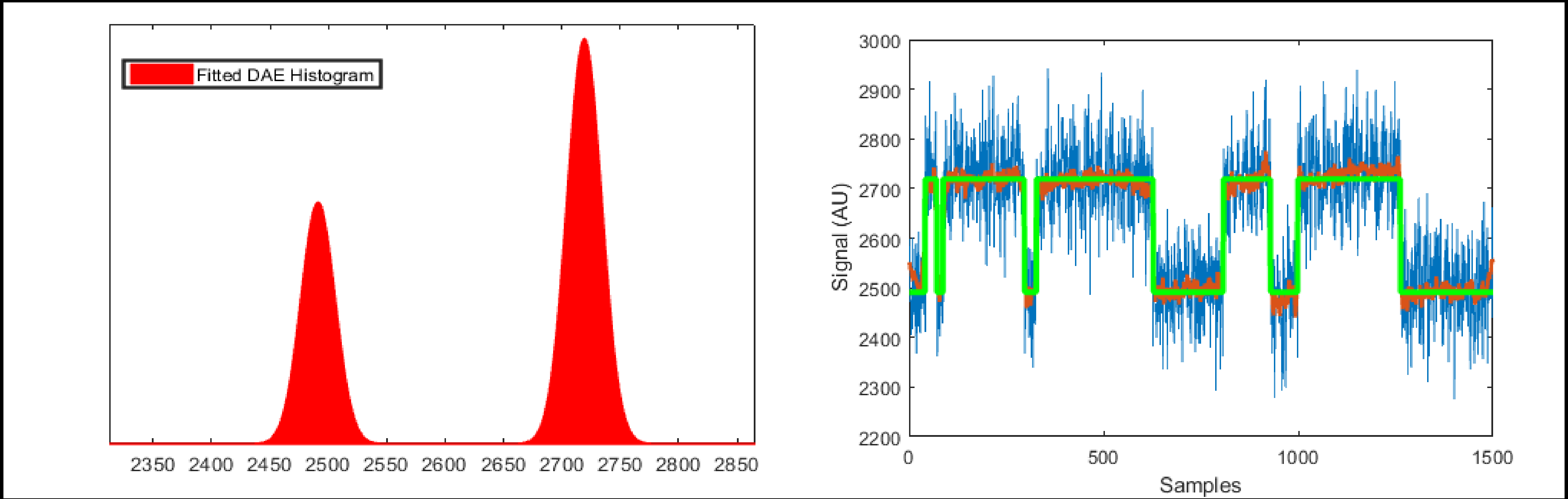



INPUT
1500 x1
L1:
Feature Maps
1500 x64
L2:
Pooled Maps
500 x64
L3:
Feature
Maps
500 x32
L4:
Pooled Maps
166 x32
Convolution
Pooling
Convolution
Pooling
Convolution
L5:
Feature Maps
166 x32
L6:
Upsampled Maps
500 x32
L7:
Feature Maps
500 x64
L8:
Feature Maps
1500 x64
OUTPUT
1500 x1
Upsampling
Convolution
Upsampling
Fully Connected

SIGNAL RECONSTRUCTION



SIGNAL RECONSTRUCTION



THANKS FOR COMING!