GeoGuessr AI

using neural networks

Student: Radu-Augustin Vele

About the project

Goal: Based on an input image (Google Street-view style) detect the geographic area where it is located.

Subgoals:

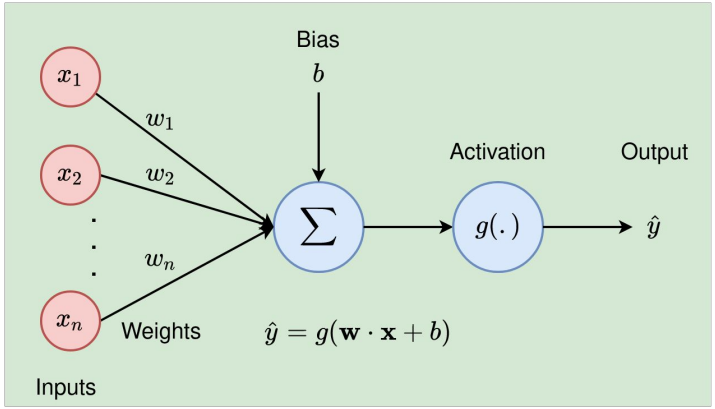
- areas as narrow as possible
- geotagged images

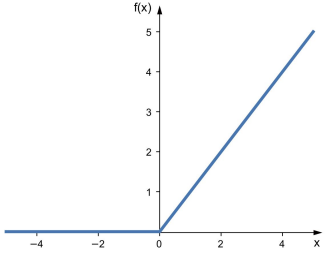
What is GeoGuessr?

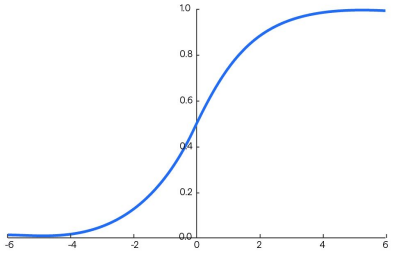
Quick Method Overview

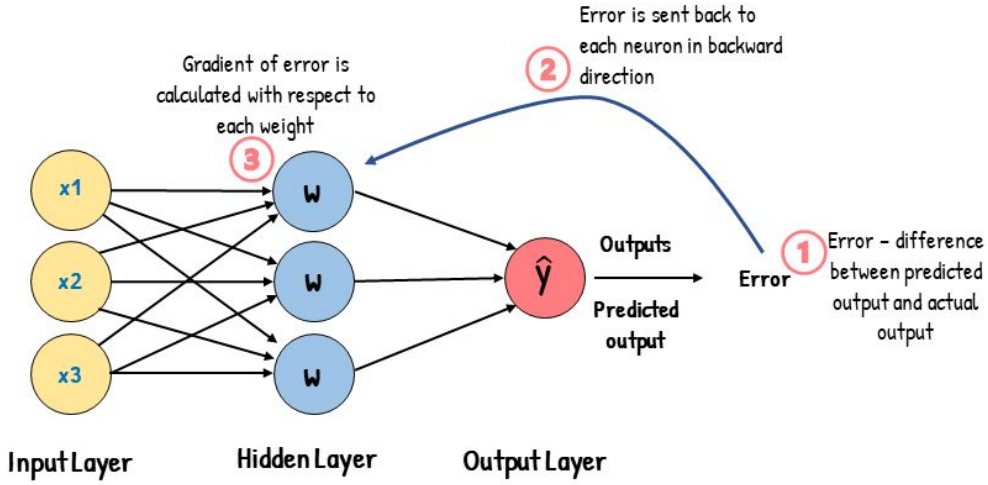
Neural Networks:

- Neuron
  - activations: ReLU, Sigmoid, Softmax
- Dense neural networks
  - interconnection
  - weights are learnt
- Learning through backpropagation



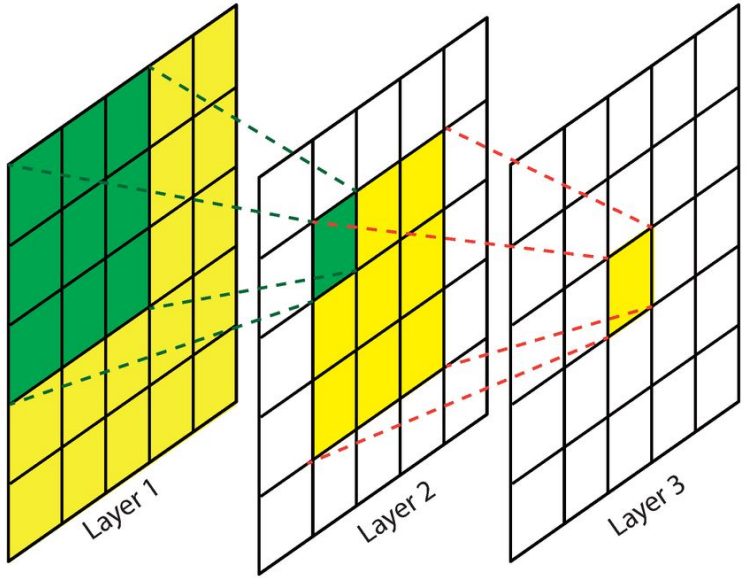


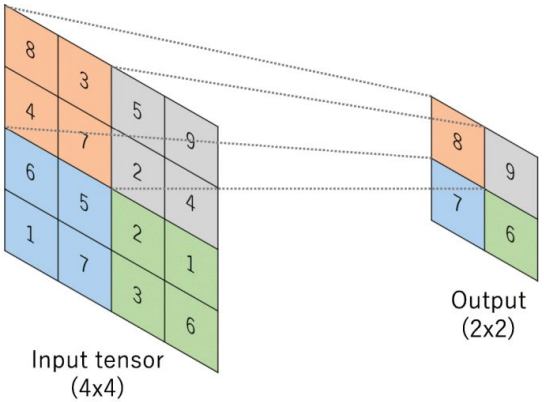




Convolutional Neural Networks:

- Kernels - Filters - For convolution
  - learnt through Backpropagation
- Hyperparameters
  - number of filters
  - kernel size
  - stride
  - padding
  - activation: ReLU
- Pooling Layer
  - Max pooling
- Coupled with DNN

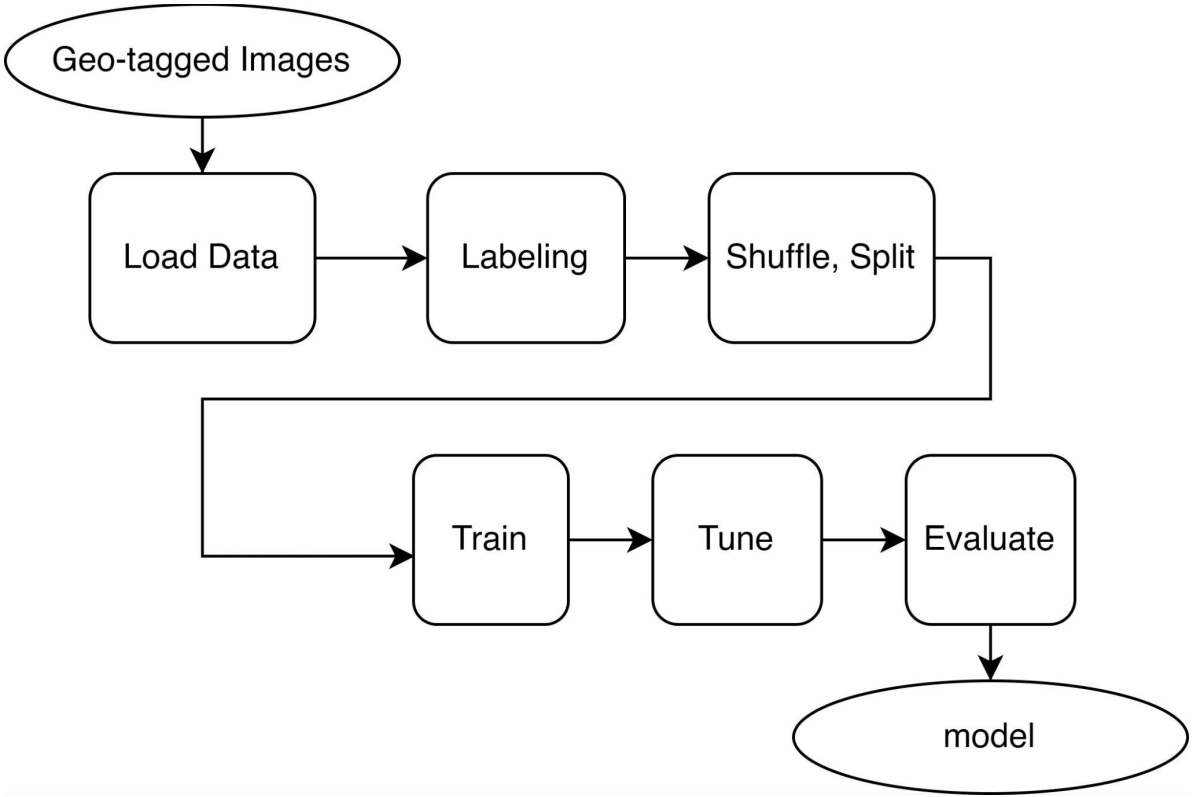




Transfer Learning:

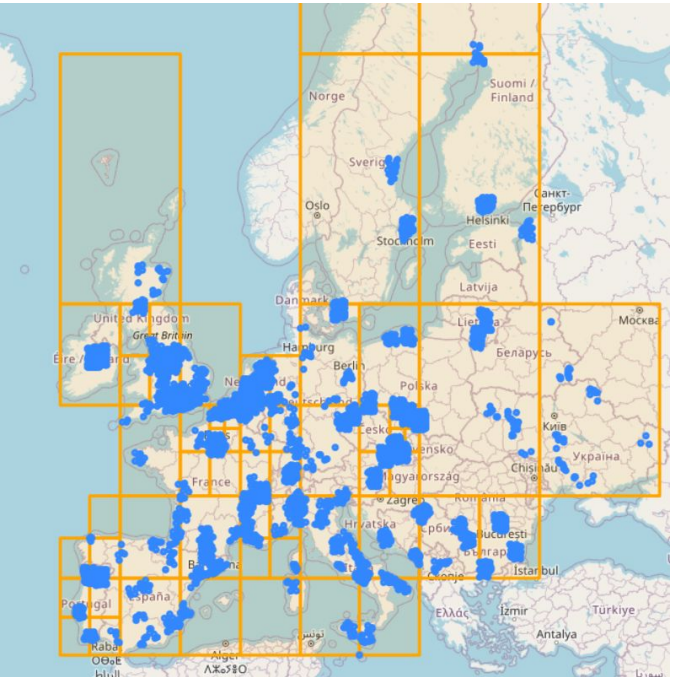
- leverage trained models

Solution steps

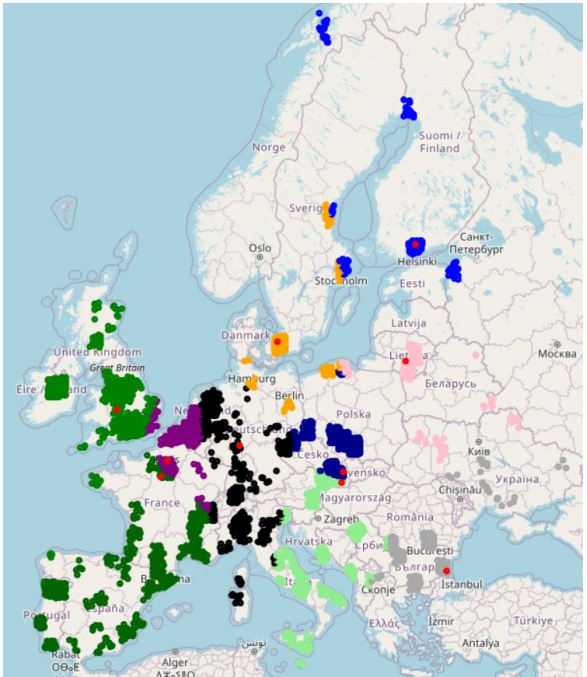


Data loading and labeling - Implementation

Grid-based labeling

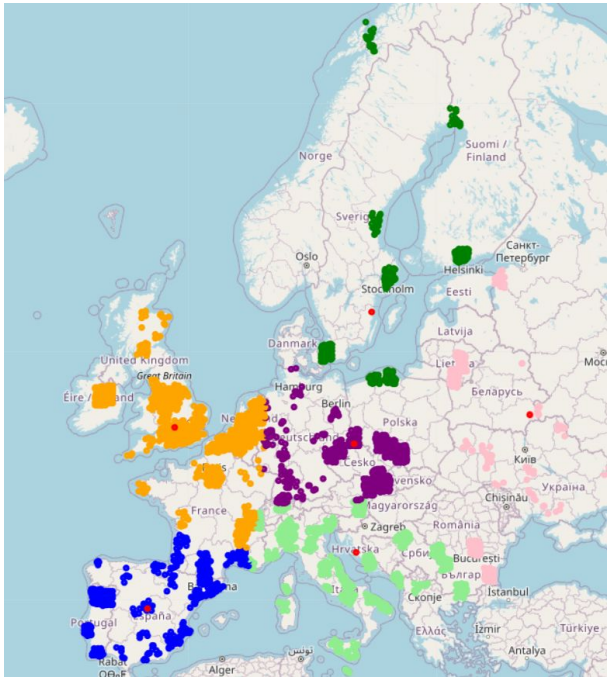


K-means clustering



K-means (1 iteration)

and handpicked centers



Training and chosen architectures

Simple CNN Regularized

Model: "Simple_CNN_224_Regularized"

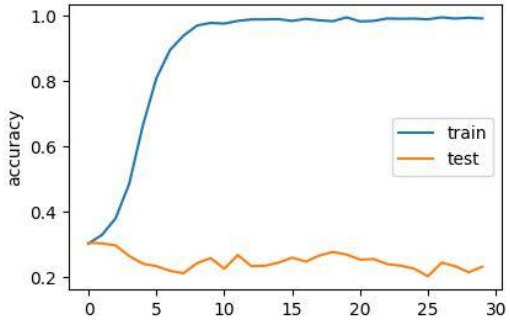
Layer (type)	Output Shape	Param #
rescaling_2 (Rescaling)	(None, 224, 224, 3)	0
Augmentation (Sequential)	(None, 224, 224, 3)	0
conv2d_2 (Conv2D)	(None, 224, 224, 32)	896
max_pooling2d_2 (MaxPooling2D)	(None, 112, 112, 32)	0
conv2d_3 (Conv2D)	(None, 112, 112, 32)	9248
max_pooling2d_3 (MaxPooling2D)	(None, 56, 56, 32)	0
flatten_1 (Flatten)	(None, 100352)	0
dense_3 (Dense)	(None, 128)	12845184
dropout (Dropout)	(None, 128)	0
dense_4 (Dense)	(None, 64)	8256
dropout_1 (Dropout)	(None, 64)	0
dense_5 (Dense)	(None, 6)	390

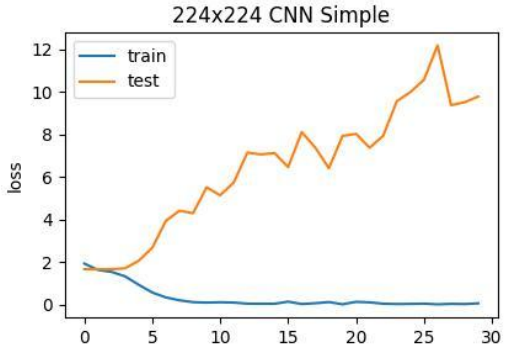
Deeper CNN Regularized

Model: "Deeper_CNN_64_Normalized"

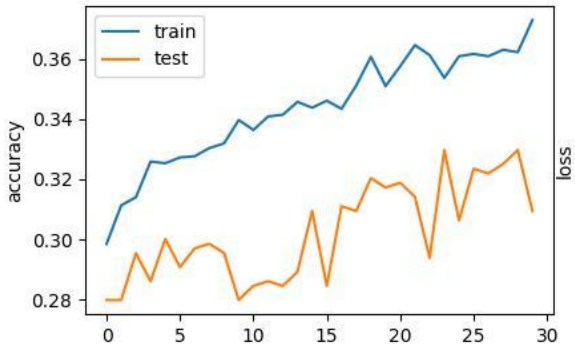
Layer (type)	Output Shape	Param #
rescaling_4 (Rescaling)	(None, 224, 224, 3)	0
Augmentation (Sequential)	(None, 224, 224, 3)	0
conv2d_9 (Conv2D)	(None, 224, 224, 64)	1792
conv2d_10 (Conv2D)	(None, 224, 224, 64)	36928
max_pooling2d_7 (MaxPooling2D)	(None, 112, 112, 64)	0
conv2d_11 (Conv2D)	(None, 112, 112, 32)	18464
conv2d_12 (Conv2D)	(None, 112, 112, 32)	9248
max_pooling2d_8 (MaxPooling2D)	(None, 56, 56, 32)	0
conv2d_13 (Conv2D)	(None, 56, 56, 16)	4624
max_pooling2d_9 (MaxPooling2D)	(None, 28, 28, 16)	0
flatten_3 (Flatten)	(None, 12544)	0
dense_9 (Dense)	(None, 128)	1605760
dropout_2 (Dropout)	(None, 128)	0
dense_10 (Dense)	(None, 64)	8256

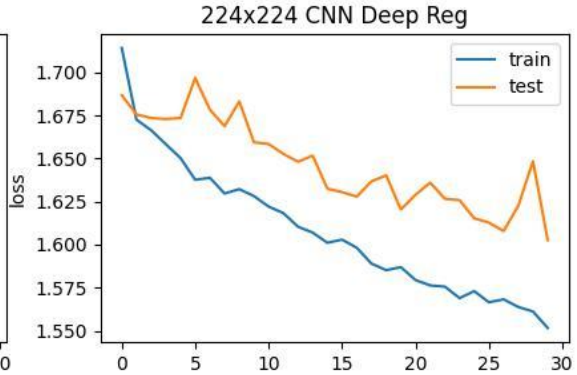
Simple CNN Regularized





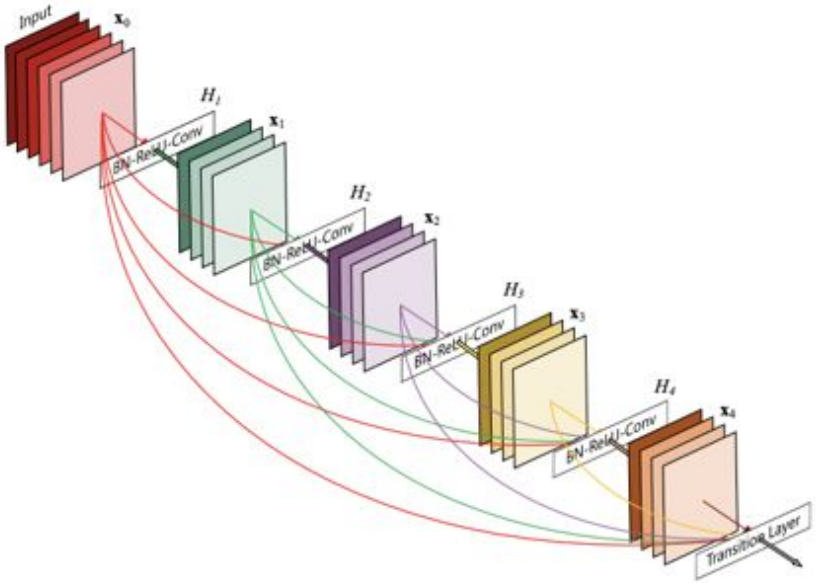
Deeper CNN Regularized

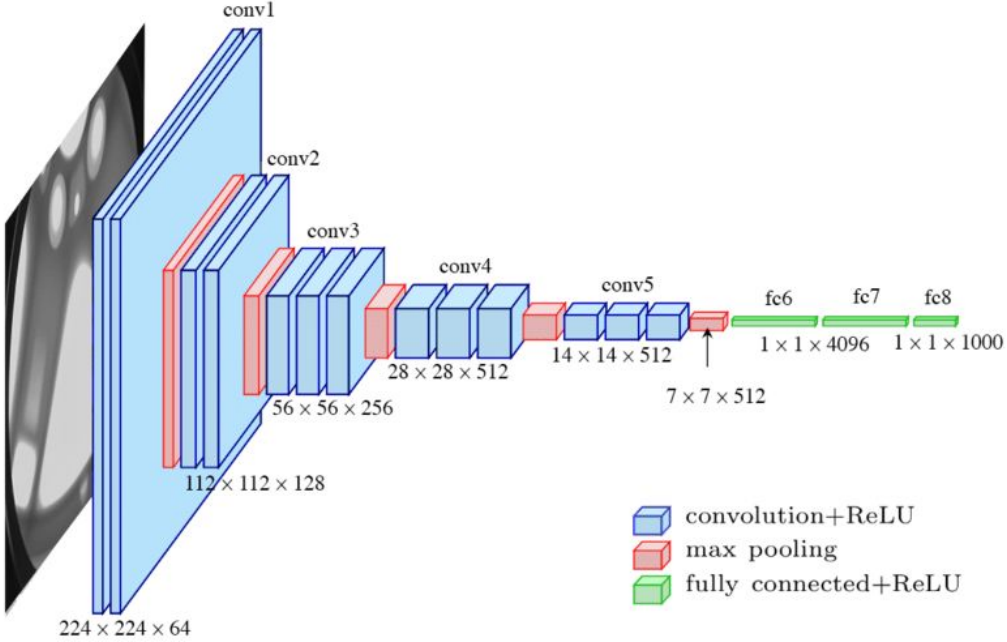




Transfer Learning

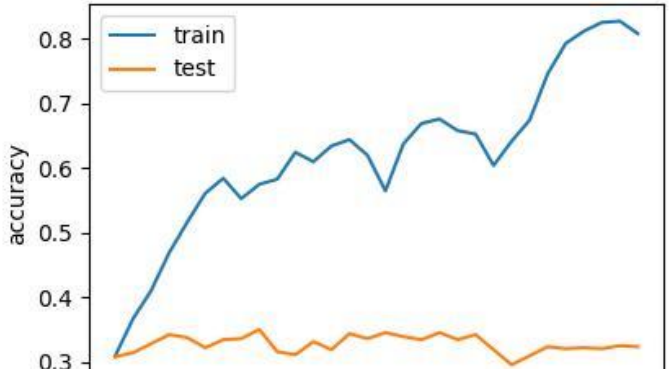
VGG



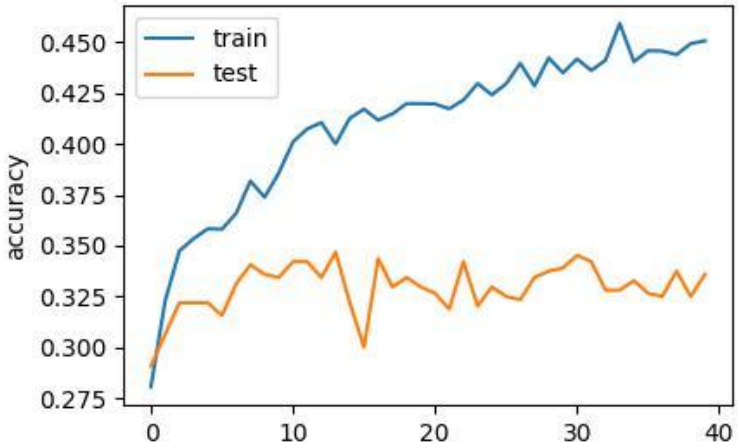


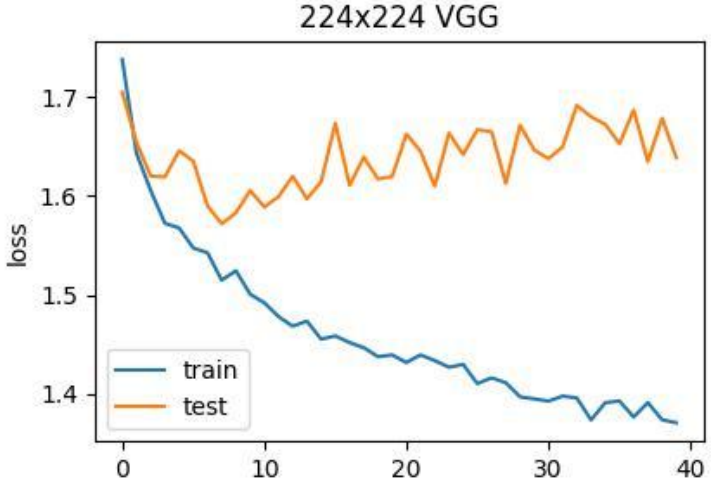
DenseNet

VGG


train
test
accuracy
0.8
0.7
0.6
0.5
0.4
0.3


224x224 VGG
train
test
loss
3.0
2.5
2.0
1.5
1.0
0.5


train
test
accuracy
0.450
0.425
0.400
0.375
0.350
0.325
0.300
0.275
0
10
20
30
40


224x224 VGG
loss
1.7
1.6
1.5
1.4
train
test
0
10
20
30
40

Evaluation and Results

Network Architecture	Hidden test set accuracy
Simple CNN 224	26.39%
Simple CNN Regularized	35.55%
Deeper CNN	34.36%
Deeper CNN Regularized	34.78%
VGG Regularized	39.28%
DenseNet	TBD - 30% approx on validation during training

Improvements, Solutions, Conclusions

Possible Sources of the low accuracy

- Data set issues (small, not evenly distributed)
- Labeling (not following well-enough geographical differences)
- Small number of epochs

Improvements

- Augment dataset (scraping from GeoGuessr)
- Improve labeling (close images - same label)