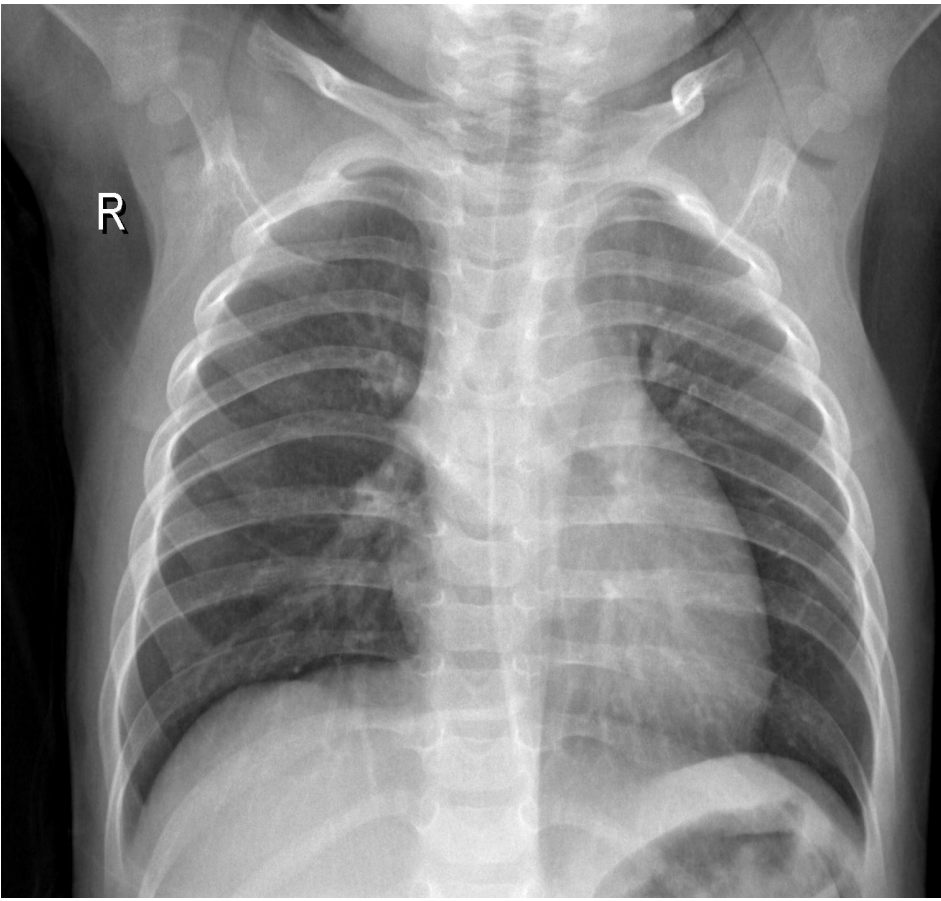


Predicting Pneumonia with Neural Networks

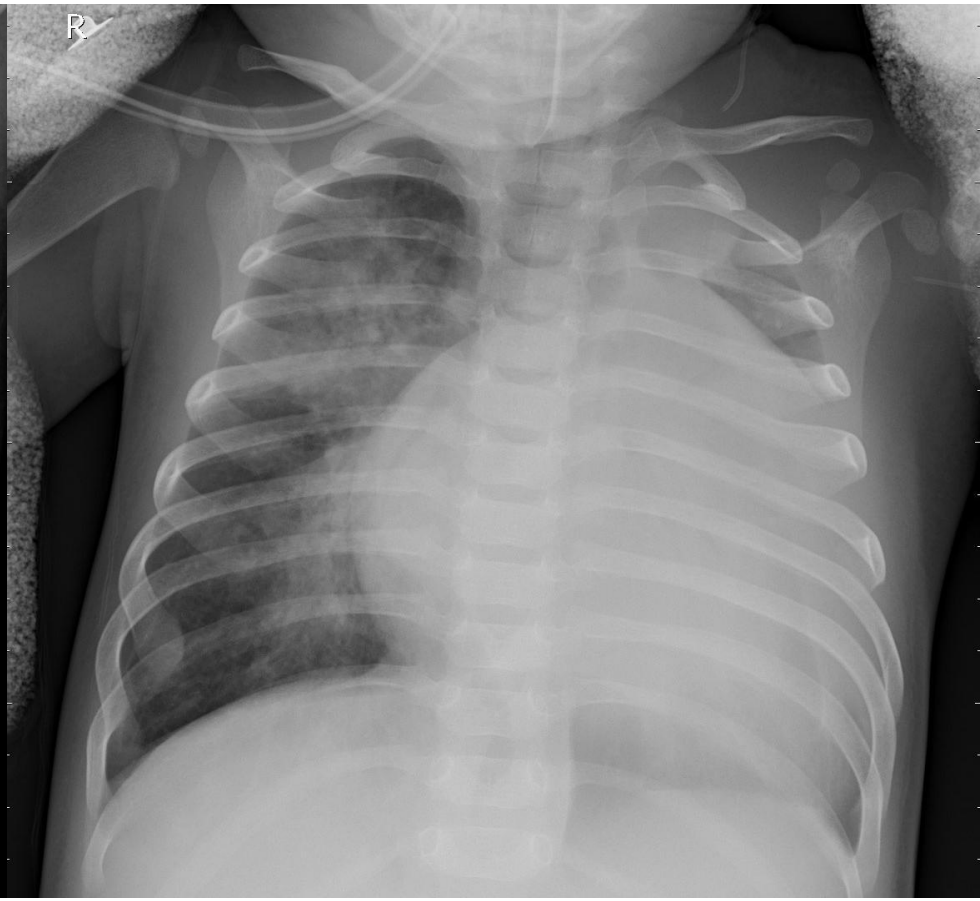
Dr. Kyle Boerstler, Ph.D.

A dark blue diagonal gradient bar that starts from the bottom left and extends towards the top right, covering the lower half of the slide.

Normal



Pneumonia



Preparing the Images

- ImageDataGenerator to read in and manipulate the files
- Dataset:
 - 5216 images in train set ($\frac{1}{3}$ healthy, $\frac{2}{3}$ pneumonia)
 - 624 images in test set (balanced)
 - 16 images in holdout set (balanced)
- Moved 70 images from Train set to holdout set to increase size of set for final comparison.
- Augmented train images by adding 15 degree rotations and 0.1 shear to randomly selected images for each epoch.
- Images in grayscale, so made a set of grayscale and a set of RGB images (using skimage) to check for differences in model performance.

Hit the Books!

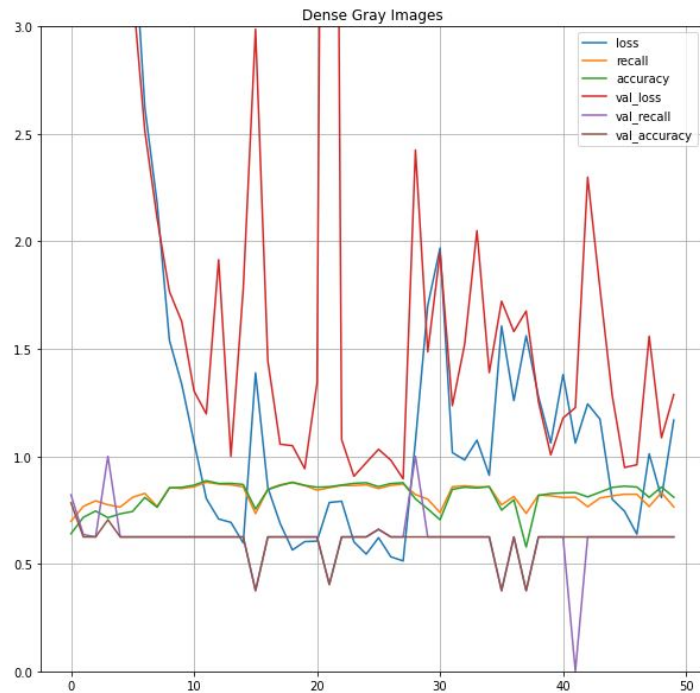
- Stick to using Adam and SGD with nesterov and momentum at first, and if time try some other optimizers.
- Use Relu for speed, but Selu with lecun normalization might be something to try.
- Learning rate, decay, and schedulers! (I didn't have time to mess with schedulers, but learned how to do it)
- Batch Normalization to prevent overfitting, and use it everywhere according to Skylar's book™.
- Tune, tune tune! Sometimes a model just needs a little tuning to work a lot better.

Building the Models: Deep, Dense Layers(w/Selu)

Model: "sequential"

Layer (type)	Output Shape	Param #
flatten (Flatten)	(None, None)	0
dense (Dense)	(None, 20)	1310740
batch_normalization (Batch Normalization)	(None, 20)	80
dense_1 (Dense)	(None, 20)	420
batch_normalization_1 (Batch Normalization)	(None, 20)	80
dense_2 (Dense)	(None, 20)	420
batch_normalization_2 (Batch Normalization)	(None, 20)	80
dense_3 (Dense)	(None, 20)	420
batch_normalization_3 (Batch Normalization)	(None, 20)	80
dense_4 (Dense)	(None, 20)	420
batch_normalization_4 (Batch Normalization)	(None, 20)	80
dense_5 (Dense)	(None, 20)	420
batch_normalization_5 (Batch Normalization)	(None, 20)	80
dense_6 (Dense)	(None, 20)	420
batch_normalization_6 (Batch Normalization)	(None, 20)	80
dense_7 (Dense)	(None, 20)	420
batch_normalization_7 (Batch Normalization)	(None, 20)	80
dense_8 (Dense)	(None, 20)	420
batch_normalization_8 (Batch Normalization)	(None, 20)	80
dense_9 (Dense)	(None, 20)	420
batch_normalization_9 (Batch Normalization)	(None, 20)	80

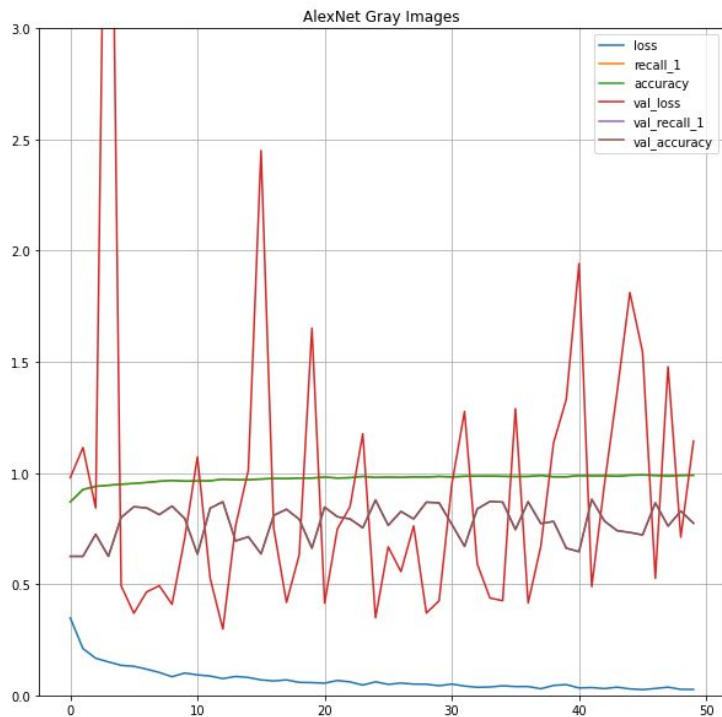
...for fifty layers!



Building the Models: AlexNet (reduced params)

Model: "sequential_8"

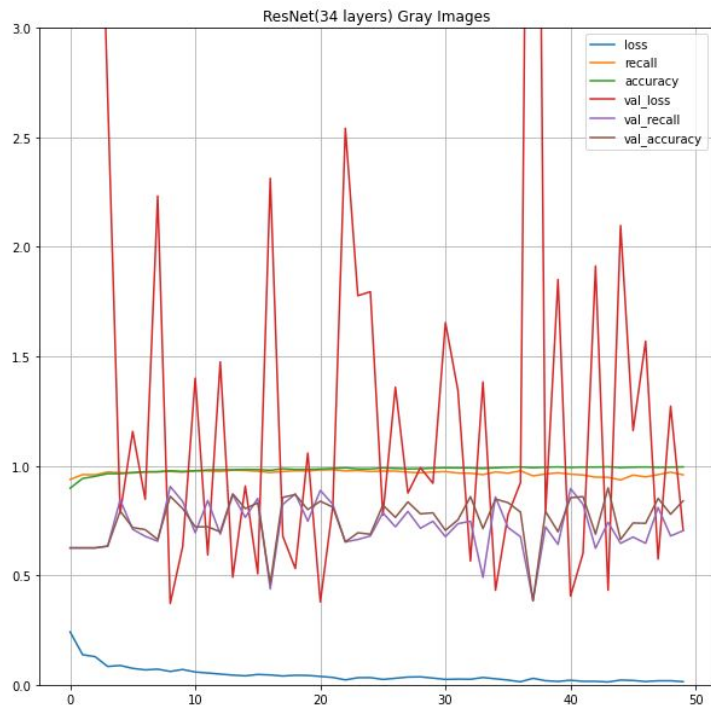
Layer (type)	Output Shape	Param #
conv2d_38 (Conv2D)	(None, 55, 55, 96)	11712
max_pooling2d_18 (MaxPooling)	(None, 28, 28, 96)	0
batch_normalization_5 (Batch Normalization)	(None, 28, 28, 96)	384
conv2d_39 (Conv2D)	(None, 28, 28, 256)	614656
max_pooling2d_19 (MaxPooling)	(None, 13, 13, 256)	0
batch_normalization_6 (Batch Normalization)	(None, 13, 13, 256)	1024
conv2d_40 (Conv2D)	(None, 13, 13, 384)	885120
batch_normalization_7 (Batch Normalization)	(None, 13, 13, 384)	1536
conv2d_41 (Conv2D)	(None, 13, 13, 384)	1327488
batch_normalization_8 (Batch Normalization)	(None, 13, 13, 384)	1536
conv2d_42 (Conv2D)	(None, 13, 13, 256)	884992
batch_normalization_9 (Batch Normalization)	(None, 13, 13, 256)	1024
dense_17 (Dense)	(None, 13, 13, 100)	25700
dropout_11 (Dropout)	(None, 13, 13, 100)	0
dense_18 (Dense)	(None, 13, 13, 50)	5050
dropout_12 (Dropout)	(None, 13, 13, 50)	0
flatten_5 (Flatten)	(None, 8450)	0
dense_19 (Dense)	(None, 2)	16902
activation_12 (Activation)	(None, 2)	0
Total params: 3,777,124		
Trainable params: 3,774,372		
Non-trainable params: 2,752		



Building the Models: ResNet (34 layers)

Model: "sequential_11"

Layer (type)	Output Shape	Param #
conv2d_44 (Conv2D)	(None, 112, 112, 64)	9408
batch_normalization_11 (Batch Normalization)	(None, 112, 112, 64)	256
activation_14 (Activation)	(None, 112, 112, 64)	0
max_pooling2d_21 (MaxPooling)	(None, 56, 56, 64)	0
residual_unit_1 (ResidualUnit)	(None, 56, 56, 64)	74240
residual_unit_2 (ResidualUnit)	(None, 56, 56, 64)	74240
residual_unit_3 (ResidualUnit)	(None, 56, 56, 64)	74240
residual_unit_4 (ResidualUnit)	(None, 28, 28, 128)	238912
residual_unit_5 (ResidualUnit)	(None, 28, 28, 128)	295936
residual_unit_6 (ResidualUnit)	(None, 28, 28, 128)	295936
residual_unit_7 (ResidualUnit)	(None, 28, 28, 128)	295936
residual_unit_8 (ResidualUnit)	(None, 14, 14, 256)	920576
residual_unit_9 (ResidualUnit)	(None, 14, 14, 256)	1181696
residual_unit_10 (ResidualUnit)	(None, 14, 14, 256)	1181696
residual_unit_11 (ResidualUnit)	(None, 14, 14, 256)	1181696
residual_unit_12 (ResidualUnit)	(None, 14, 14, 256)	1181696
residual_unit_13 (ResidualUnit)	(None, 14, 14, 256)	1181696
residual_unit_14 (ResidualUnit)	(None, 7, 7, 512)	3676160
residual_unit_15 (ResidualUnit)	(None, 7, 7, 512)	4722688
residual_unit_16 (ResidualUnit)	(None, 7, 7, 512)	4722688
global_average_pooling2d_1 (Global Average Pooling)	(None, 512)	0
flatten_6 (Flatten)	(None, 512)	0
dense_20 (Dense)	(None, 2)	1026
Total params: 21,302,722		
Trainable params: 21,285,698		
Non-trainable params: 17,024		



Building the Models: MNIST CNN Color

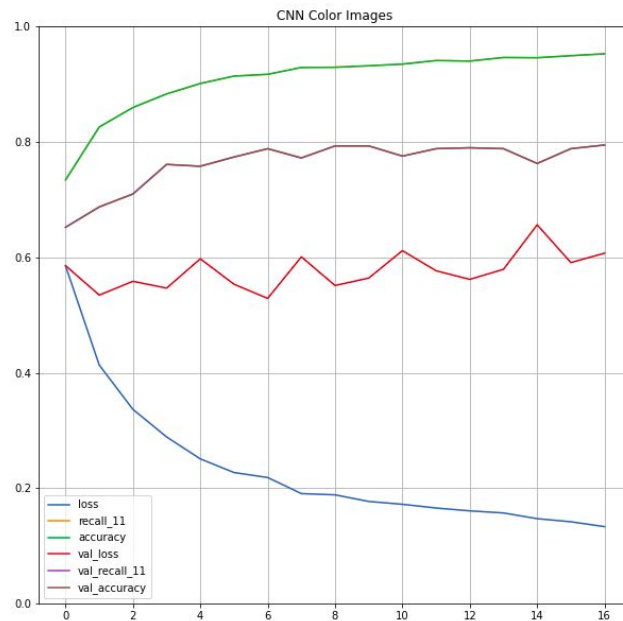
Model: "sequential_9"

Layer (type)	Output Shape	Param #
batch_normalization_18 (Batch Normalization)	(None, None, None, 3)	12
conv2d_37 (Conv2D)	(None, None, None, 64)	23296
max_pooling2d_20 (MaxPooling2D)	(None, None, None, 64)	0
batch_normalization_19 (Batch Normalization)	(None, None, None, 64)	256
conv2d_38 (Conv2D)	(None, None, None, 32)	18464
max_pooling2d_21 (MaxPooling2D)	(None, None, None, 32)	0
conv2d_39 (Conv2D)	(None, None, None, 16)	4624
conv2d_40 (Conv2D)	(None, None, None, 8)	136
conv2d_41 (Conv2D)	(None, None, None, 4)	36
max_pooling2d_22 (MaxPooling2D)	(None, None, None, 4)	0
flatten_9 (Flatten)	(None, None)	0
dense_18 (Dense)	(None, 50)	2928250
dropout_9 (Dropout)	(None, 50)	0
dense_19 (Dense)	(None, 2)	102
activation_9 (Activation)	(None, 2)	0

Total params: 2,975,176

Trainable params: 2,975,042

Non-trainable params: 134



Building the Models: MNIST CNN Grayscale

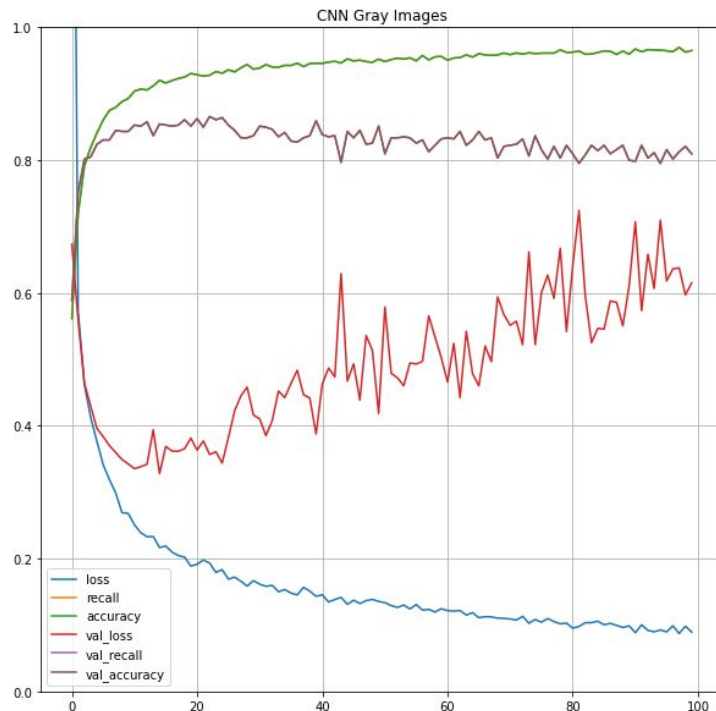
Model: "sequential_1"

Layer (type)	Output Shape	Param #
batch_normalization_2 (Batch Normalization)	(None, None, None, 1)	4
conv2d_5 (Conv2D)	(None, None, None, 64)	7808
max_pooling2d_3 (MaxPooling2D)	(None, None, None, 64)	0
batch_normalization_3 (Batch Normalization)	(None, None, None, 64)	256
conv2d_6 (Conv2D)	(None, None, None, 32)	18464
max_pooling2d_4 (MaxPooling2D)	(None, None, None, 32)	0
conv2d_7 (Conv2D)	(None, None, None, 16)	4624
conv2d_8 (Conv2D)	(None, None, None, 8)	136
conv2d_9 (Conv2D)	(None, None, None, 4)	36
max_pooling2d_5 (MaxPooling2D)	(None, None, None, 4)	0
flatten_1 (Flatten)	(None, None)	0
dense_2 (Dense)	(None, 50)	2928250
dropout_1 (Dropout)	(None, 50)	0
dense_3 (Dense)	(None, 2)	102
activation_1 (Activation)	(None, 2)	0

Total params: 2,959,680

Trainable params: 2,959,550

Non-trainable params: 130

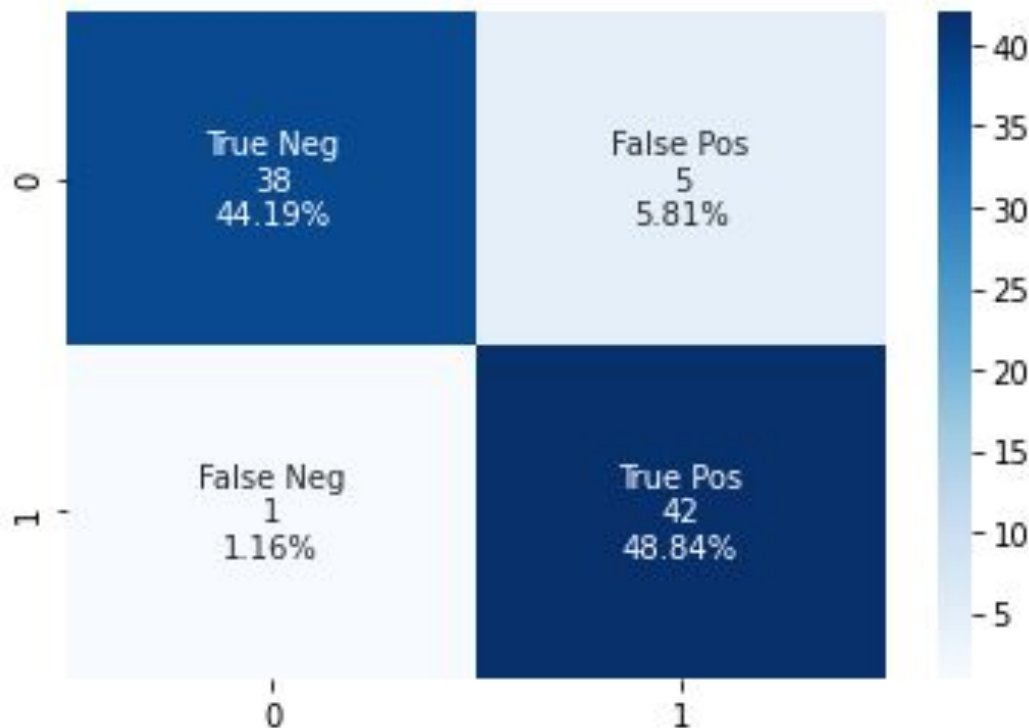


Hyperparameters and other features that seem important to this dataset:

- Learning Rate: Tuning this far lower than I initially thought produced better results across all models.
- BatchNormalization layers: Adding these helped my models train longer without overfitting.
- Kernel Size: If I start with larger kernel sizes and work my way down, I generally saw better results.
- Dropout: Helpful, but don't put too many layers of this or else you give your network brain damage.
- Trainable Parameters: Land meant what he said by keeping it under ~4 million trainable parameters. Any of my models over this just couldn't get anywhere.

And the winner is...

The MNIST CNN Grayscale! (modified)



- Precision: 89.4%
- Recall: 97.7%
- F_Score: 93.3%
- Accuracy: 93.0%

I can't tell either...

