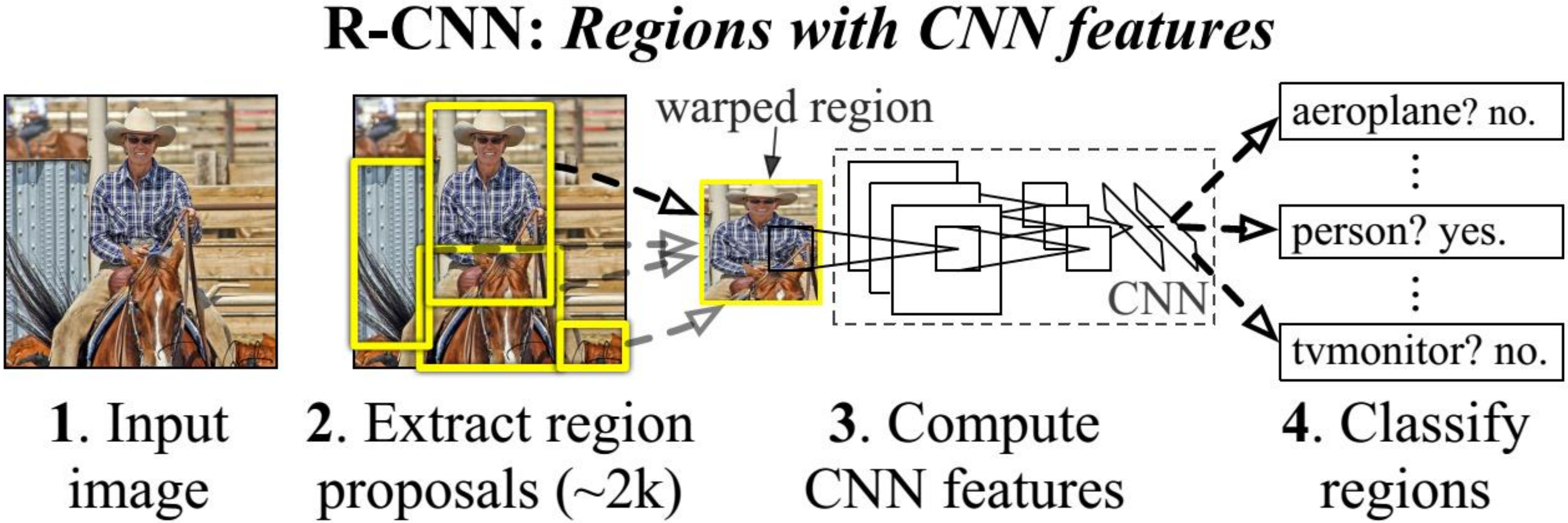
K Keras를 활용한

Image Classification Mini-App 구현

안영준 | 안철환 | 조대현 | 홍정현

Table of Contents

1. 주제 선정 및 역할 분담
2. DenseNet 아키텍쳐 소개
3. 모델 학습 과정 및 결과
4. Mini-App
5. 실습



R-CNN: *Regions with CNN features*
warped region
aeroplane? no.
person? yes.
tvmonitor? no.
CNN
1. Input image
2. Extract region proposals (~2k)
3. Compute CNN features
4. Classify regions

Can we implement Deep Learning in R?

Deep Learning in R

안영준 :
DenseNet 아키텍쳐
모델 구현(using R)

조대현 :
학습 그래프 Visualize
& moving plot

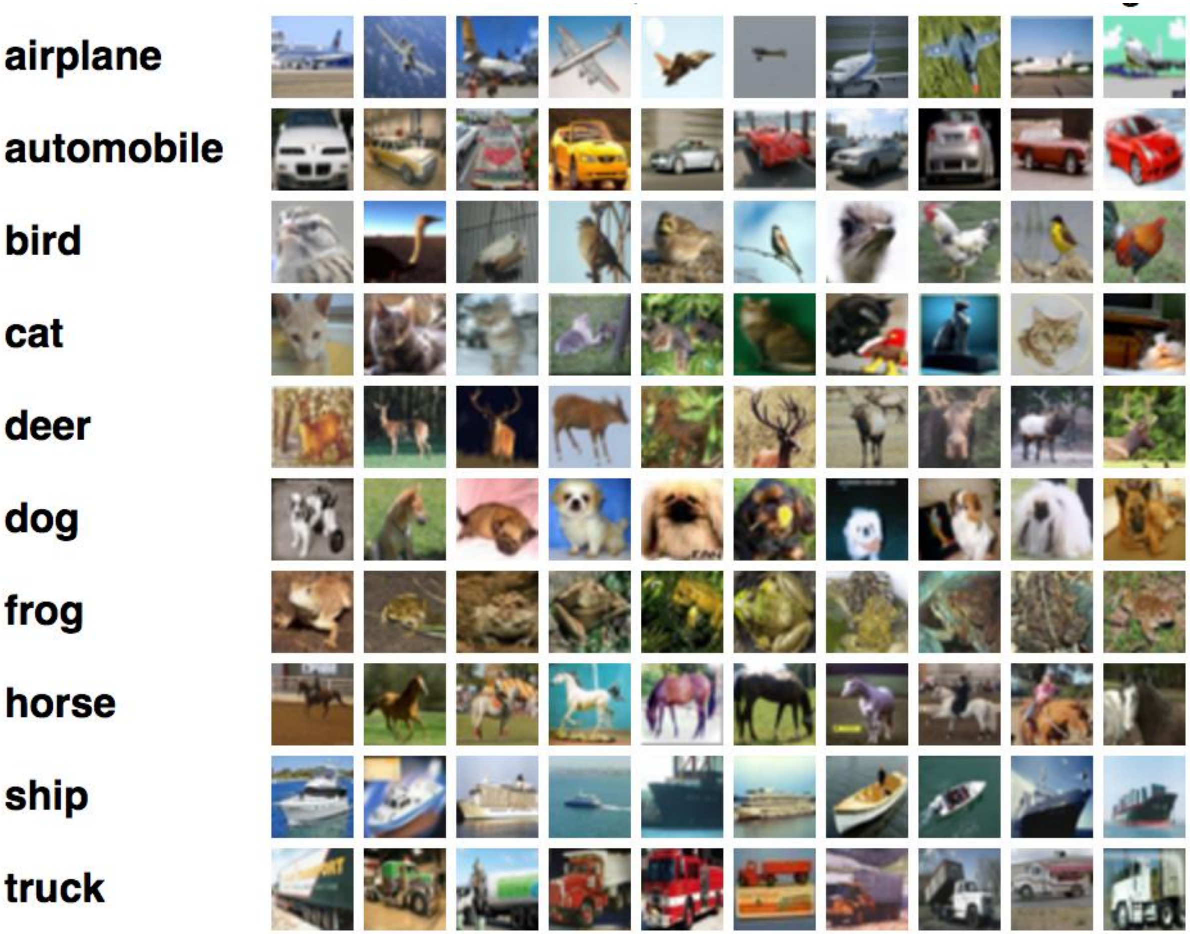
안철환 :
DeseNet분석 및
hyper-parameters
연구 및 적용

홍정현 :
Shiny App 를 이용한
Mini-App 구축

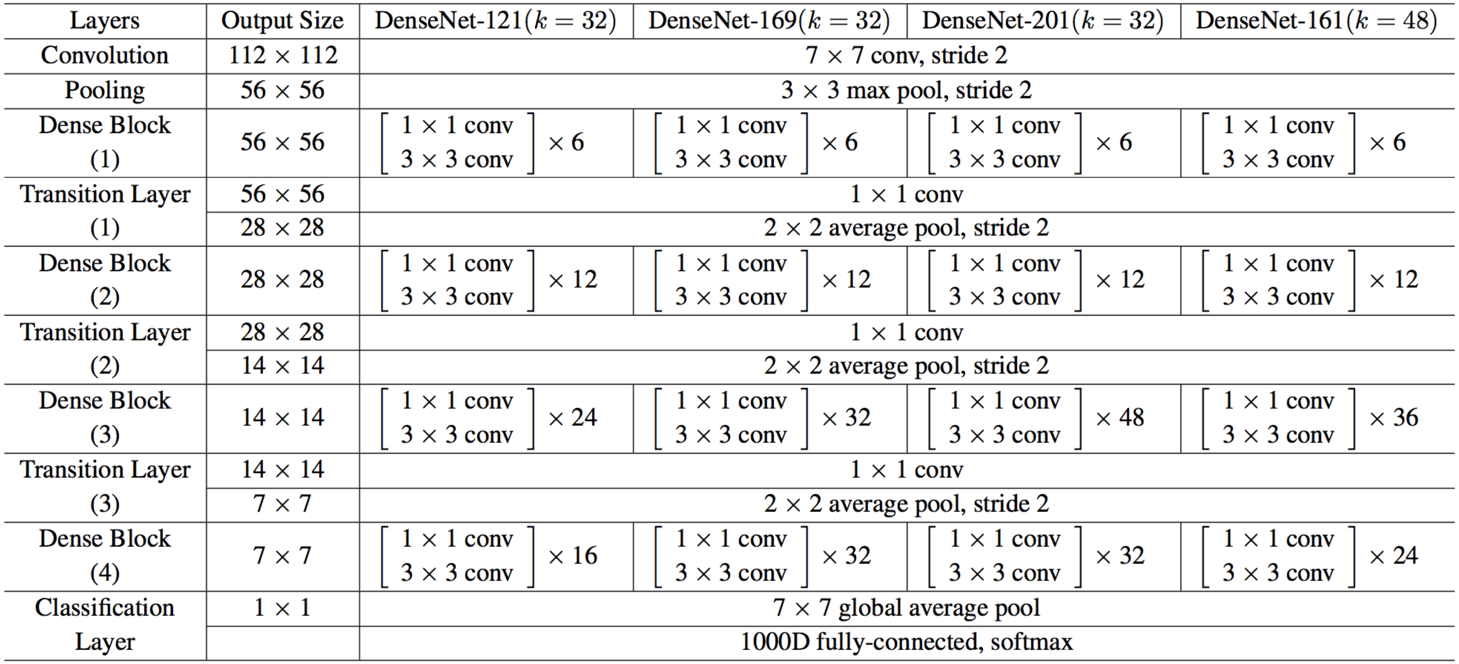
With Keras & Shiny

Datasets (CIFA-10)

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.



DenseNet (CNN Arichtecture)



Layers	Output Size	DenseNet-121$(k=32)$	DenseNet-169$(k=32)$	DenseNet-201$(k=32)$	DenseNet-161$(k=48)$
Convolution	112×112	7×7 conv, stride 2			
Pooling	56×56	3×3 max pool, stride 2			
Dense Block (1)	56×56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56×56	1×1 conv			
	28×28	2×2 average pool, stride 2			
Dense Block (2)	28×28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28×28	1×1 conv			
	14×14	2×2 average pool, stride 2			
Dense Block (3)	14×14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 36$
Transition Layer (3)	14×14	1×1 conv			
	7×7	2×2 average pool, stride 2			
Dense Block (4)	7×7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$
Classification Layer	1×1	7×7 global average pool			
		1000D fully-connected, softmax			

Table 1. DenseNet architectures for ImageNet. The growth rate for the first 3 networks is $k = 32$, and $k = 48$ for DenseNet-161. Note that each “conv” layer shown in the table corresponds the sequence BN-ReLU-Conv.

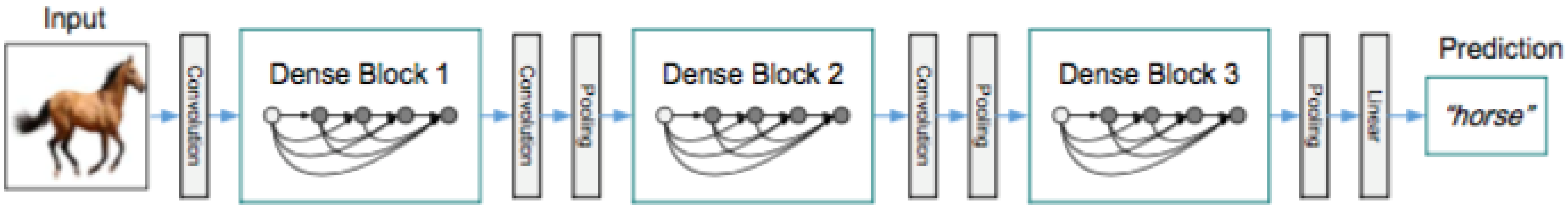


Figure 2: A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature-map sizes via convolution and pooling.

Moving Chart

acc.html

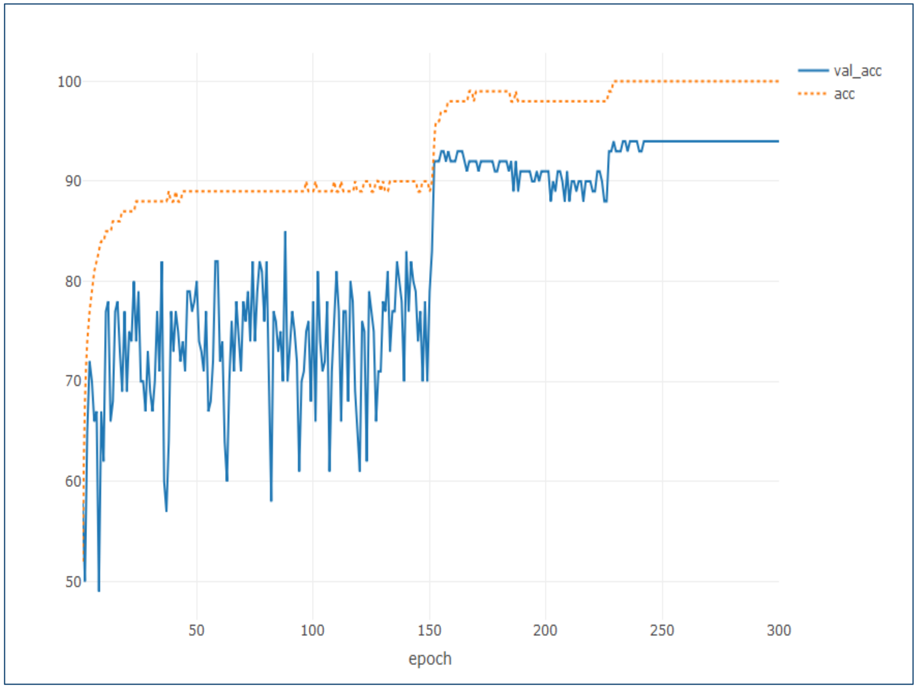
https://drive.google.com/open?id=1zmYjYuGjZRvvnNXaVttgje8DHhy2h94A

loss.html

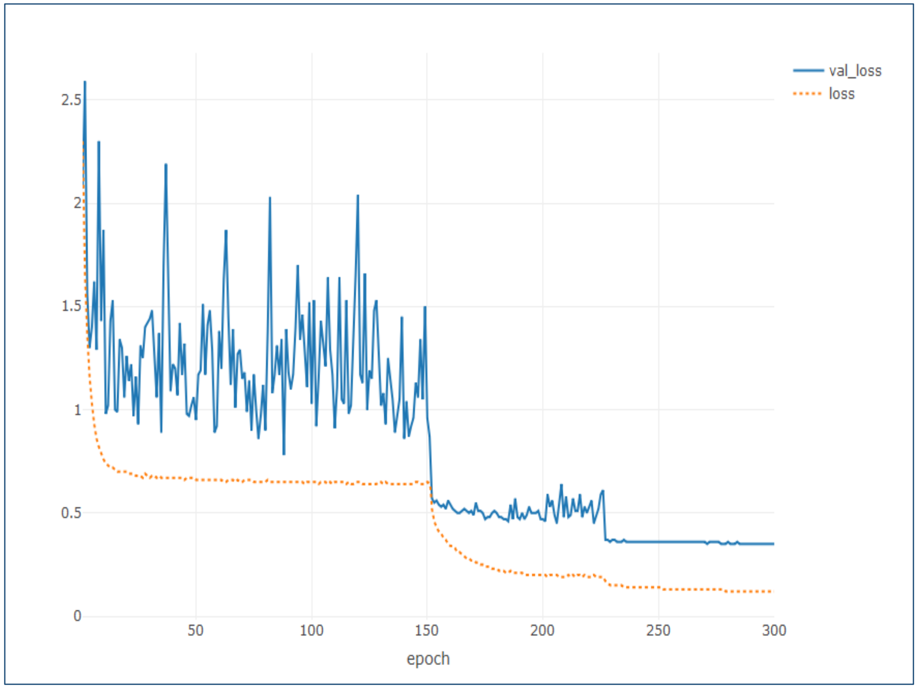
https://drive.google.com/open?id=1SaHntEhBIq4Q9k-cVpE5GaSnz_47TmVa

Results

■ Accuarcy



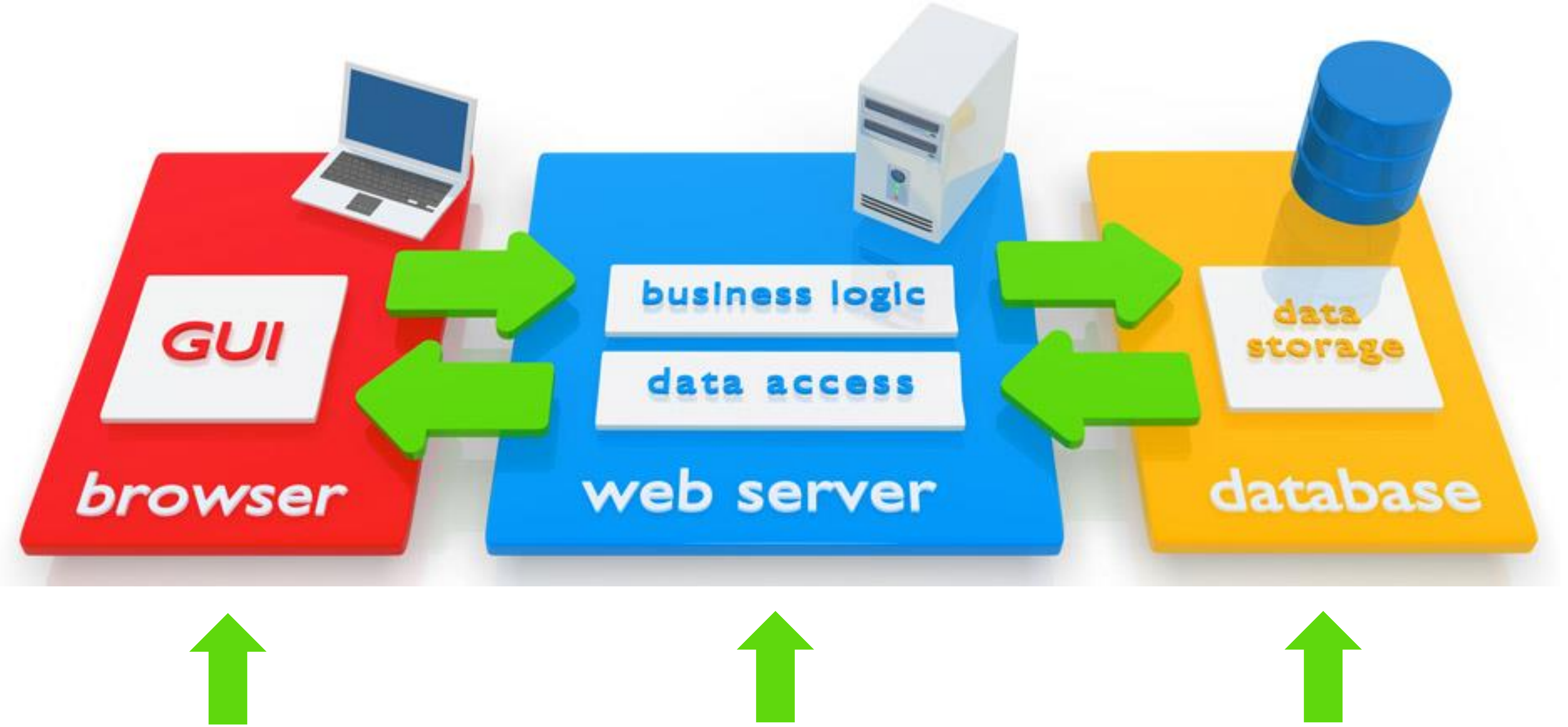
■ Loss



- Train : 50,000 / Test : 10,000

- Accuarcy : ***0.9351***
- Epoch : 300
- Drop-out rate : 0.2
- Learning Rate : 0.1(epoch<=150) -> 0.01(epoch <= 225) -> 0.001(else))

Mini-App (Shiny)



Server + UI

1) Resize Image

2) Predict with pre-trained model

R Data
+ Deep Learning Model

Let’s Try

https://aidencahn.shinyapps.io/cifar10_densenet/