*Machine Learning in Ruby*

# Handwriting recognition

David Andrews

RYATTA GROUP

# Quick start: handwriting recognition

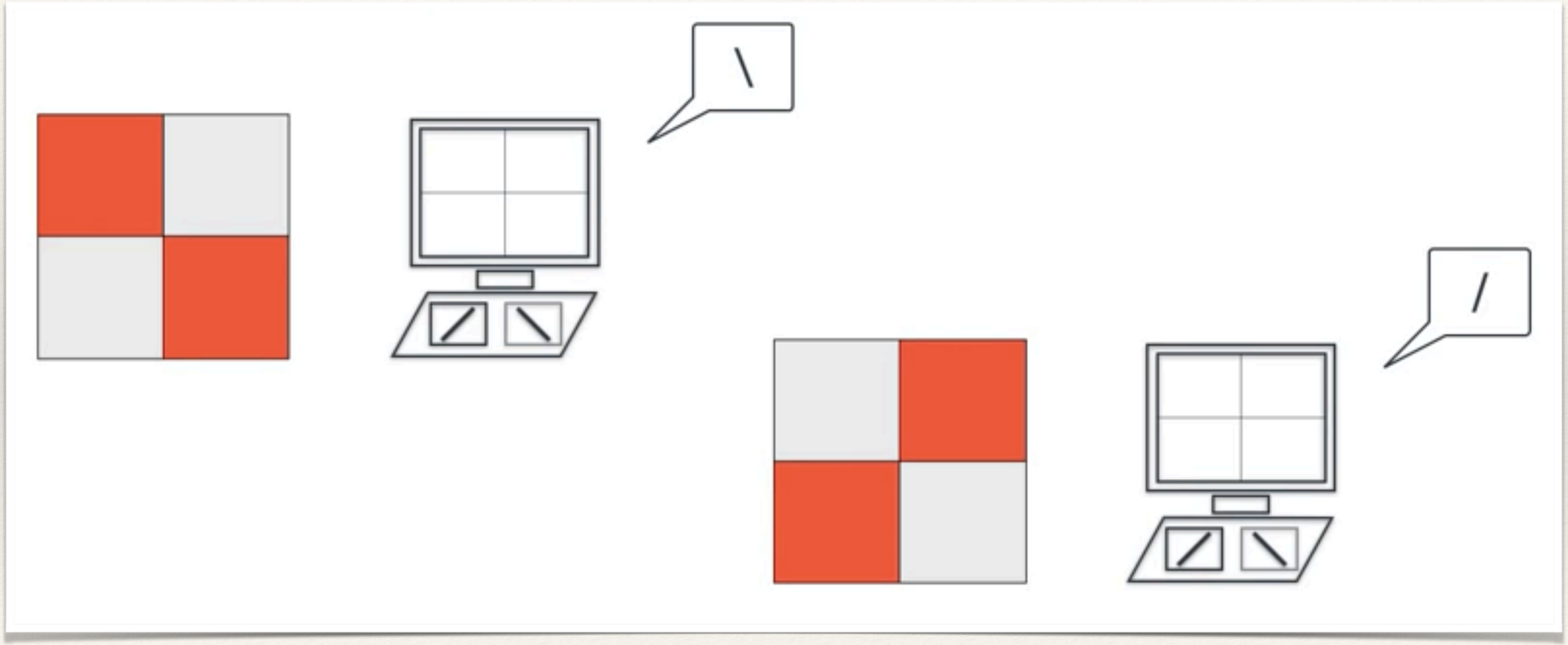



Image: Luis Serrano

# Quick start: images are vectorized

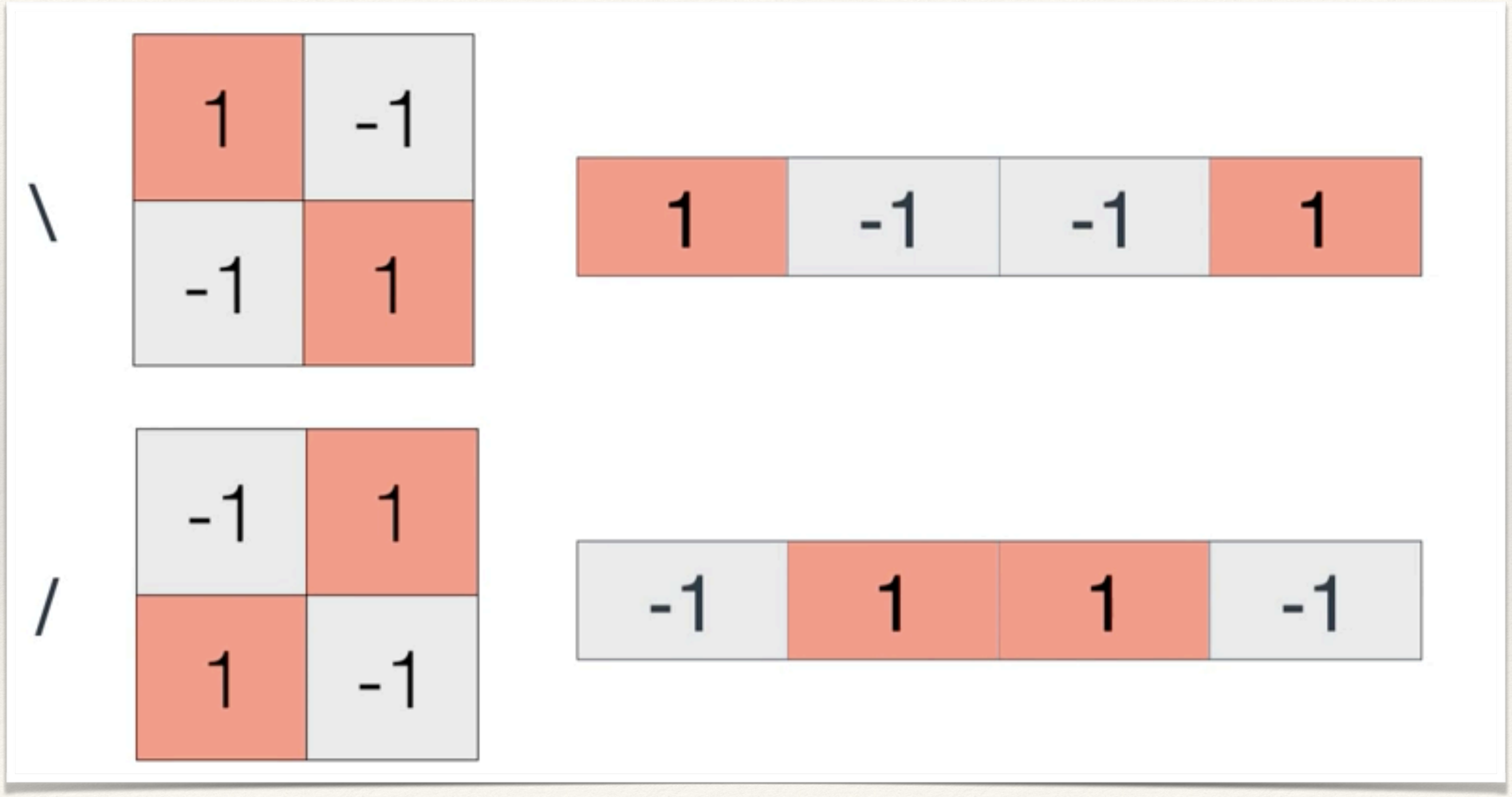



Image: Luis Serrano

# Quick start: features

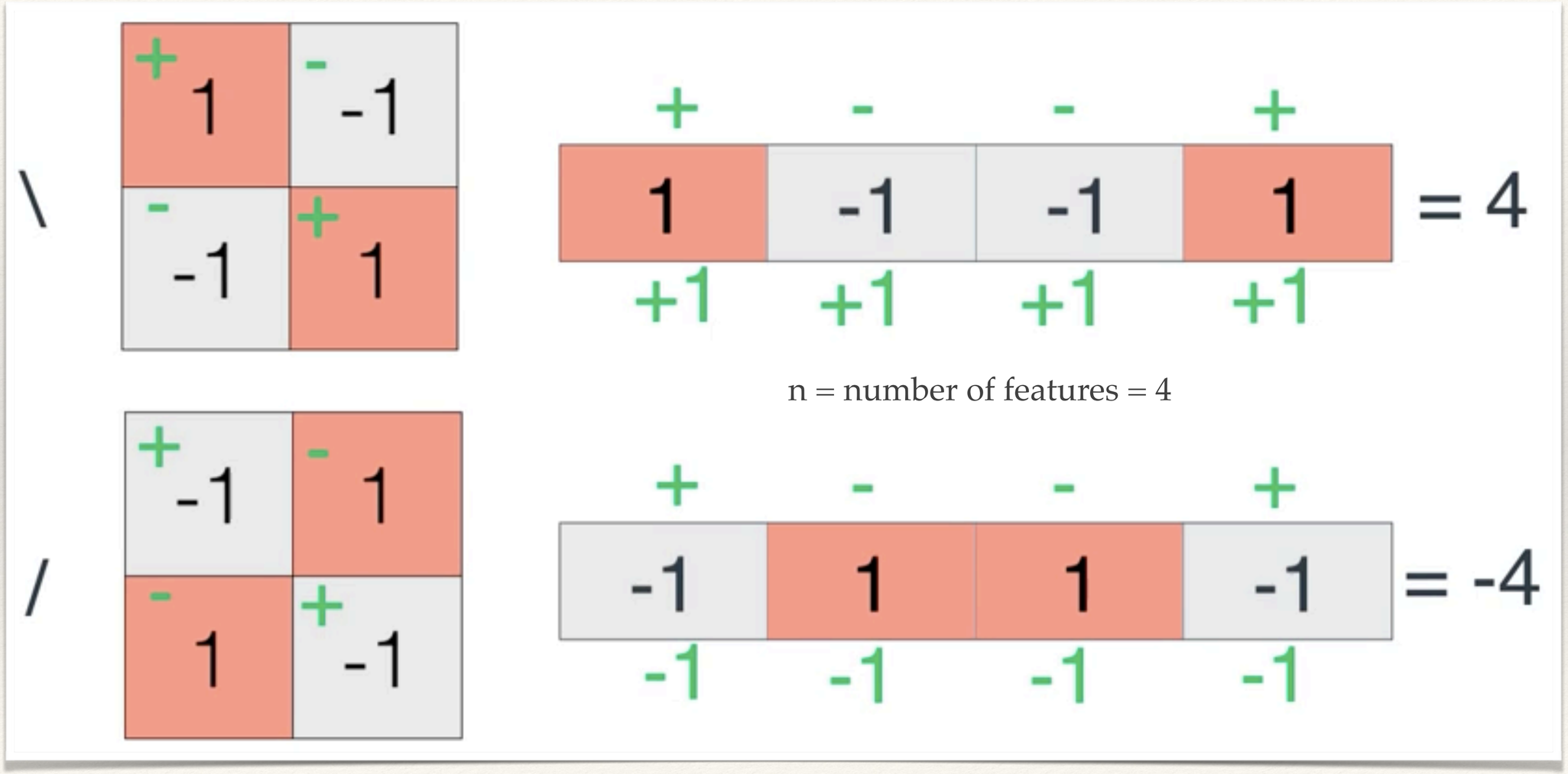



Image: Luis Serrano

# Quick start: simple classifier hypothesis

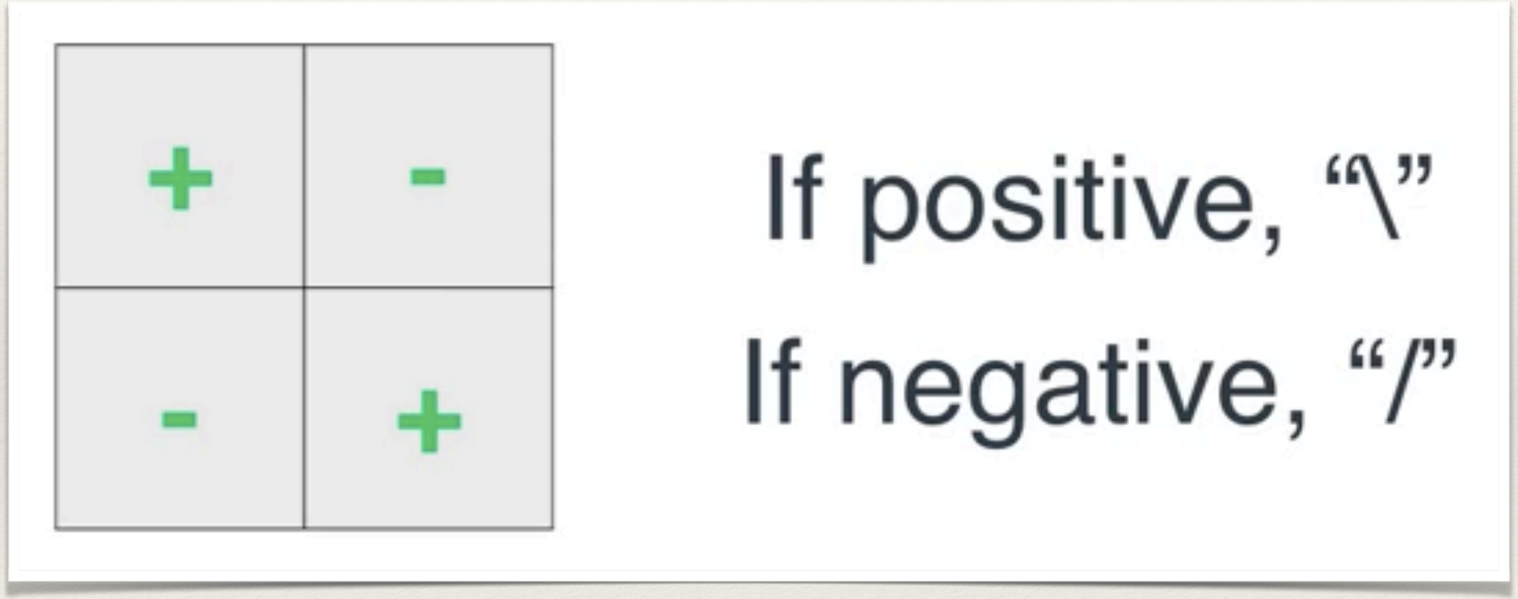



Image: Luis Serrano

# Quick start: sloppier examples

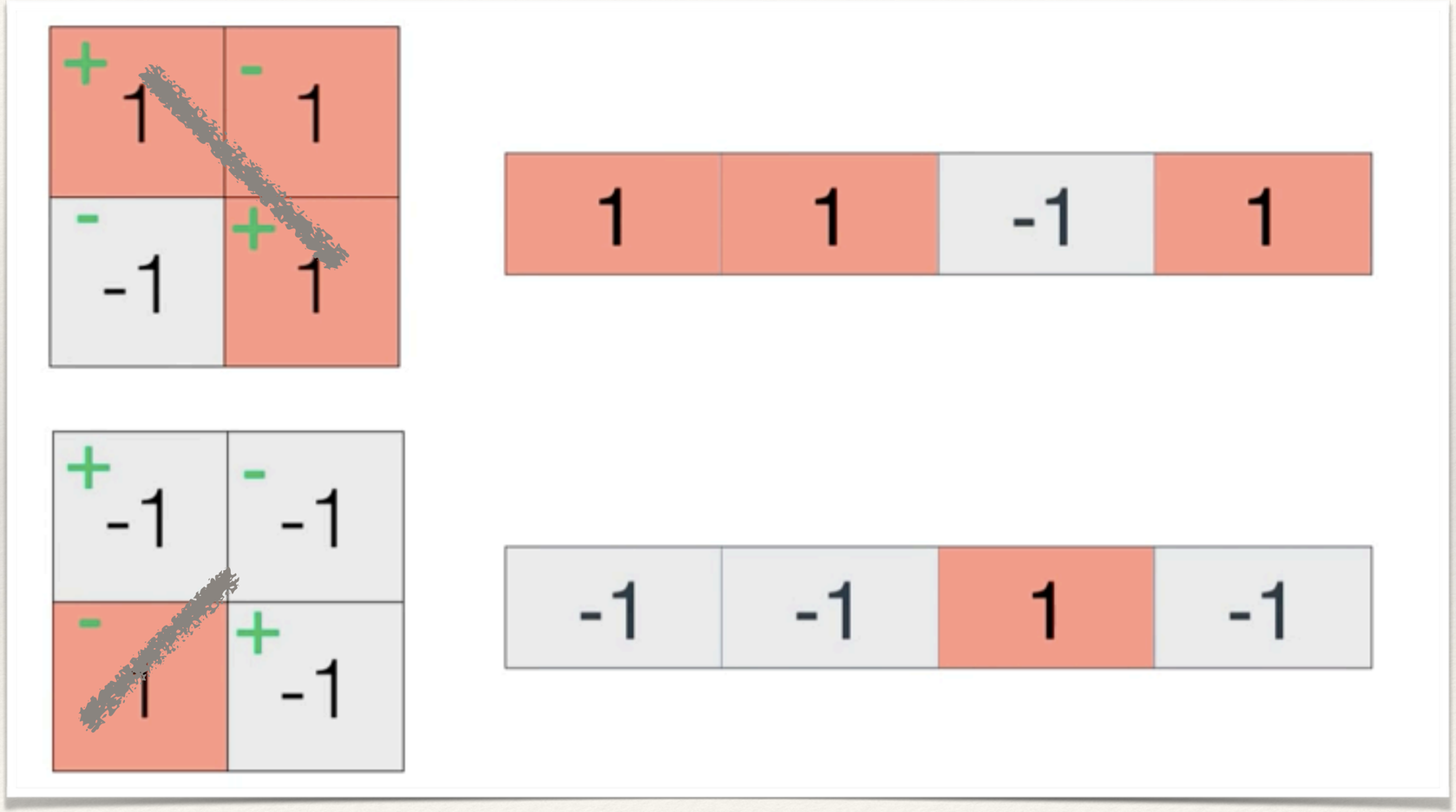



Image: Luis Serrano

# Exit Ramp 1

- If you are new to Ruby and/or have limited knowledge of calculus and algebra, you may wish to implement a simple classifier as we have defined here.
- Here's some Ruby to get you started:

```
theta = [1, -1, -1 , 1]
forward_slash = [-1, 1, 1, -1]
back_slash = [1, -1, -1, 1]
example = forward_slash
h = theta.zip(example).map { |te, ee| te * ee }.inject(0) { |sum, e| sum + e }
```

- Create a .rb file and rework the above so you can submit different examples and evaluate your hypothesis. Once you have that working, have the code output its best guess for the character, based on the input "image".

# Supporting more complexity

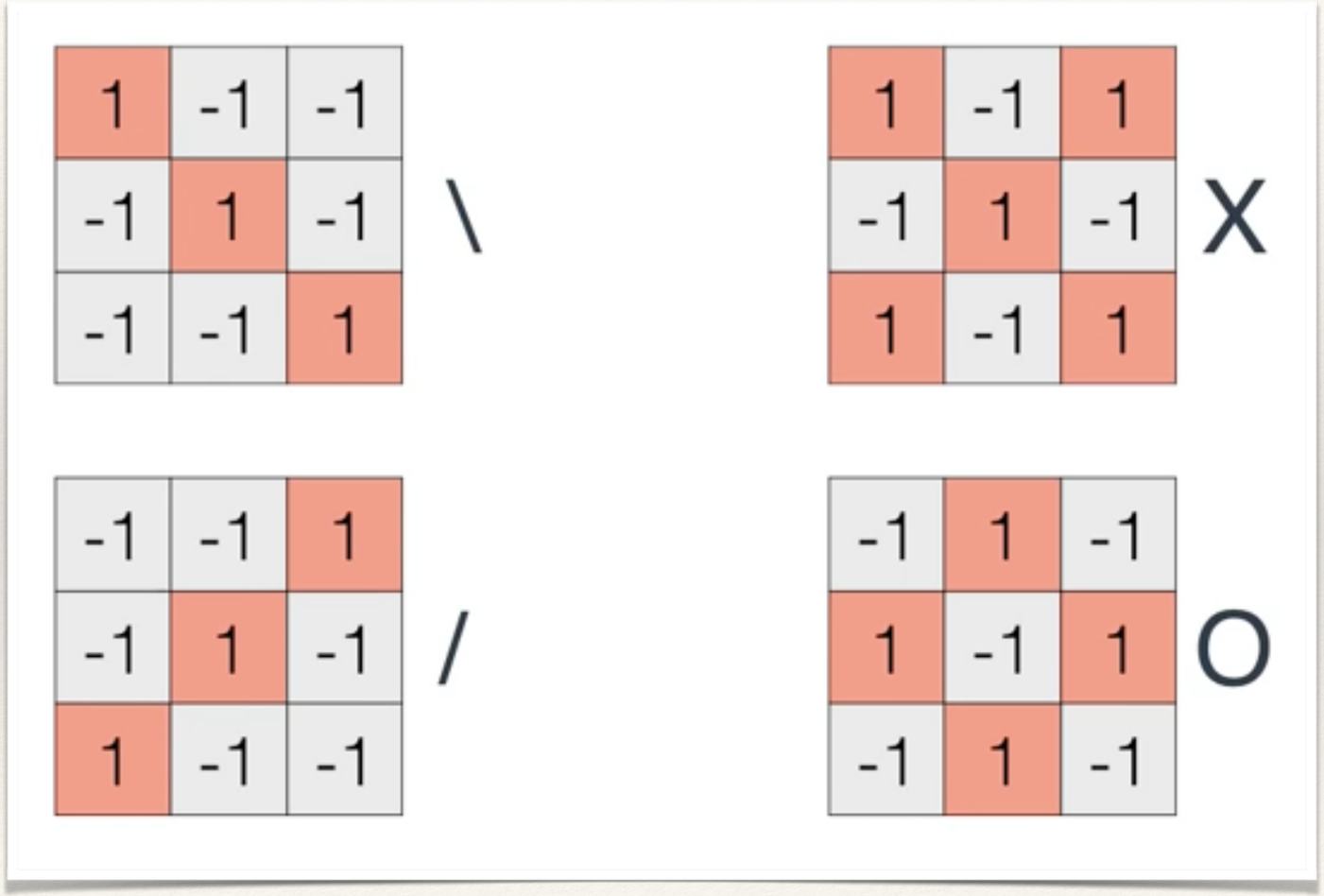



Image: Luis Serrano

# What we know

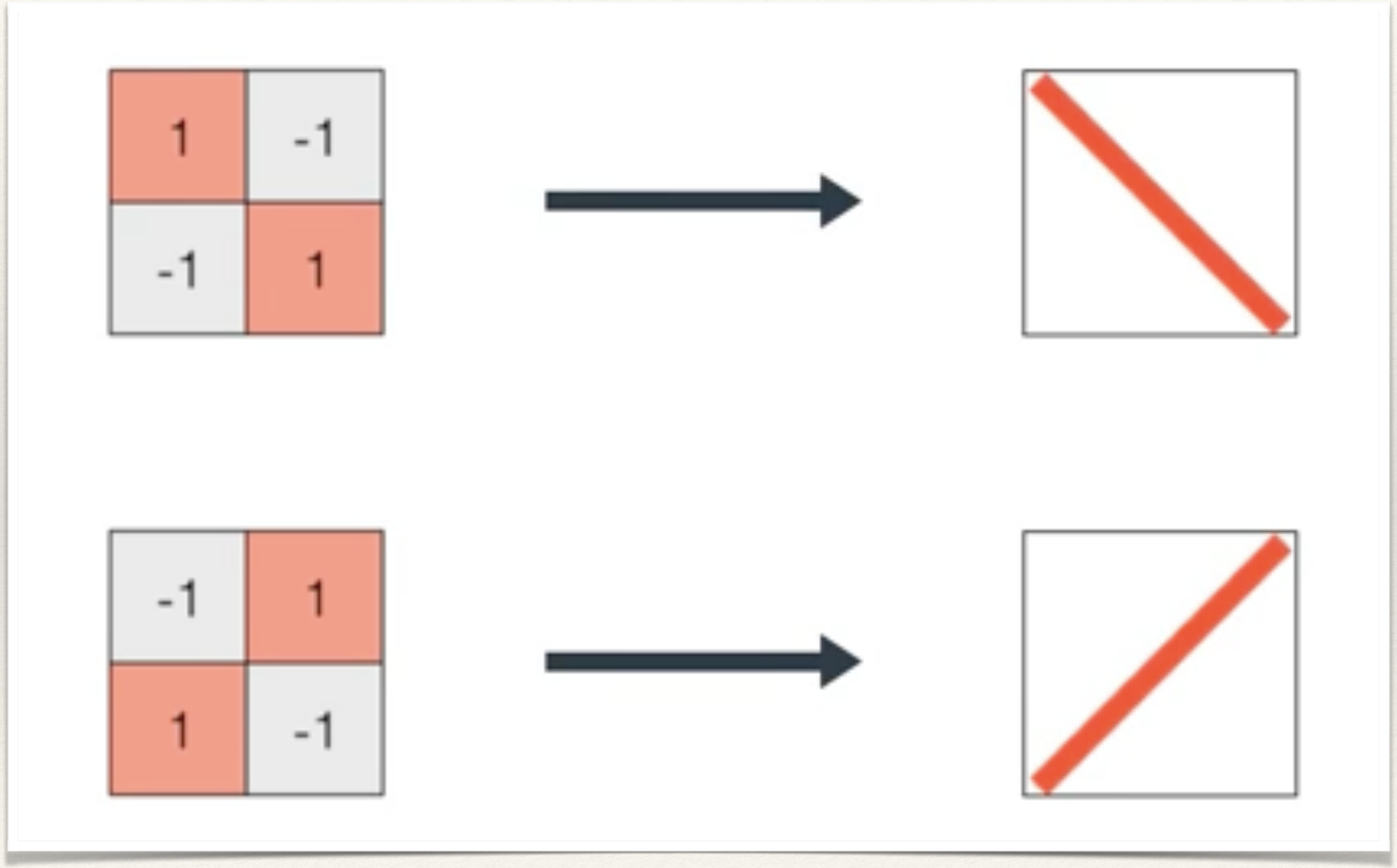



Image: Luis Serrano

# Applying filters

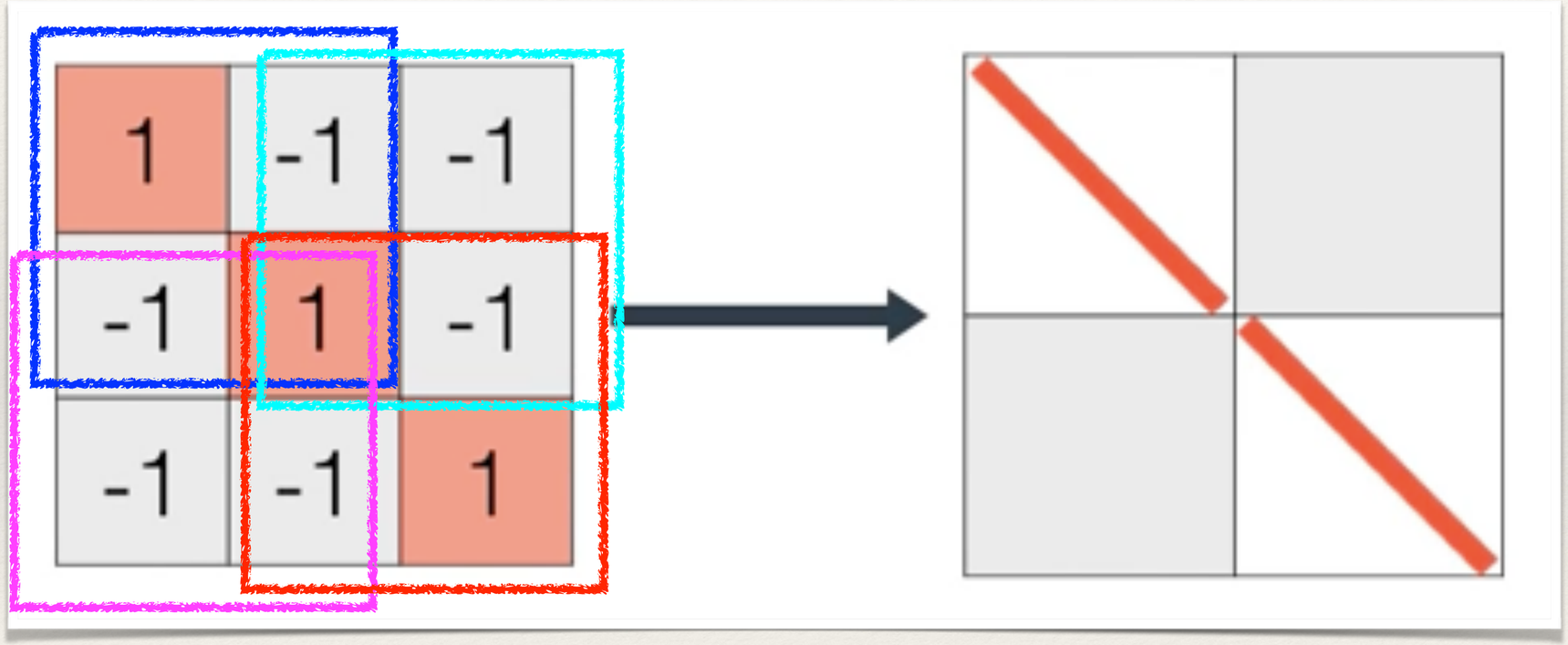



Image: Luis Serrano

# Convolutional Neural Network

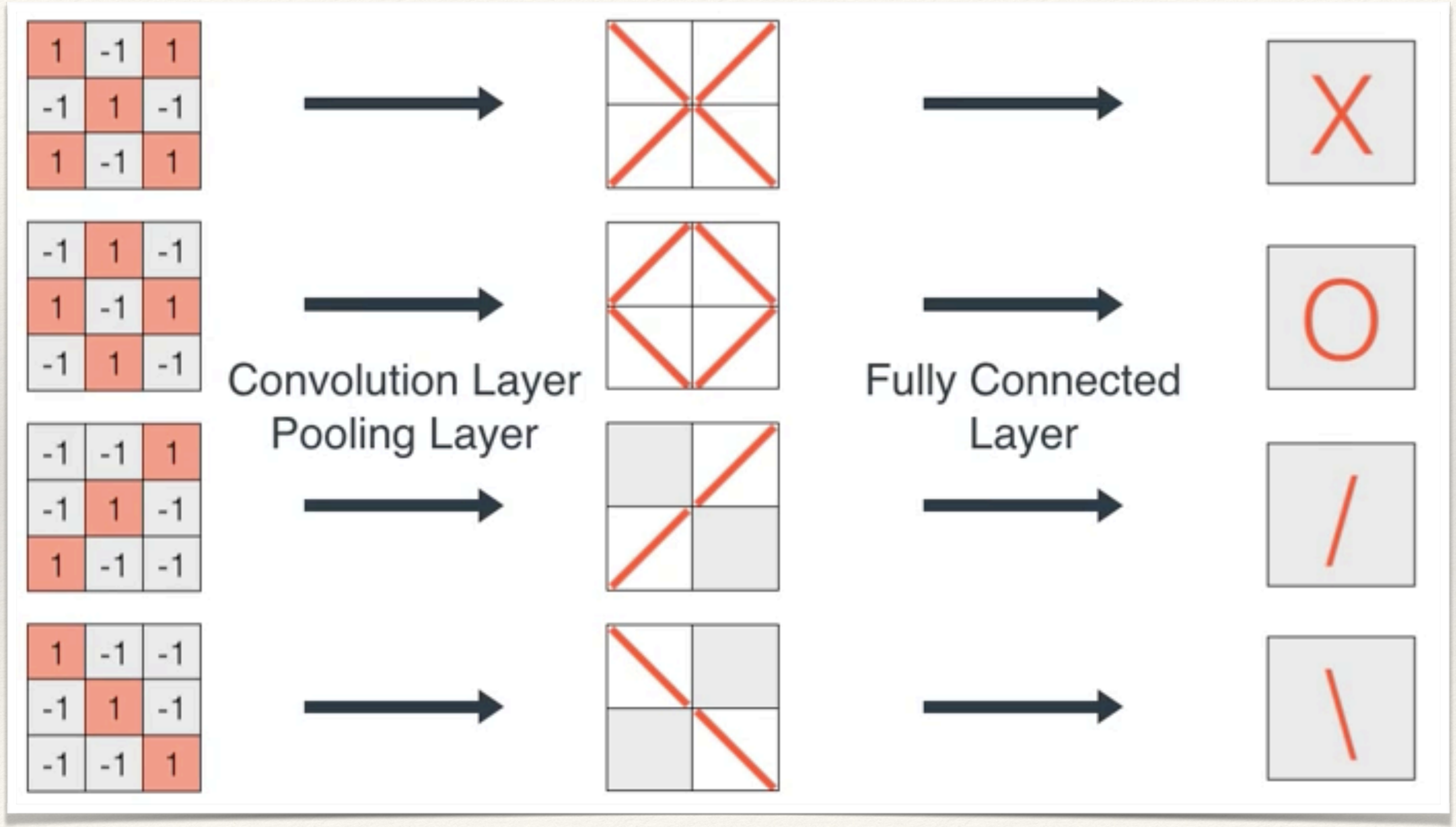



Image: Luis Serrano

# Recognizing a "\"

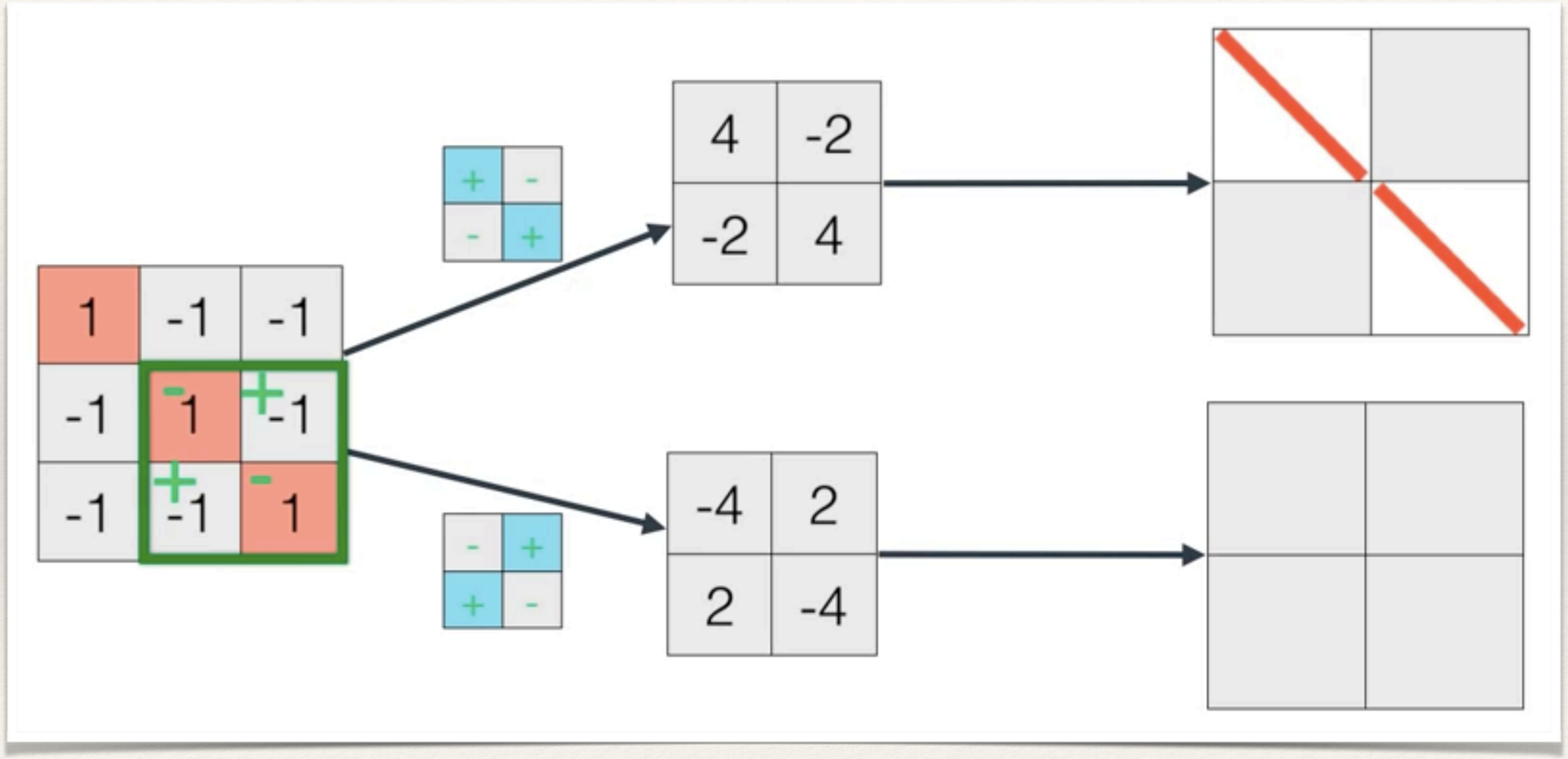
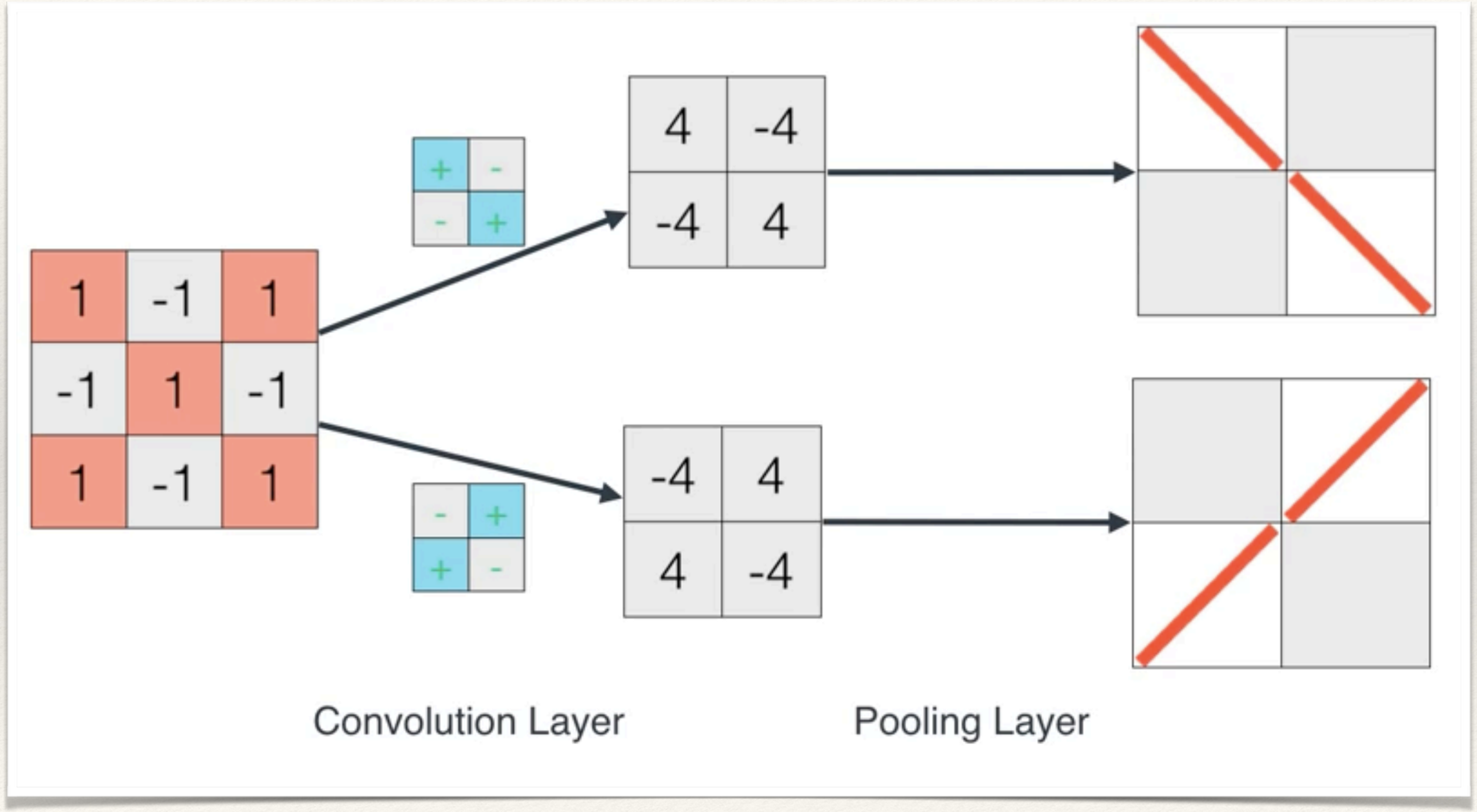



Image: Luis Serrano

# Recognizing an "X"



# Fully connected layer

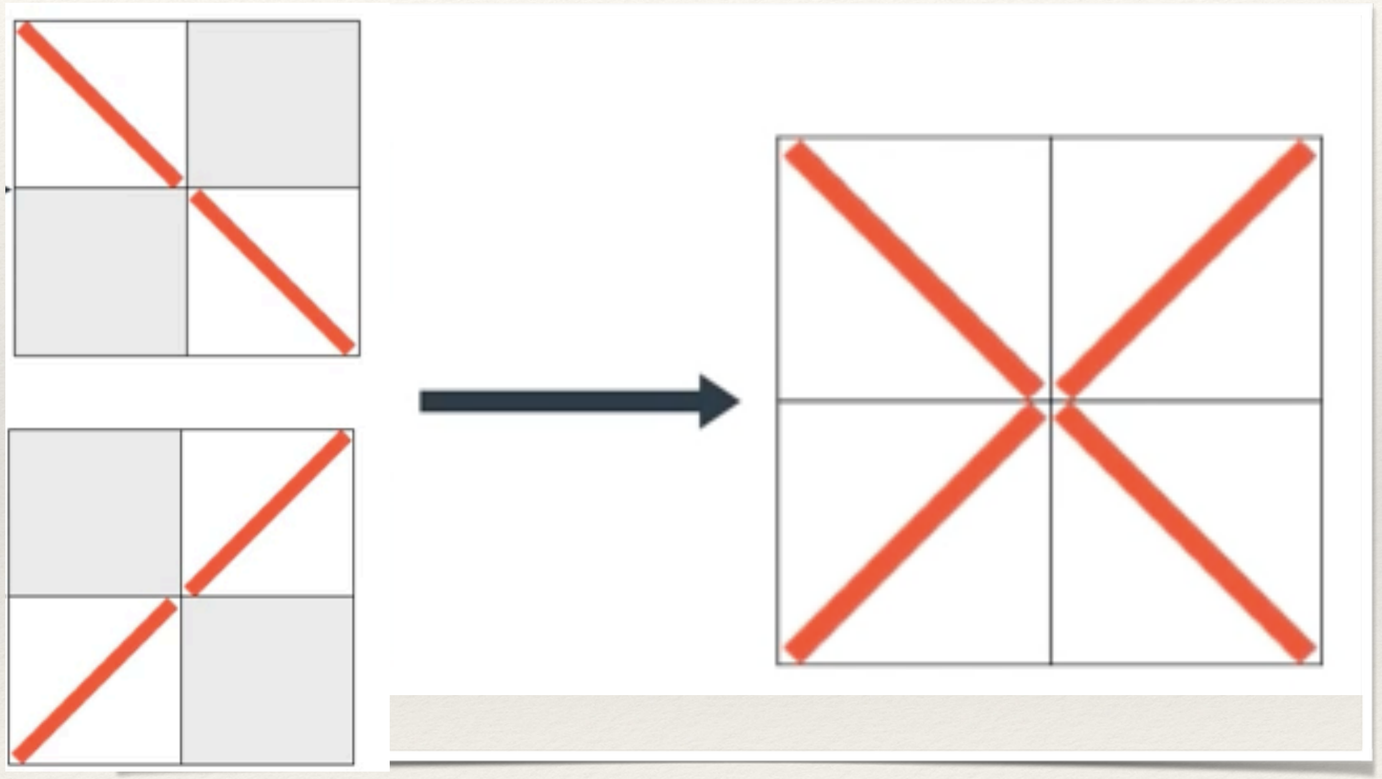


Image: Luis Serrano

# Recognizing an "O"

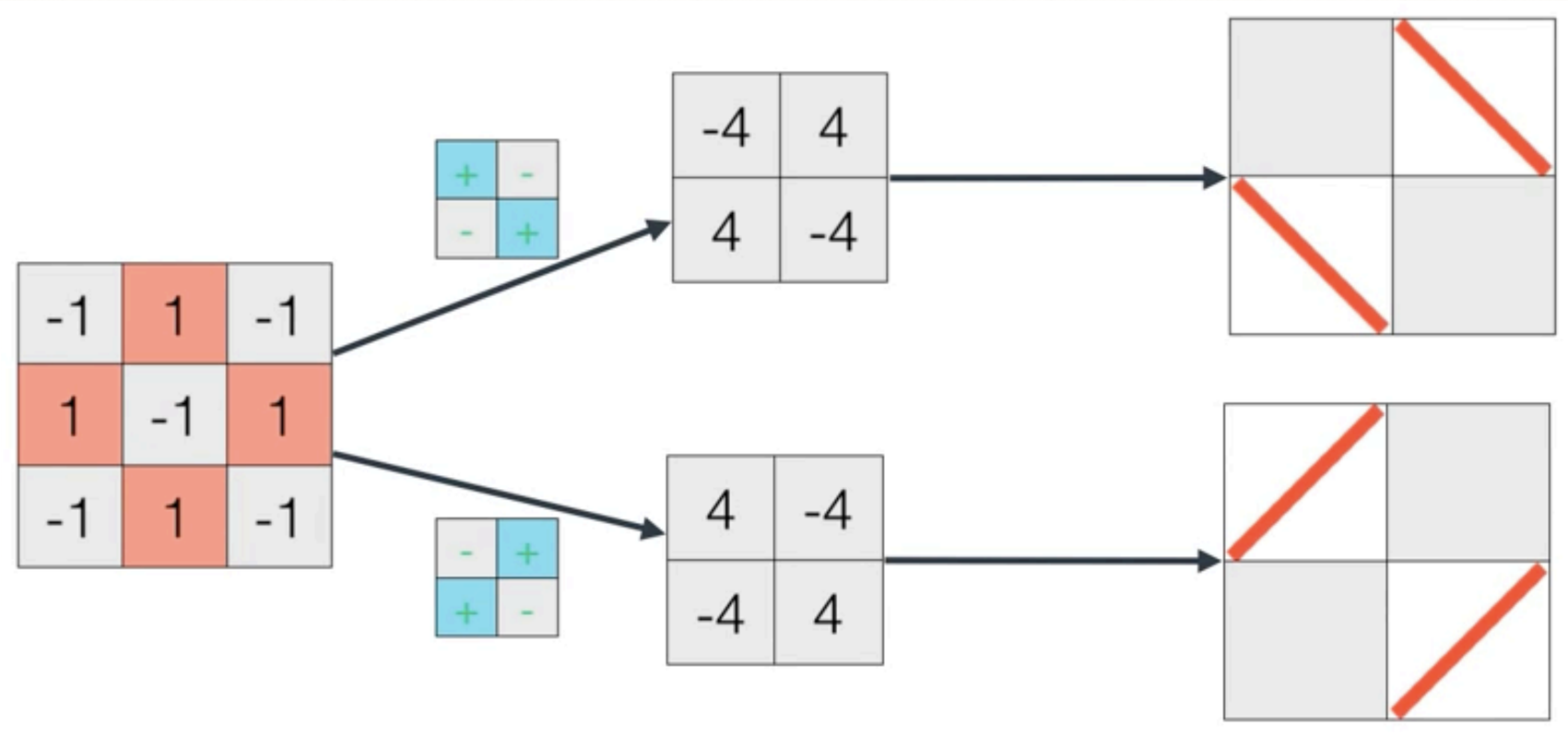



Image: Luis Serrano

# Fully connected layer

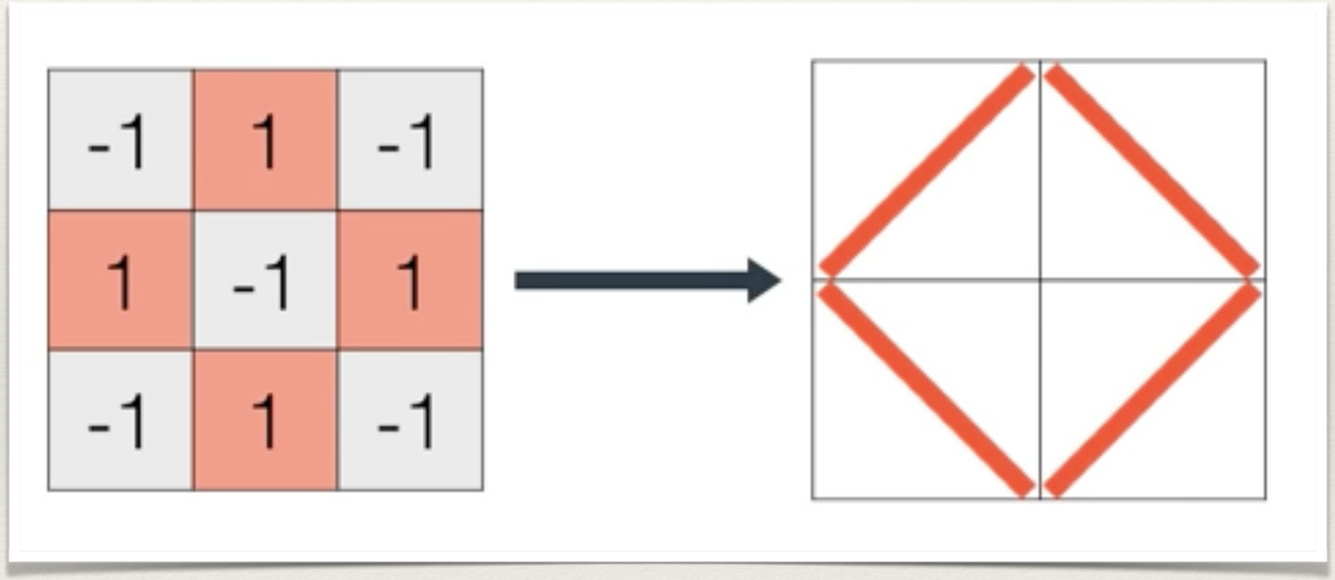



Image: Luis Serrano

# Fully connected layer filters

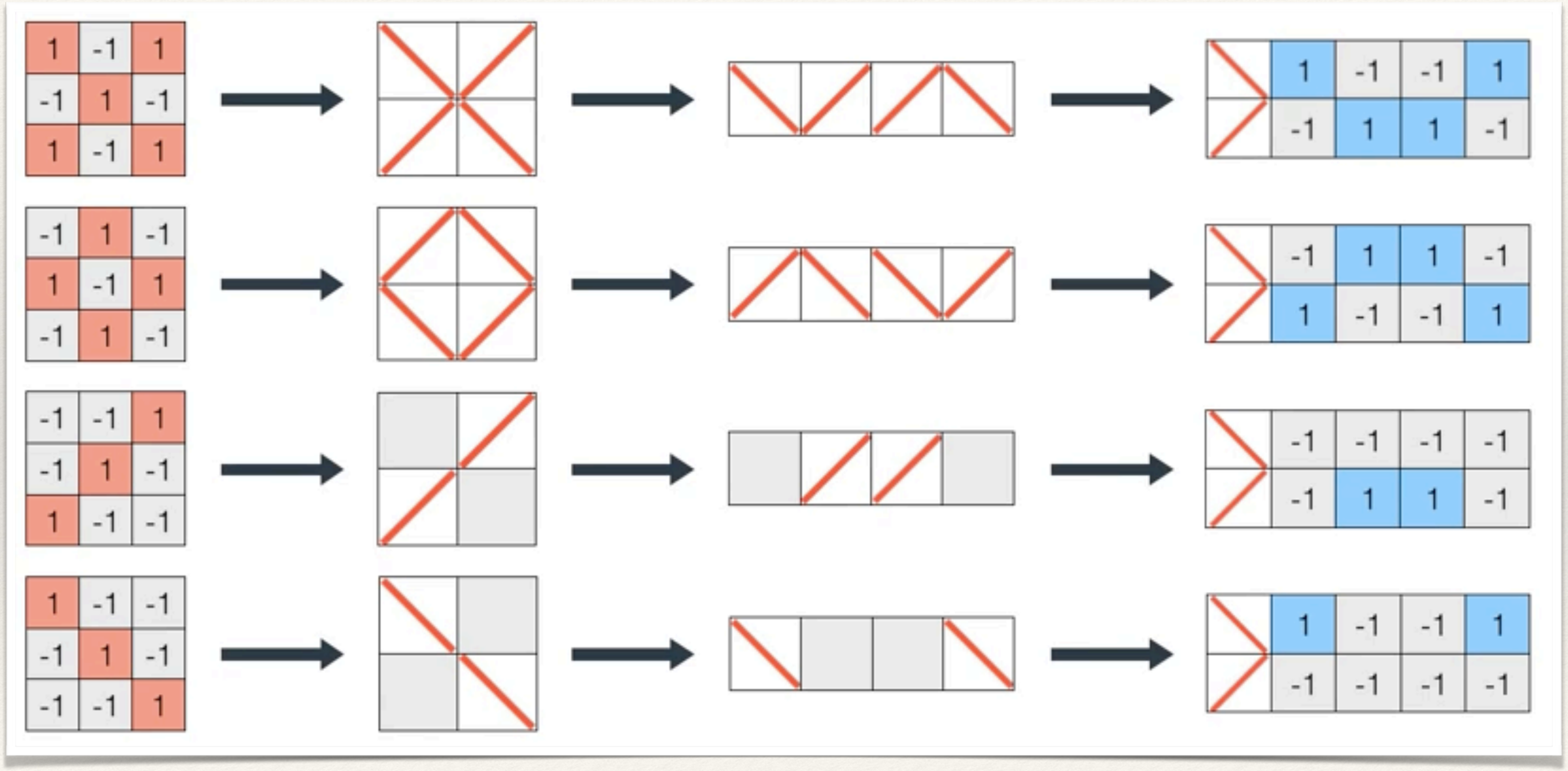



Image: Luis Serrano

# Fully connected layer filters


Image: Luis Serrano

# Fully connected scoring

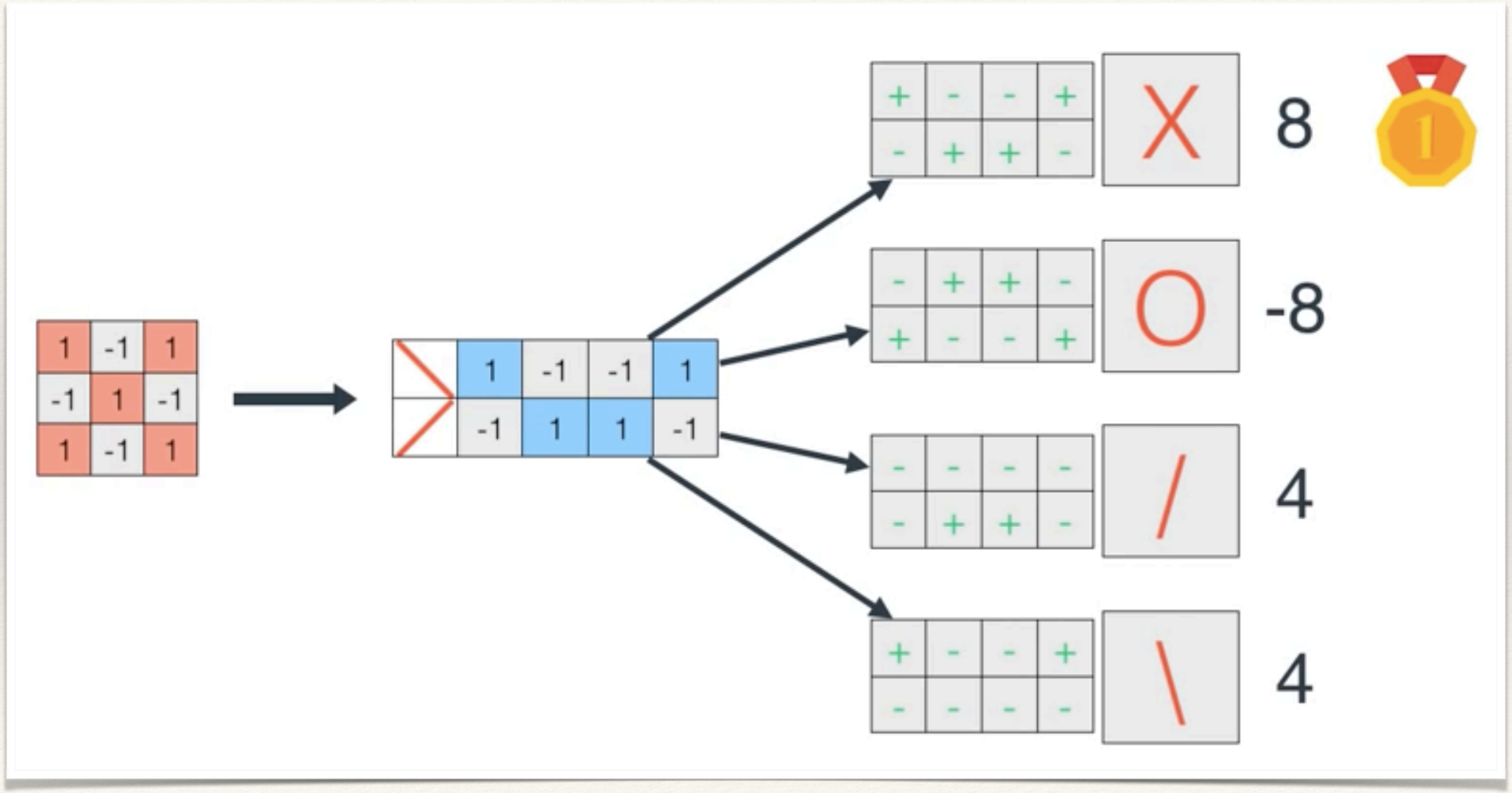



Image: Luis Serrano

# Scoring an "X"

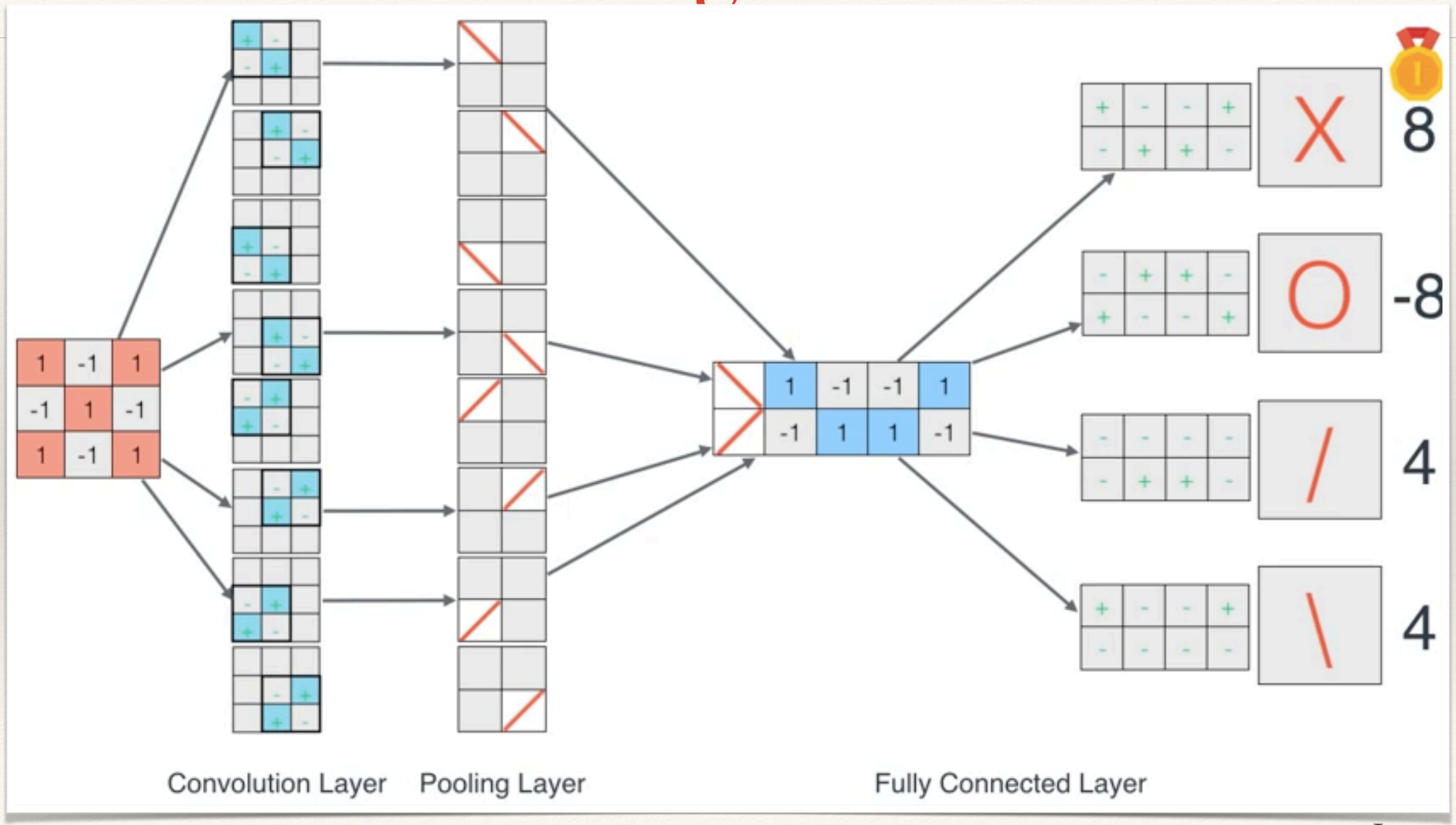



Image: Luis Serrano

# A full Convulutional Neural Network

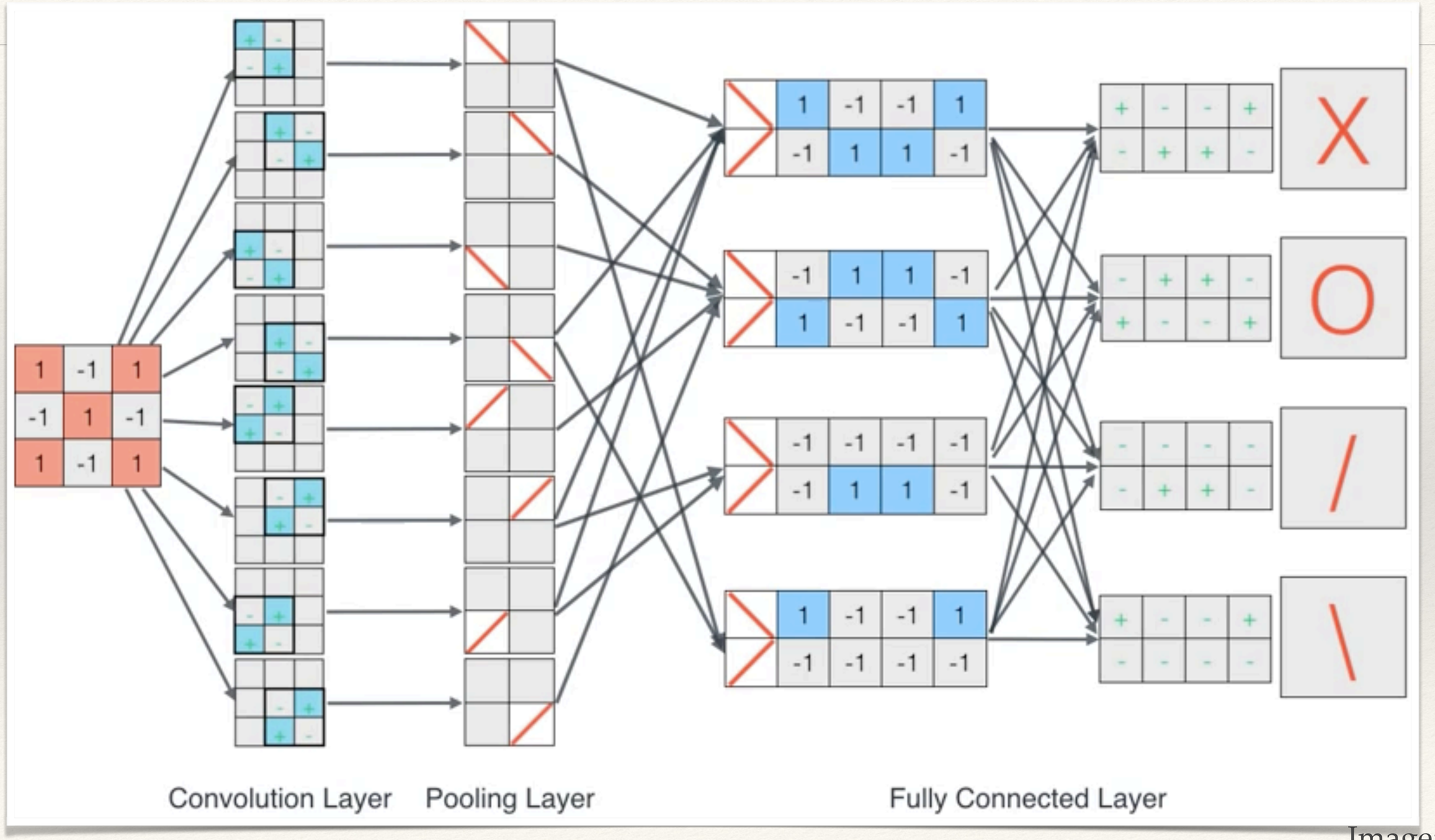



Image: Luis Serrano

# Exit Ramp 2

- If you want a bigger challenge, create a classifier as we have defined here.
- You will need to implement the three layers and various filters to make the neural network identify the images.
- Create some .rb scripts and combine them into a fourth neural_net.rb application. The script should accept 3x3 input “images” and output the final scores for each of the possible symbols.

# More complex examples

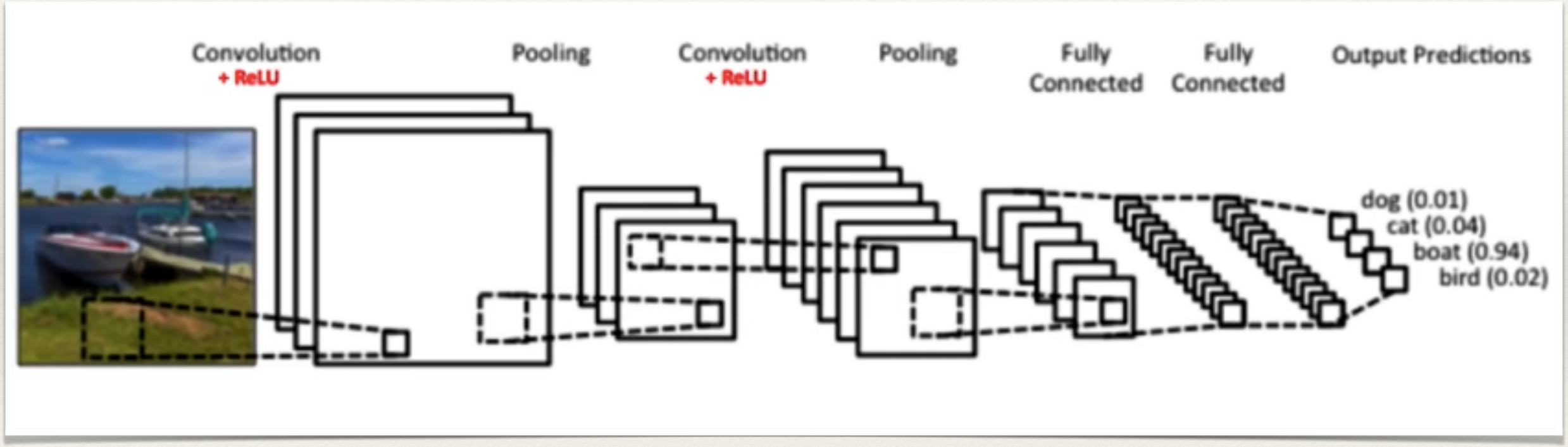
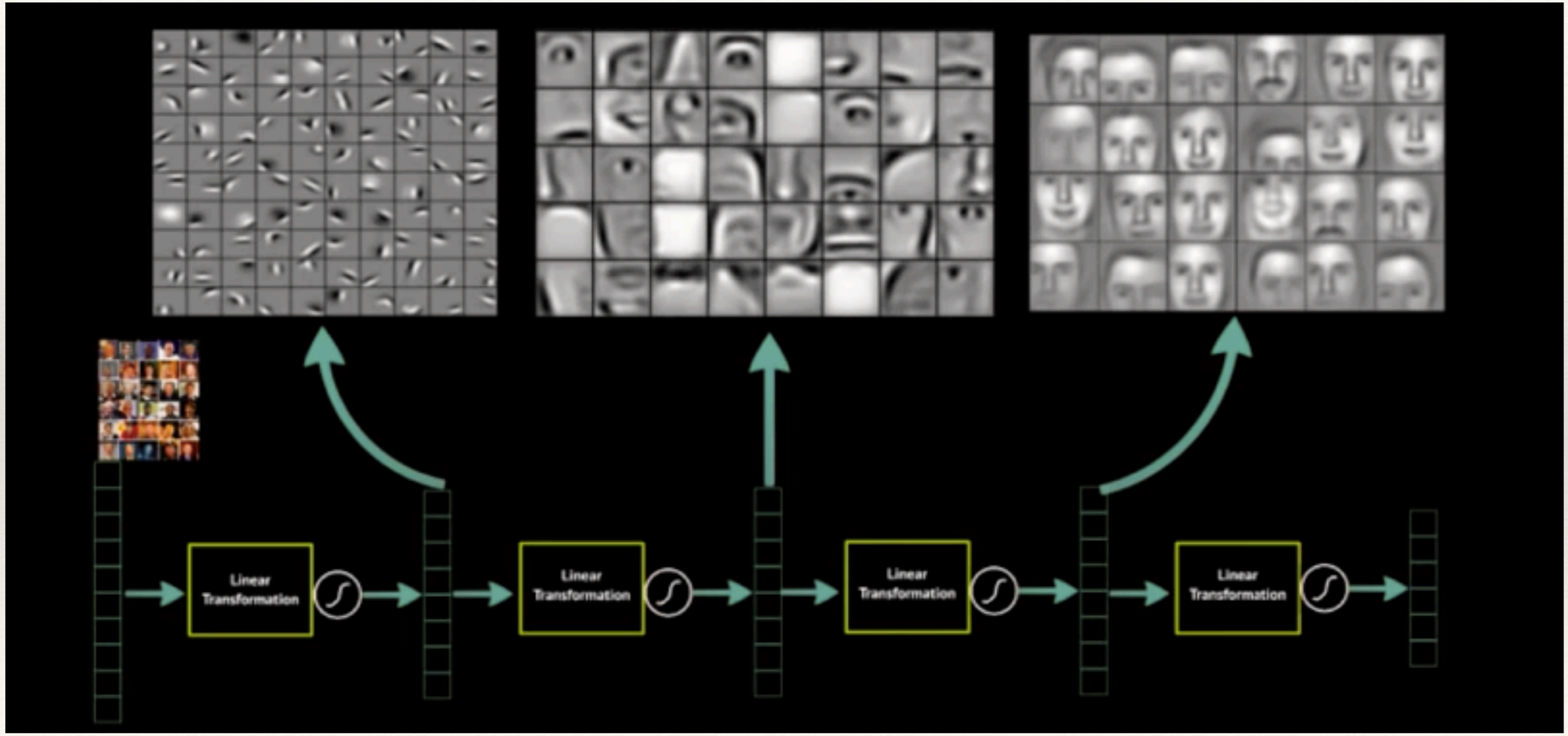
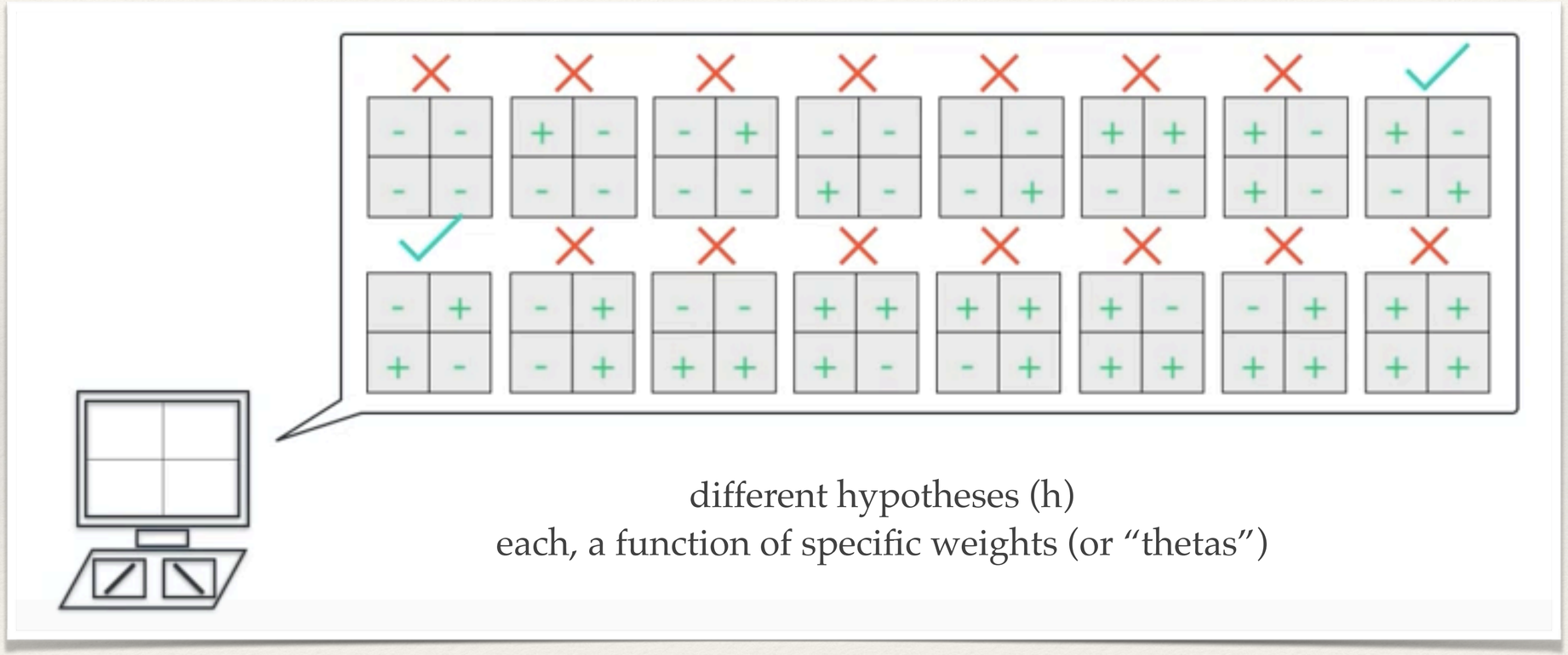



Image: Luis Serrano

# Facial recognition



# Machine learning



different hypotheses (h)
each, a function of specific weights (or “thetas”)

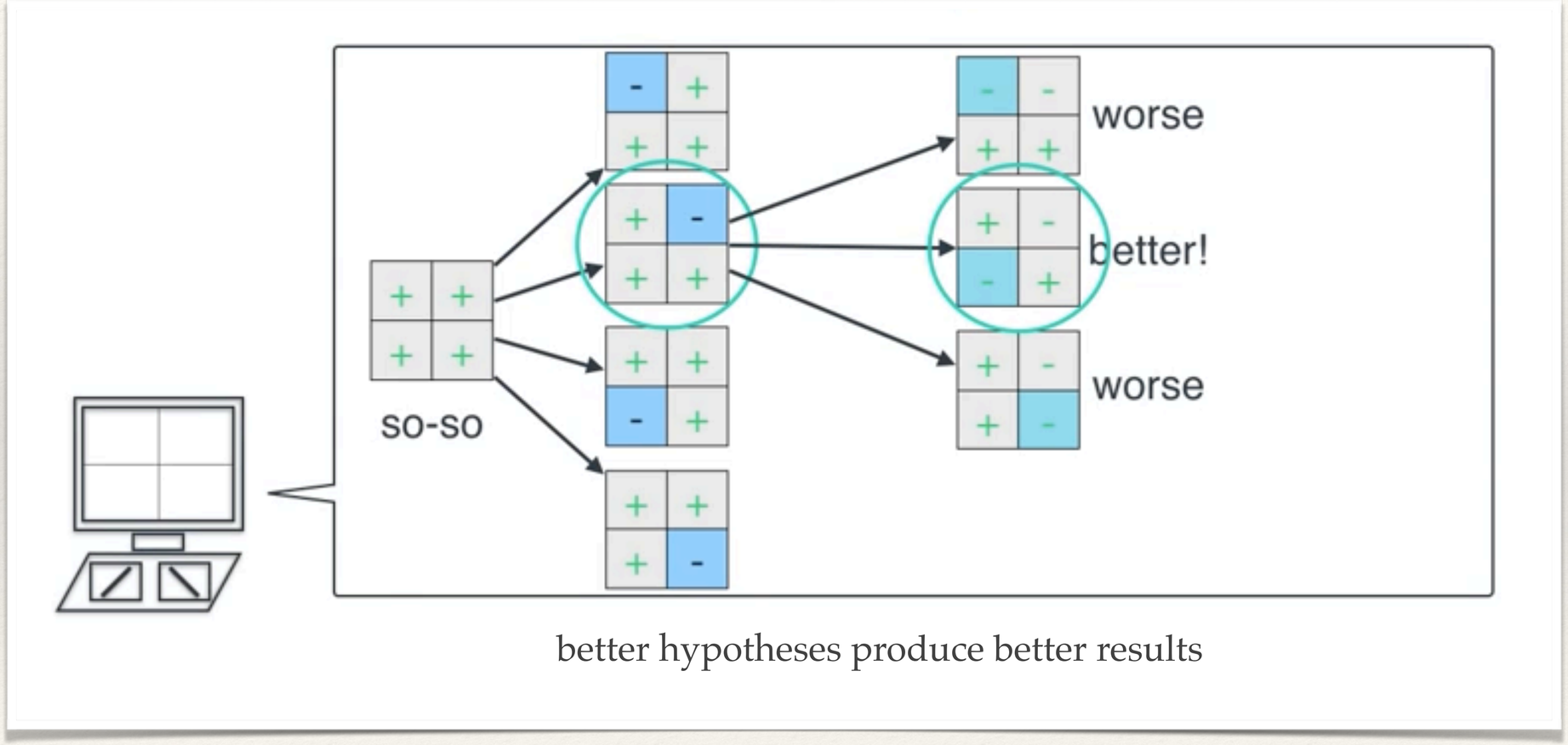
Image: Luis Serrano

# What is machine learning?

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

- Tom Mitchell

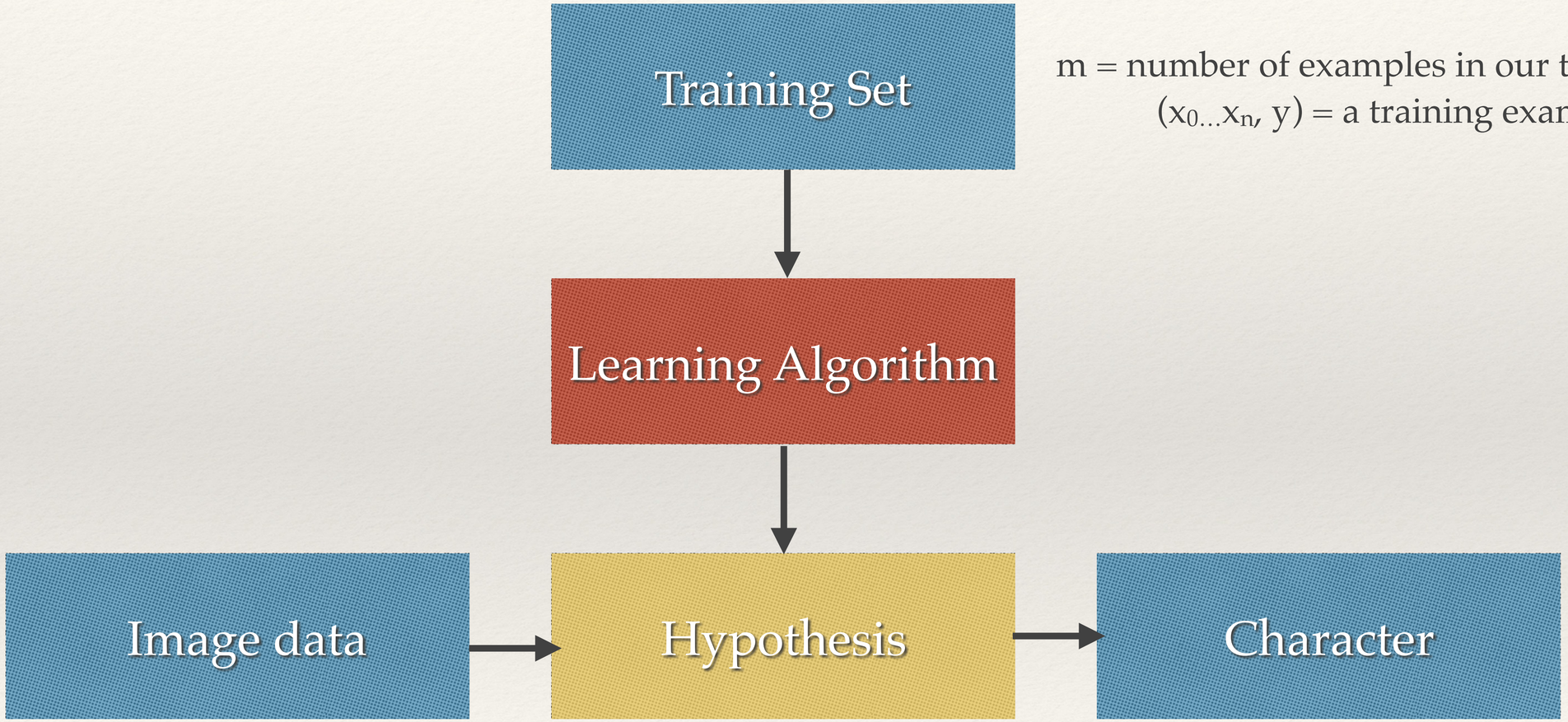
# Improving P




better hypotheses produce better results

Image: Luis Serrano

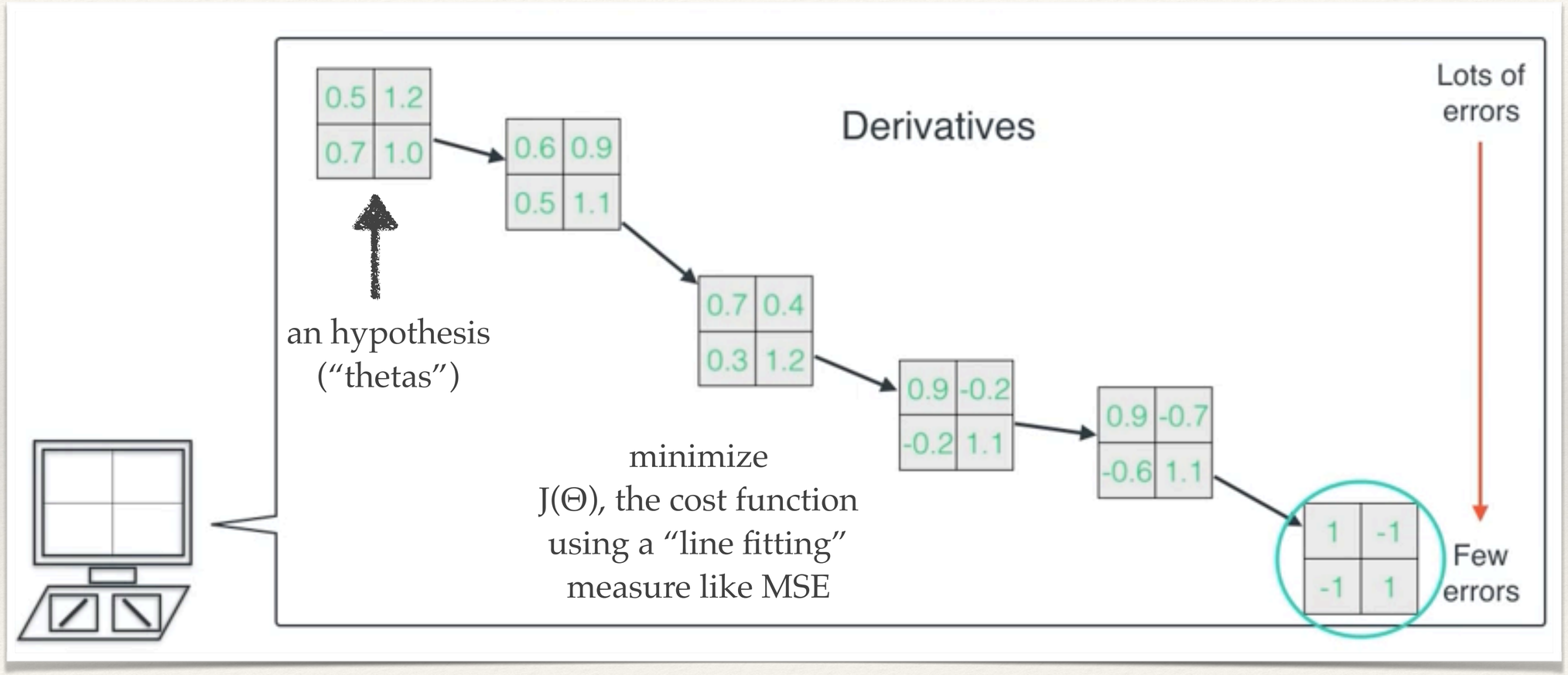
# Supervised learning block diagram




m = number of examples in our training set

$(x_{0 \ldots} x_n, y)$ = a training example

# Gradient descent




$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(\hat{y}_i - y_i\right)^2 = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x_i) - y_i\right)^2$$

Image: Luis Serrano

# Finding a local minimum

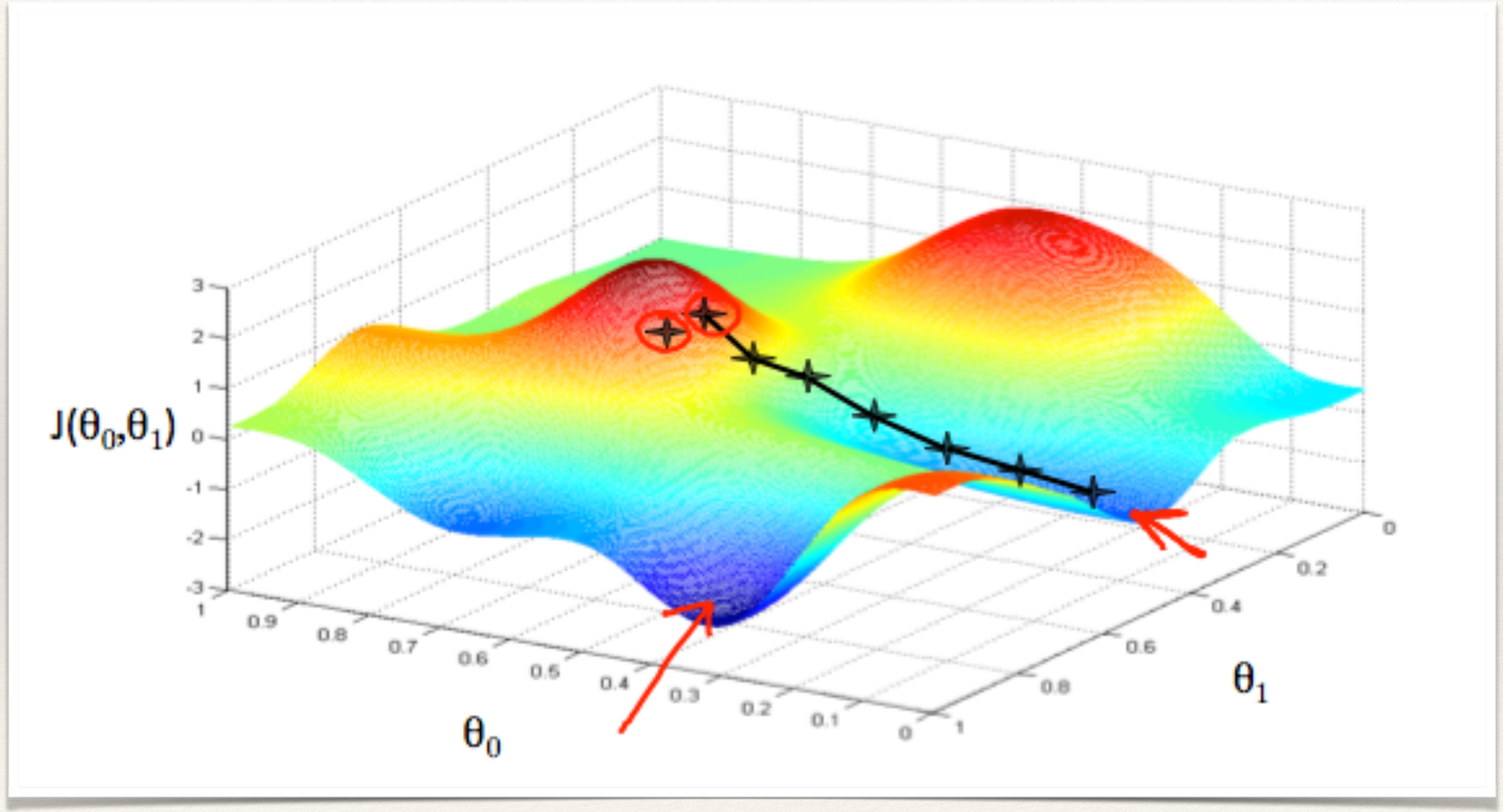



Image: Andrew Ng

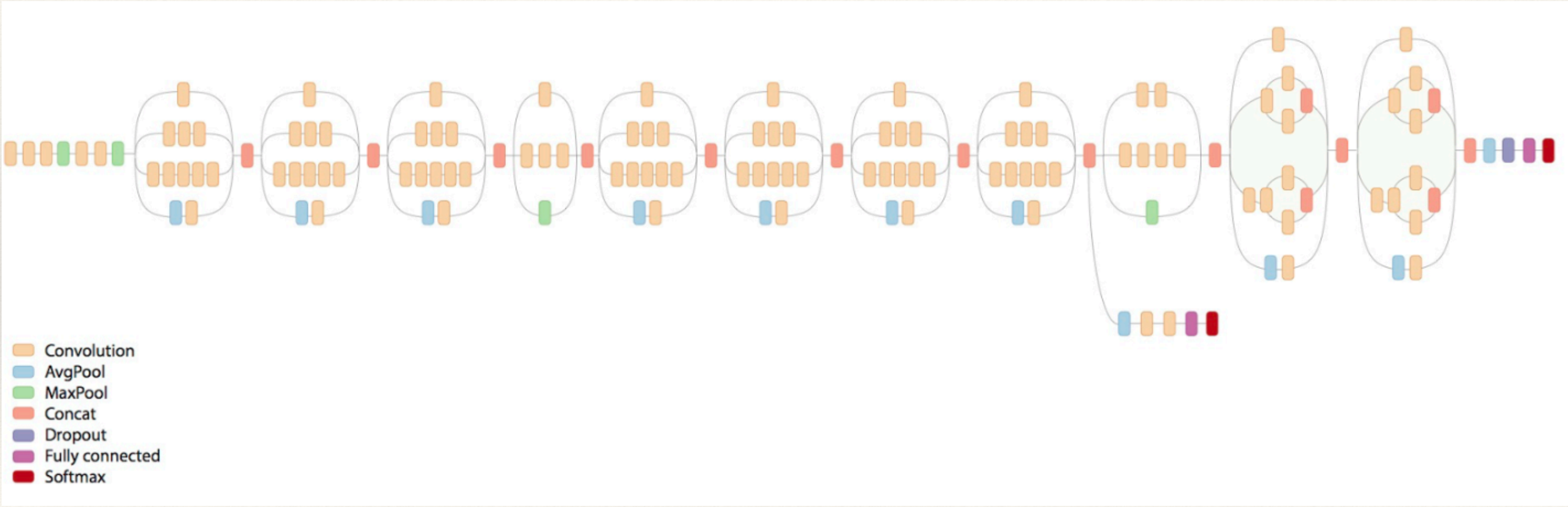
# Exit Ramp 3

- Run the following:

```
ruby mnist.rb
```

- Explore the code in the repo.

# Homework: Image recognition with Inception-v3




https://medium.com/@Arafat./image-recognition-in-ruby-tensorflow-df5d5c05389b