CrePAS 6th the first session

Alexnet

ImageNet Classification with Deep Convolutional Neural Networks

크레파스 1기 | 손경일

Meaning(Importance)

Architecture

Layers_Analysis

Discussion

의의

Meaning

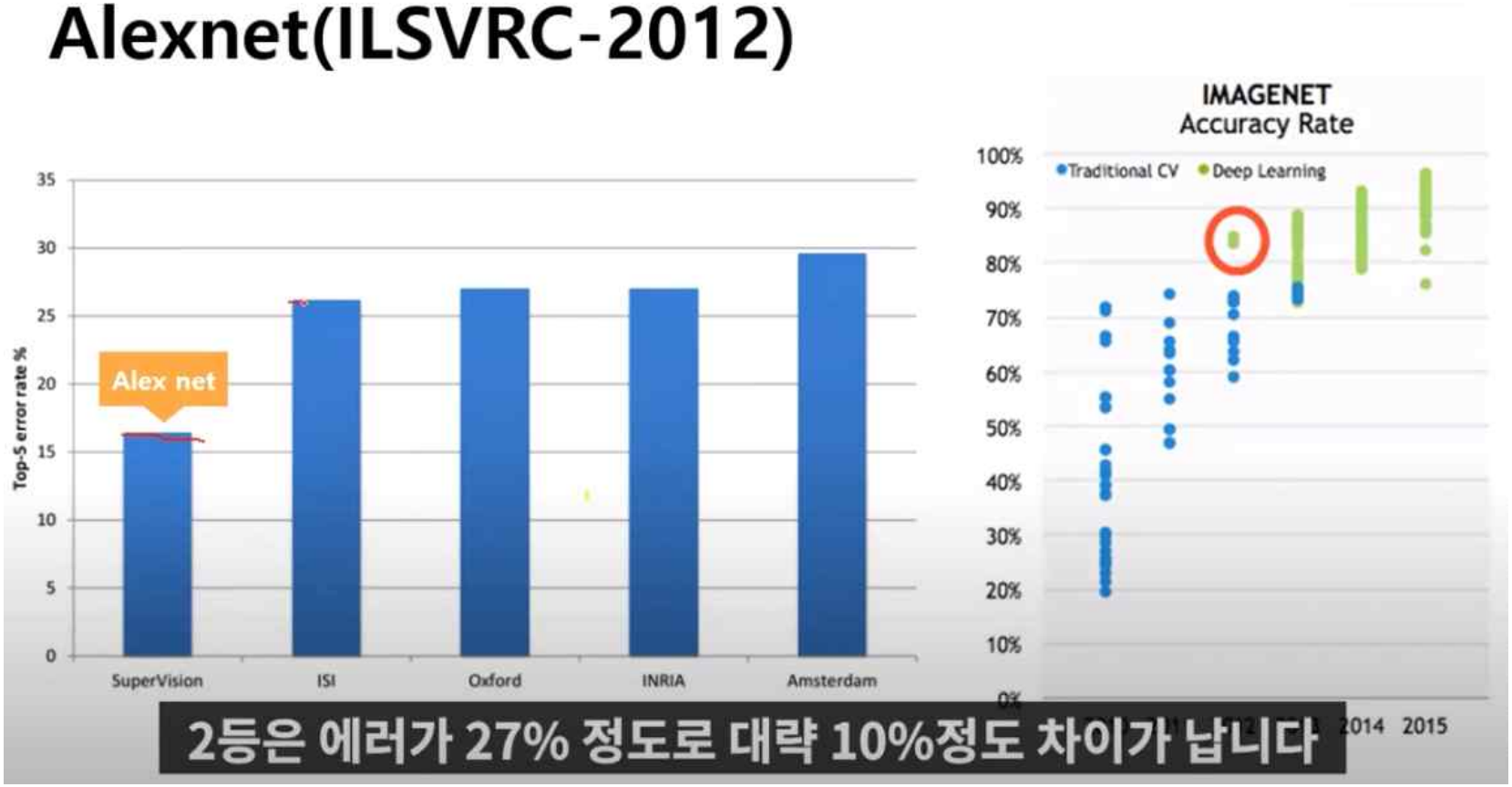
ILSVRC-2012

◎ ImageNet Large-Scale Visual Recognition Challenge
비전 분야에서의 올림픽

❯❯ Alexnet이 2등보다 top-5 error rate에서 10% 이상 더 낮았음

* The term top-5 error rate refers method of benchmarking machine learning models in the ImageNet Large Scale Visual Recognition Competition.
The model is considered to have classified a given image correctly if the target label is one of the model's top 5 predictions.

Alexnet(ILSVRC-2012)



» 기존 SVM 머신러닝보다 비약적으로 나은 성능을 내면서, 본격적으로 이미지 분야에서 딥러닝의 시대를 엶

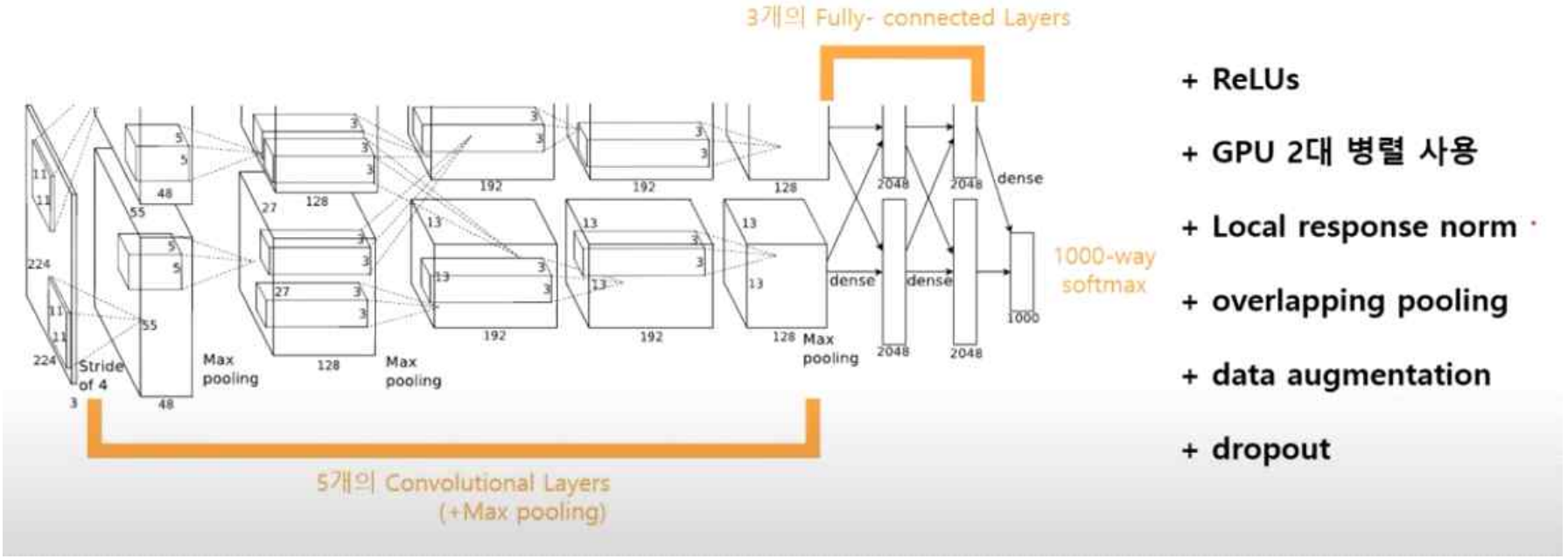
* Support Vector Machine vs Deep Learning
SVM에서는 feature를 모델에 제공해야 하는 반면, cnn은 중요한 features를 자동으로 골라냄
특히 CNN의 경우, 초기 layer: low level features, 레이어 깊이가 깊어질수록 high level features표현을 학습

Architecture

Architecture

- Architecture Summary
- Dataset
- Competence

» Overall



- 6천만개의 parameters, 65만개의 neurons, 8 layers

>> Dataset

- 1000개의 이미지 카테고리
- Test set이 Labeling된 ILSVRC-2010데이터를 주로 사용
- 120만개의 taining set, 5만개의 validation set, 15만개의 test set 사용
- 모든 이미지를 256*256 사이즈의 RGB컬러 사진으로 크기 통일
  : Resize & Crop

» 차별점

Architecture Competence

- ✓ Relu Nonlinearity
- ✓ Training on multiple GPUs
- ✓ Local Response Normalization
- ✓ Prevent Overfitting: Overlapping Pooling

2.3.1 성능 향상의 일등 공신

ActivationFunc:Relu

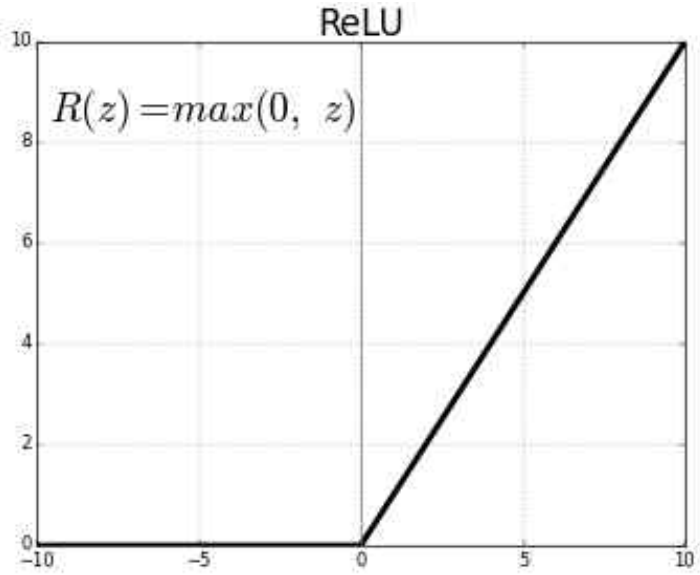
❖ 기존 활성화 함수
Sigmoid
Tanh

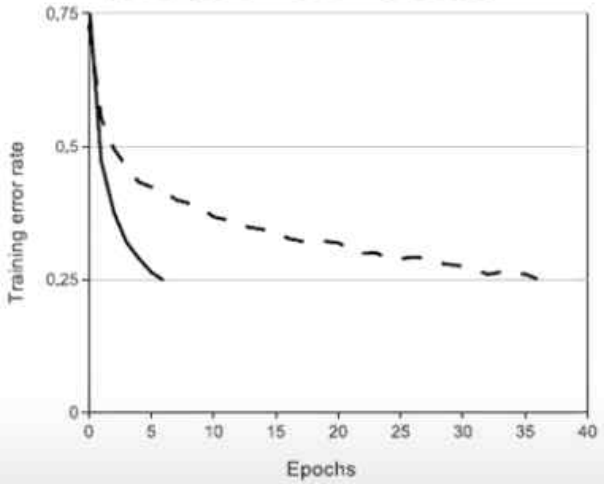
» RELU

ReLU

실제 인간의 뇌처럼 신경망을 구성한다면 활성화 함수는 tanh, sigmoid가 되어야 하는데, 왜 ReLU가 성능이 좋을까요? 그 이유는 ReLU가 **non-saturation**하기 때문입니다. 그래서 이때 **non-saturation**이 무엇일까요?

> saturation
>
> sigmoid, tanh 함수는 어떠한 값이 들어와도 -1~1, 혹은 0~1 사이의 값을 배출합니다. 그렇다면 큰 값이 들어왔을 때도, 저 사이의 값을 내놓아 **saturation(포화)**됩니다. 하지만 ReLU는 max(0, x)이므로 어떠한 값이 들어와도 제한이 없습니다. 값이 크면 클수록 더큰 gradient를 가지게 됩니다.



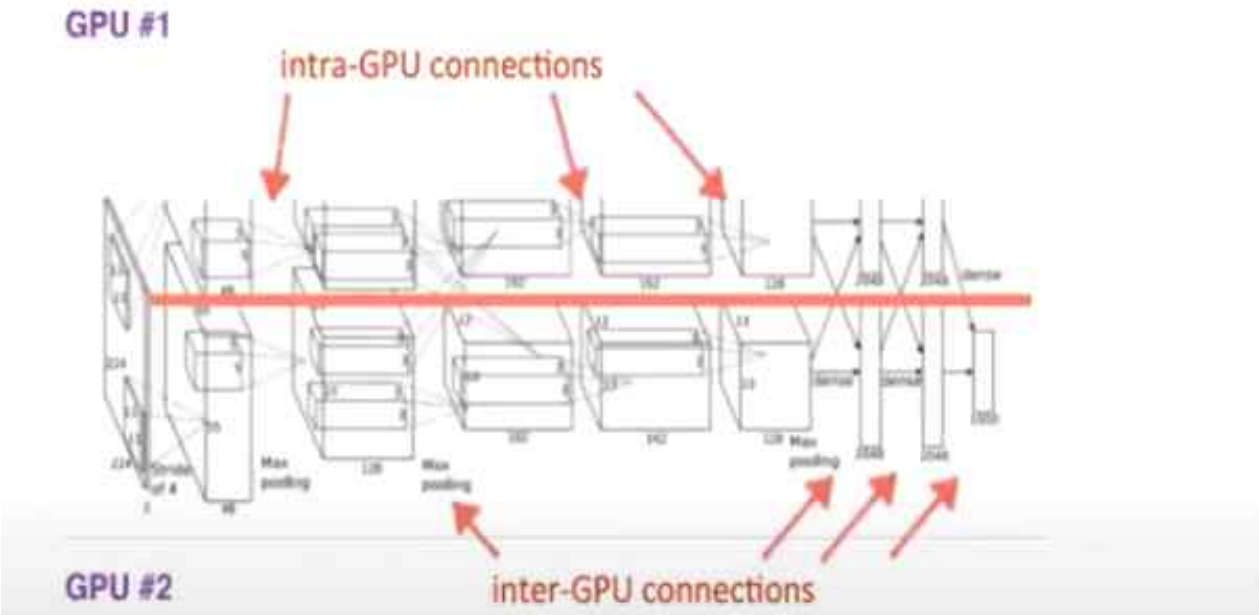


Error rate가 0.25에 도달하기 위한 epoch 수

약 6배 빠르게 수렴

3.3.2

Multiple GPU



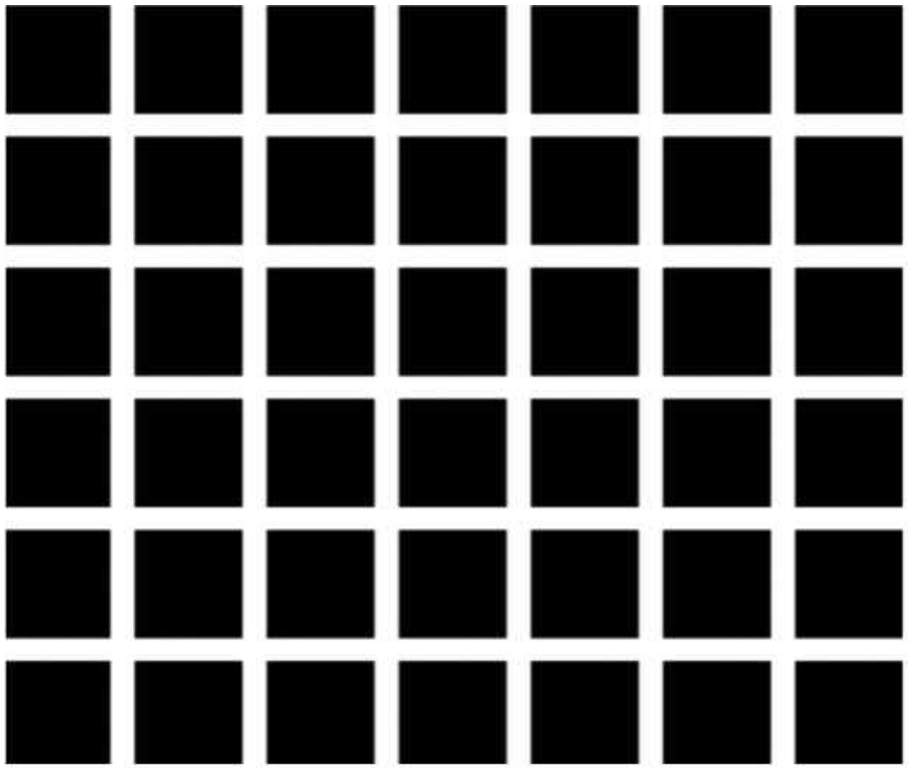
- 메모리 부족으로 GPU 2대를 사용하여 데이터 병렬 처리
- Intra GPU connection: 1,2,4,5번째 conv layers에서는 같은 GPU내에서의 커널만 사용 가능
- Inter GPU connection: 3번째 conv layer과 3개의 fully connected layers에서는 두 GPU의 모든 kernal들 사용 가능
- GPU사용함으로써 top5 error: 1.2%, top1 error: 1.7% 절감

Prvt Overfiting

Despite the attractive qualities of CNNs, and despite the relative efficiency of their local architecture, they have still been prohibitively expensive to apply in large scale to high-resolution images. Luckily, current GPUs, paired with a highly-optimized implementation of 2D convolution, are powerful enough to facilitate the training of interestingly-large CNNs, and recent datasets such as ImageNet contain enough labeled examples to train such models without severe overfitting.

which are detailed in Section 3. The size of our network made overfitting a significant problem, even with 1.2 million labeled training examples, so we used several effective techniques for preventing overfitting, which are described in Section 4. Our final network contains five convolutional and

Local Response Normalization



hermann-grid

where the sum runs over n "adjacent" kernel maps at the same spatial position, and N is the total number of kernels in the layer. The ordering of the kernel maps is of course arbitrary and determined before training begins. This sort of response normalization implements a form of lateral inhibition inspired by the type found in real neurons, creating competition for big activities amongst neuron outputs computed using different kernels. The constants k, n, α, and β are hyper-parameters whose

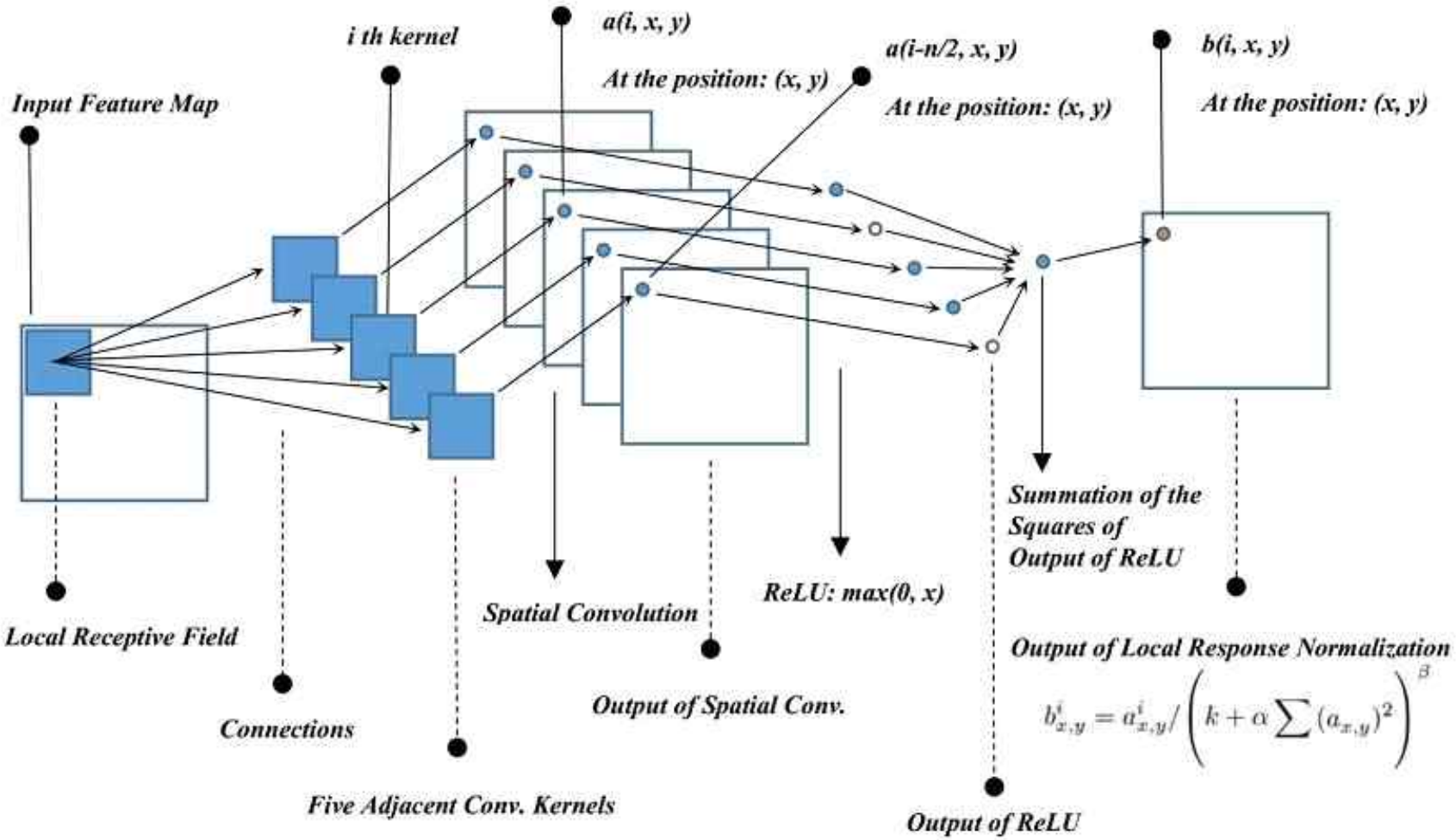
Prvt Overfitting(1): Local Response Normalization

- Relu 함수는 unbounded하기 때문에 LRN이 일반화에 도움이 됨
- 특정 값에 의해 과대 적합되는 것을 방지할 수 있음
- top-1, top-5 error rates를 각각 1.4%, 1.2% 감소시킴.

$$b^i_{x,y} = a^i_{x,y} \Big/ \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^{\beta}$$

논문에서는 k=2 , n = 5 , $\alpha = 10^{-4}$, $\beta = 0.75$ 사용

Local Response Normalization

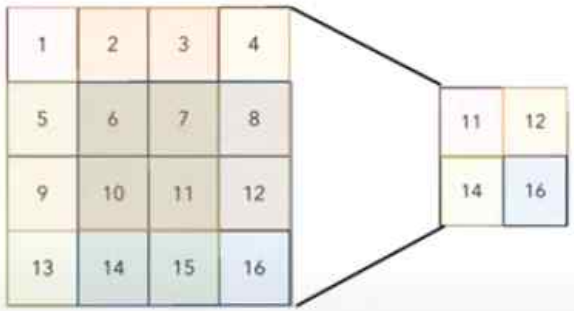


2-2

Prvt Overfiting(2): Overlapping Pooling

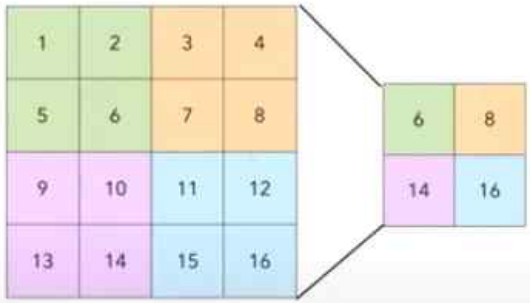
- **Overlapping pooling**

Pooling window 의 크기 > stride의 크기



- **Traditional pooling**

Pooling window 의 크기 = stride의 크기



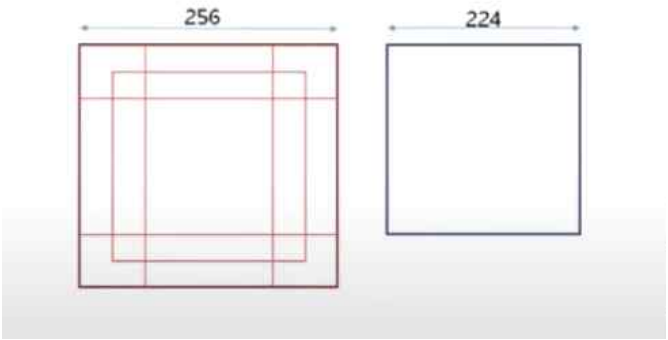
- 과대 적합 방지
- top-1, top-5 error rates를 각각 0.4%, 0.3% 감소시킴.

Prvt Overfiting(3): Data Augmentation

The easiest and most common method to reduce overfitting on image data is to artificially enlarge the dataset using label-preserving transformations (e.g., [25, 4, 5]). We employ two distinct forms of data augmentation, both of which allow transformed images to be produced from the original images with very little computation, so the transformed images do not need to be stored on disk. In our implementation, the transformed images are generated in Python code on the CPU while the GPU is training on the previous batch of images. So these data augmentation schemes are, in effect, computationally free.

Prvt Overfiting(3): Data Augmentation

* Train: Image Translation &
  Horizontal reflection &
  Flip

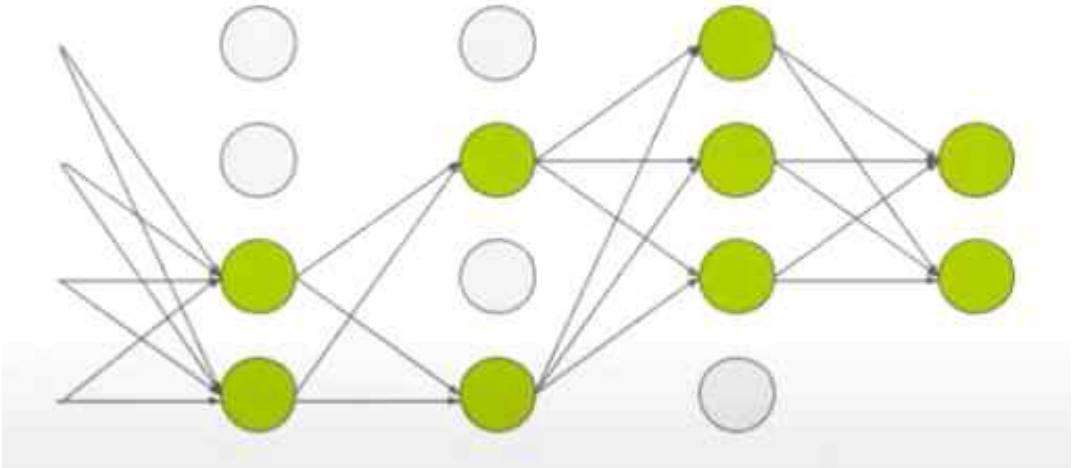


* Test: Crop & Flip augmentation

- Augmentation을 통해 과대 적합 방지
- Train data의 경우, 데이터 양 2048배 증가
- Test의 경우 10배 증가. <- 10개의 데이터의 소프트맥스 결과값을 평균을 내어 라벨 만듦

Prvt Overfiting(4): Dropout

- 0.5의 확률로 hidden neuron의 값을 0으로 바꿔줌
- Dropout된 hidden neuron은 순전파, 역전파에서 영향을 미치지 않는다.
- 뉴런들 사이의 의존성을 낮추고, co-adaptation(뉴런간의 동조화)을 피할 수 있다.

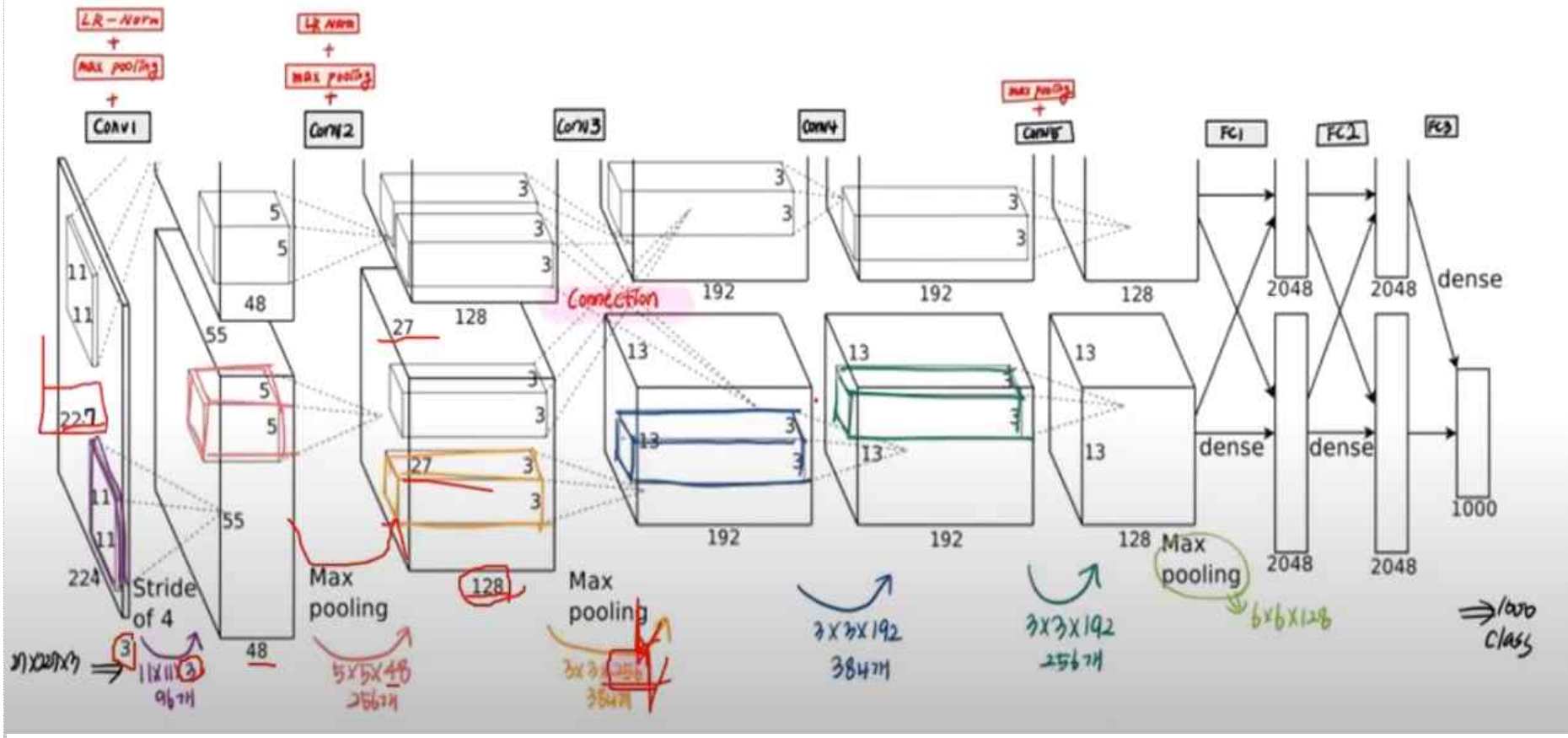


- 3개의 fully connected layers 중 앞의 2개에만 적용
- Test시에는 dropout X, 그냥 소프트맥스 결과물에다 0.5를 곱해줌

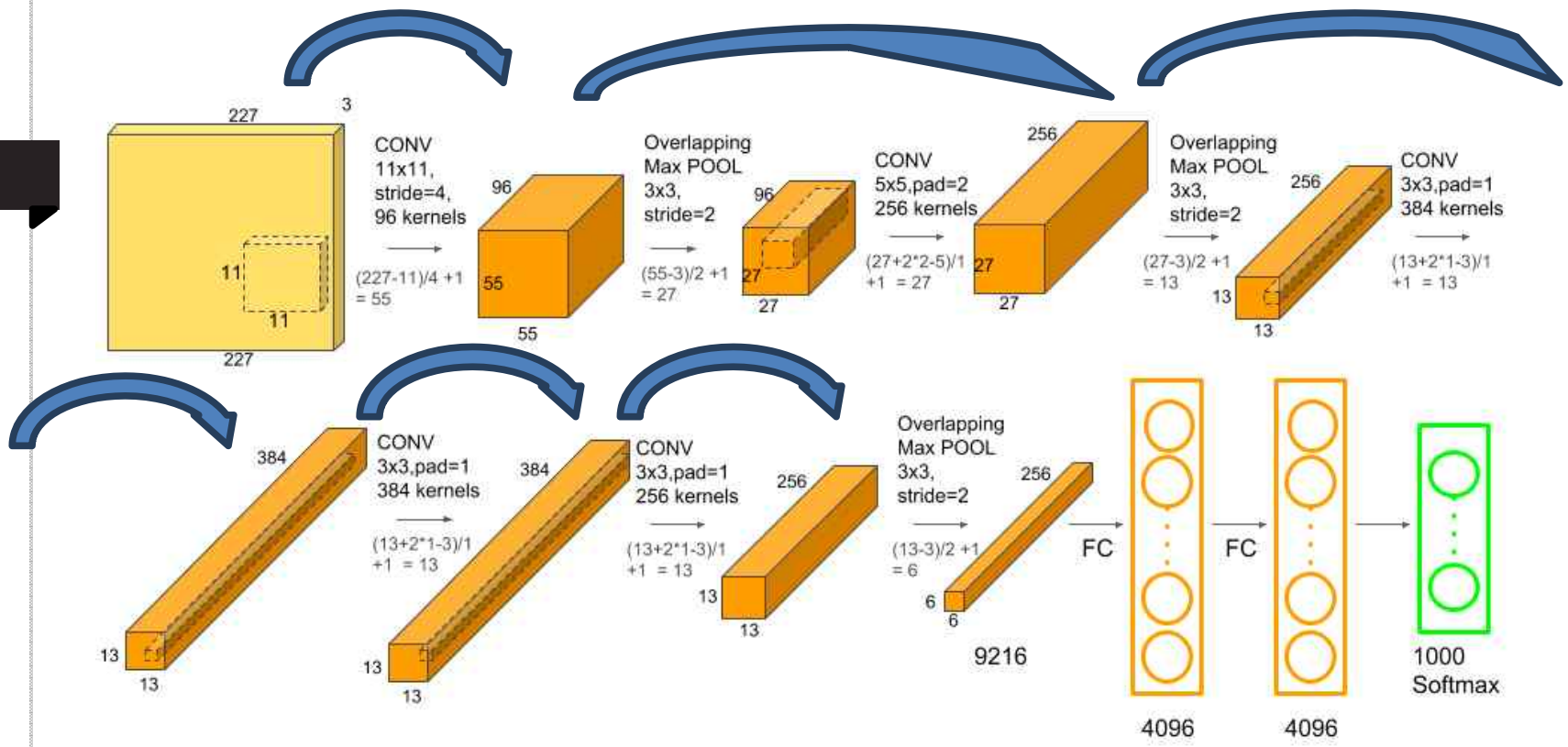
Layers Analysis

» Layers Configuration

3-5. Overall Architecture



Layers Configuration



connected layers are connected to all neurons in the previous layer. Response-normalization layers follow the first and second convolutional layers. Max-pooling layers, of the kind described in Section 3.4, follow both response-normalization layers as well as the fifth convolutional layer. The ReLU non-linearity is applied to the output of every convolutional and fully-connected layer.

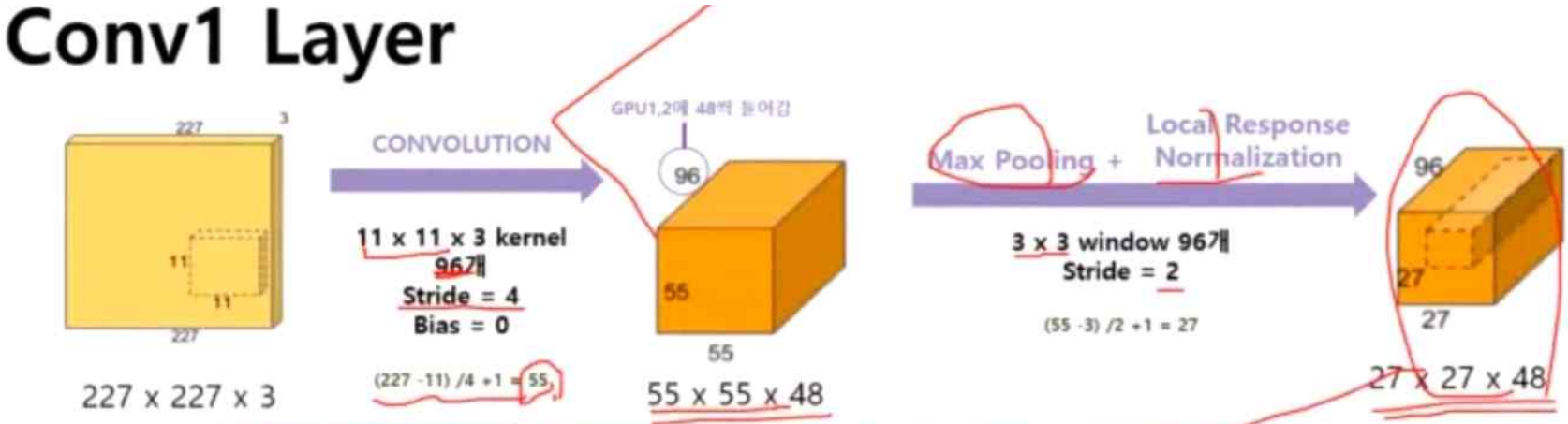
Total size ; N (i.e. N x N)

Filter size ; F (i.e. F x F)

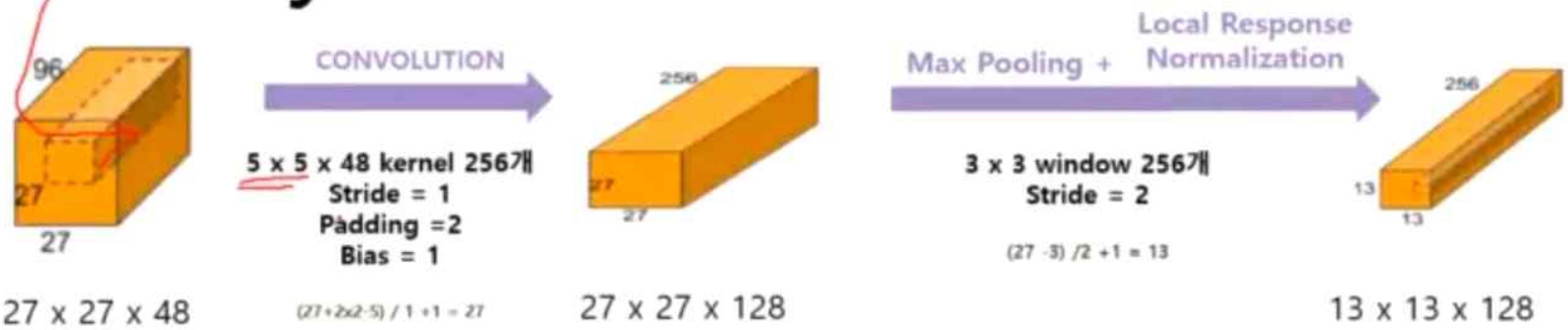
Output size ; k:=(N-F)/stride + 1 where k is an integer.

Layers Configuration

Conv1 Layer



Conv2 Layer



- # when filter size = 10
- model = Sequential()
- model.add(Conv2D(input_shape = (10, 10, 3), filters = 10, kernel_size = (3,3), strides = (1,1), padding = 'same'))
- print(model.output_shape)

» Layers Configuration

Discussion

» Details of Learning

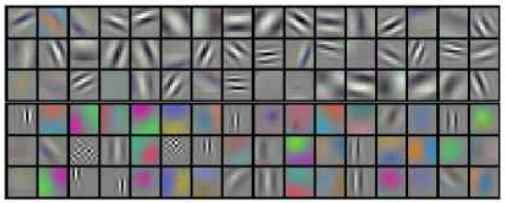
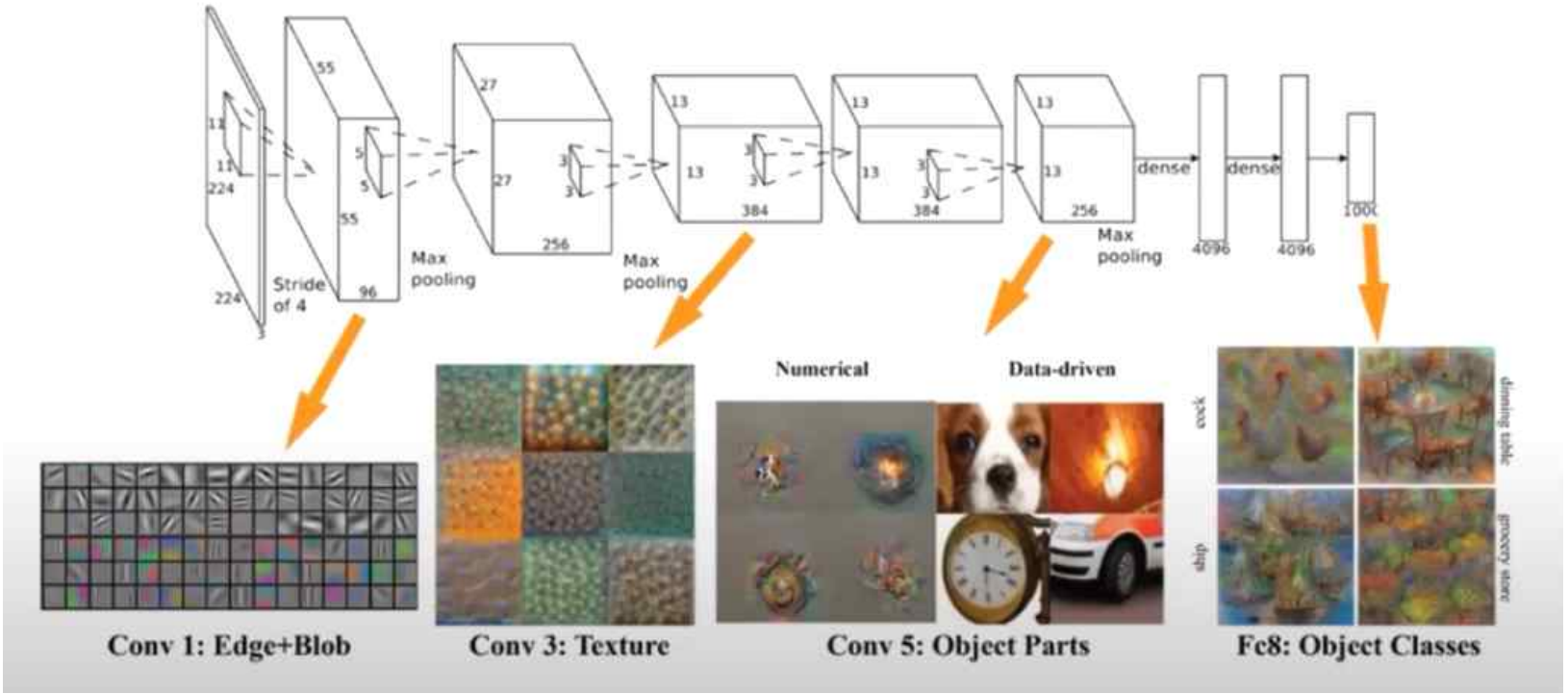


Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

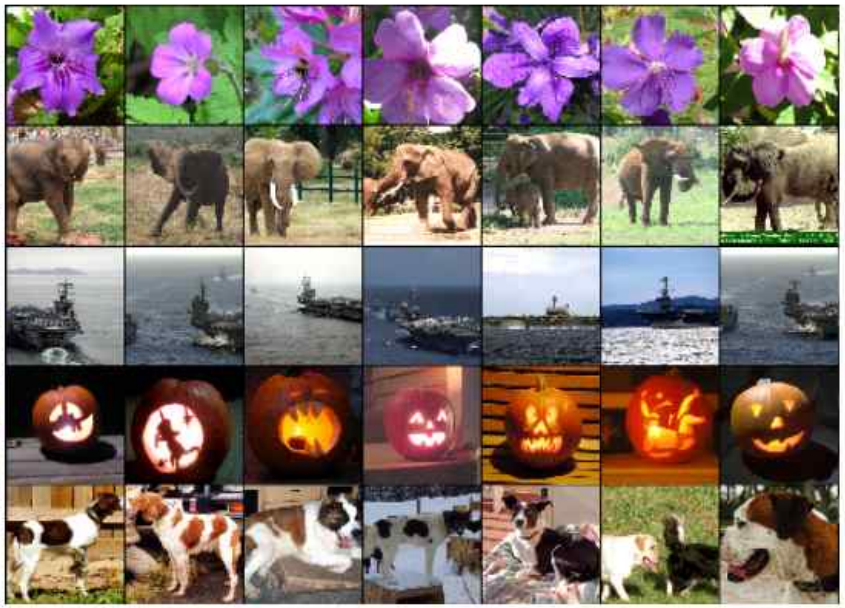
Figure 3 shows the convolutional kernels learned by the network's two data-connected layers. The network has learned a variety of frequency- and orientation-selective kernels, as well as various colored blobs. Notice the specialization exhibited by the two GPUs, a result of the restricted connectivity described in Section 3.5. The kernels on GPU 1 are largely color-agnostic, while the kernels on on GPU 2 are largely color-specific. This kind of specialization occurs during every run and is independent of any particular random weight initialization (modulo a renumbering of the GPUs).

- 서로 제한된 연결성을 가졌던 gpu결합 구조로 인해
- gpu1은 색감과 관련 없는 정보,
- gpu2는 색감과 관련 있는 정보에 특화되어 학습하였다.

» Details of Learning



» Qualitative Evaluations



- Top 5 errors에 걸렸던 사례들마저 오류가 난 것이 납득이 가는 사진들이었다.

Our results show that a large, deep convolutional neural network is capable of achieving record-breaking results on a highly challenging dataset using purely supervised learning. It is notable that our network's performance degrades if a single convolutional layer is removed. For example, removing any of the middle layers results in a loss of about 2% for the top-1 performance of the network. So the depth really is important for achieving our results.

+)데이터 양 더 많이 확보하고 computational power도 강해지면 지도 학습 이용해 보아야겠다

Reference

- https://curaai00.tistory.com/4
- https://towardsdatascience.com/implementing-alexnet-cnn-architecture-using-tensorflow-2-0-and-keras-2113e090ad98
- https://brunch.co.kr/@itschloe1/8#:~:text=%EB%94%A5%EB%9F%AC%EB%8B%9D%EA%B3%BC%20%EC%A0%84%ED%86%B5%EC%A0%81%EC%9D%B4,%EC%96%91%EC%97%90%20%EB%94%B0%EB%A5%B8%20%EC%84%B1%EB%8A%A5%EC%9E%85%EB%8B%88%EB%8B%A4.&text=%EC%99%9C%EB%83%90%ED%95%98%EB%A9%B4%20%EB%94%A5%EB%9F%AC%EB%8B%9D%20%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98%EC%9D%80,%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98%EC%9D%B4%20%EB%8D%94%20%EC%9A%B0%EC%84%B8%ED%95%A9%EB%8B%88%EB%8B%A4.
- https://www.youtube.com/watch?v=40Gdctb55BY
- https://learnopencv.com/number-of-parameters-and-tensor-sizes-in-convolutional-neural-network/

Reference

- https://m.blog.naver.com/PostView.nhn?blogId=laonple&logNo=220542170499&proxyReferer=https:%2F%2Fwww.google.com%2F

CrePAS 6th the first session

아주 상식적인 프레젠테이션 기법

Thank you

크레파스 1기 | 물고기