1. Using Image Generator, how do you label images? 1 / 1 point

- ○ It's based on the file name
- ○ You have to manually do it
- ○ TensorFlow figures it out from the contents
- ◉ It's based on the directory the image is contained in

✓ **Correct**

2. What method on the Image Generator is used to normalize the image? 1 / 1 point

- ○ normalize_image
- ◉ rescale
- ○ normalize
- ○ Rescale_image

✓ **Correct**

3. How did we specify the training size for the images? 1 / 1 point

- ○ The training_size parameter on the validation generator
- ○ The training_size parameter on the training generator
- ◉ The target_size parameter on the training generator
- ○ The target_size parameter on the validation generator

✓ **Correct**

4. When we specify the input_shape to be (300, 300, 3), what does that mean? 1 / 1 point

- ○ There will be 300 images, each size 300, loaded in batches of 3
- ◉ Every Image will be 300x300 pixels, with 3 bytes to define color
- ○ Every Image will be 300x300 pixels, and there should be 3 Convolutional Layers
- ○ There will be 300 horses and 300 humans, loaded in batches of 3

✓ **Correct**

5. If your training data is close to 1.000 accuracy, but your validation data isn't, what's the risk here? 1 / 1 point

- ○ You're overfitting on your validation data
- ○ No risk, that's a great result
- ○ You're underfitting on your validation data
- ◉ You're overfitting on your training data

✓ **Correct**

6. Convolutional Neural Networks are better for classifying images like horses and humans because: 1 / 1 point

- ○ In these images, the features may be in different parts of the frame
- ○ There's a wide variety of horses
- ○ There's a wide variety of humans
- ◉ All of the above

✓ **Correct**

7. After reducing the size of the images, the training results were different. Why? 1 / 1 point

- ○ There was less information in the images
- ○ The training was faster
- ○ There was more condensed information in the images
- ◉ We removed some convolutions to handle the smaller images

✓ **Correct**