

ConvNet for Image Classification in the Study of Plankton Recognition

Xihuan Zeng, Rihan Chen, Jingxiang Li (G19)

School of Statistics, UMN

Problem Statement

Analysis Goal





- ▶ Classify ocean planktons given their images
- ▶ Input: 30,000 Images, each contains a single organism
- ▶ Output: 121 Labels of the plankton

Two Stages

- ▶ Pilot Study: 7,000 images with 8 labels
- ▶ Complete Study: 30,000 images with 121 different labels





Dataset

Table 1: Sample from Dataset, Part 1

1	acantharia_protist	
2	appendicularian_s_shape	
4	chaetognath_non_sagitta	
4	copepod_calanoid	

Dataset for Pilot Study

Table 2: Sample from Dataset, Part 2

5	copepod_cyclopoid_oithona	
6	trichodesmium_bowtie	
7	trichodesmium_puff	
8	trichodesmium_tuft	

Data Augmentation

Transformation

- ▶ Flip
- ▶ Rotation
- ▶ Rescaling
- ▶ Translation

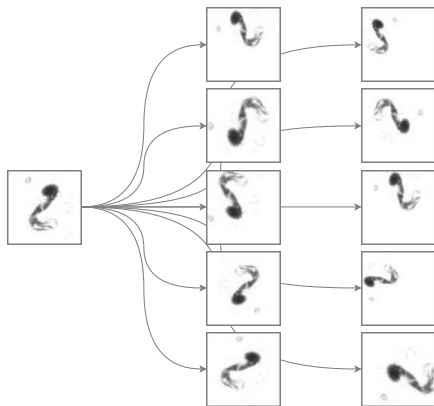


Figure 1: Original (Left) v.s. Augmented (Right)

Selected Models

Deep Learning

- ▶ Convolutional Neural Network (Keras)

Ensemble Methods

- ▶ Random Forest (scikit-learn)
- ▶ Gradient Boost (dmlc-XGBoost)

Prediction Accuracy

Pilot Study (8 Classes)

- ▶ ConvNet: **0.9365**
- ▶ Gradient Boost: 0.8464
- ▶ Random Forest: 0.8546

Complete Study (121 Classes)

- ▶ ConvNet: **0.674**

ConvNet Structure

Pilot Study (8 Classes)

- ▶ Modify on the ConvNet structure for *cifar10* data set.
- ▶ Relatively simple structure in terms of **depth** in case of *over-fitting*.
- ▶ 4 Convolution Layers, 2 Max Pooling Layers and 2 Fully Connected Layers.

ConvNet Structure

Table 3: Structure of ConvNet for Pilot Study

Layer	Size	Output
input		(32, 1, 32, 32)
augmentation		(32, 1, 32, 32)
convolution	32 3×3 kernels	(32, 32, 32, 32)
convolution	32 3×3 kernels	(32, 32, 32, 32)
max-pooling	2×2 , stride 2	(32, 32, 16, 16)
dropout	$p = 0.25$	(32, 32, 16, 16)
convolution	64 3×3 kernels	(32, 64, 16, 16)
convolution	64 3×3 kernels	(32, 64, 16, 16)
max-pooling	2×2 , stride 2	(32, 64, 8, 8)
dropout	$p = 0.25$	(32, 64, 8, 8)
fully connected	512 hidden units	(32, 512)
dropout	$p = 0.5$	(32, 512)
fully connected	8 way soft-max	(32, 8)

Real-time Data Augmentation

Randomly modify the images during the process of optimization.

Advantages

- ▶ Greatly save the space for data storage: only the original data need to be kept.
- ▶ The amount of data augmentation is simply decided by how many epochs we run.

Price to Pay

- ▶ Slow optimization compared with off-line data augmentation.

ConvNet Structure

Complete Study (121 Classes)

- ▶ Modify on the ConvNet structure for *cifar100* data set.
- ▶ Enlarge the image dimension for building more complicated model.
- ▶ Relatively complicated structure in terms of **depth**.
- ▶ 10 Convolution Layers, 4 Max Pooling Layers and 3 Fully Connected Layers.

Symbolic vs Imperative Programs

<pre>import numpy as np a = np.ones(10) b = np.ones(10) * 2 c = b * a d = c + 1</pre>	<pre>A = Variable('A') B = Variable('B') C = B * A D = C + Constant(1) <i># compiles the function</i> f = compile(D) d = f(A=np.ones(10), B=np.ones(10)*2)</pre>
---	--

Figure 2: Imperative (Left) v.s. Symbolic (Right)

Symbolic vs Imperative Programs



Figure 3: Graph Optimization

Symbolic vs Imperative Programs

Imperative style deep learning libraries

- ▶ Torch, Chainer and Minerva
- ▶ Advantage is flexibility and easier to use native language features and inject them into computation flow.

Symbolic style deep learning libraries

- ▶ Theano, CGT and Tensorflow
- ▶ Advantage is efficiency and space-saving

Modern GPU and CUDA

- ▶ Modern GPUs are very efficient at manipulating computer graphics and image processing, and their highly parallel structure makes them more effective than general-purpose CPUs for algorithms where the processing of large blocks of visual data is done in parallel.
- ▶ CUDA is a parallel computing platform created by NVIDIA. The CUDA platform is designed to work with programming languages such as C, C++ and Fortran

Reference

- ▶ Challenge Home Page:
<https://www.kaggle.com/c/datasciencebowl>
- ▶ Karas: <http://keras.io/>
- ▶ Symbolic Program:
https://mxnet.readthedocs.org/en/latest/program_model.html
- ▶ CUDA: <https://en.wikipedia.org/wiki/CUDA>