# ImageNet Classification with Deep Convolutional Neural Networks

OUTTA Paper Review Study

백승우

# 목차

# Introduction

- AlexNet 이전의 객체 인식 모델은 대부분 고전적인 ML 모델
- 수만개 정도의 작은 데이터셋(NORB, Caltech-101/256, CIFAR-10/100)을 사용

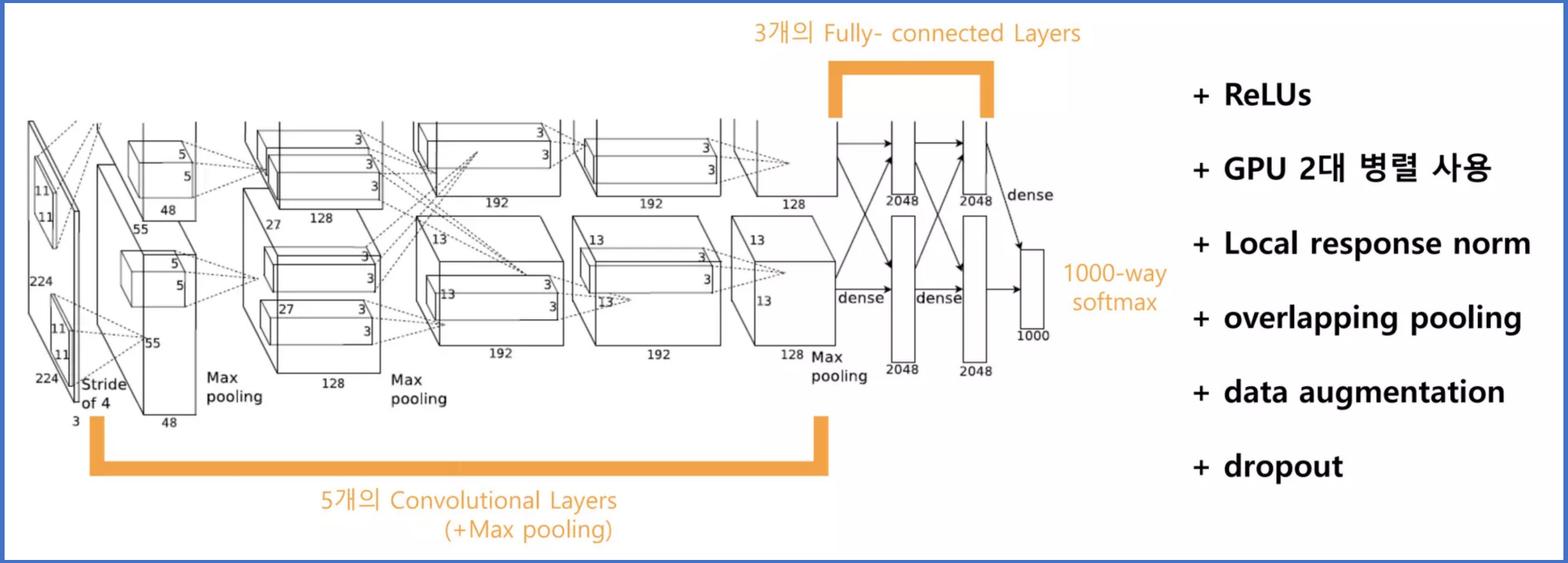
- 수십만 개의 완전 분할 된 이미지로 구성된 LabelMe 등장
- 1500 만 개 이상의 고해상도 이미지로 구성된 ImageNet 등장

# Introduction

- 등장한 데이터셋을 처리하기 위해, 높은 학습 역량을 가진 모델 필요
- 사용되지 않은 데이터에 대해서 추론을 할 수 있는 사전 지식을 담아내야 함

- 이에 논문은 컨볼루션 신경망(CNN) 모델을 기반으로 하는 AlexNet 을 제시

# Overall Architecture




6천만개의 Parameters, 650,000개의 Neuron을 가지며 8개의 Layer 로 이루어져 있음

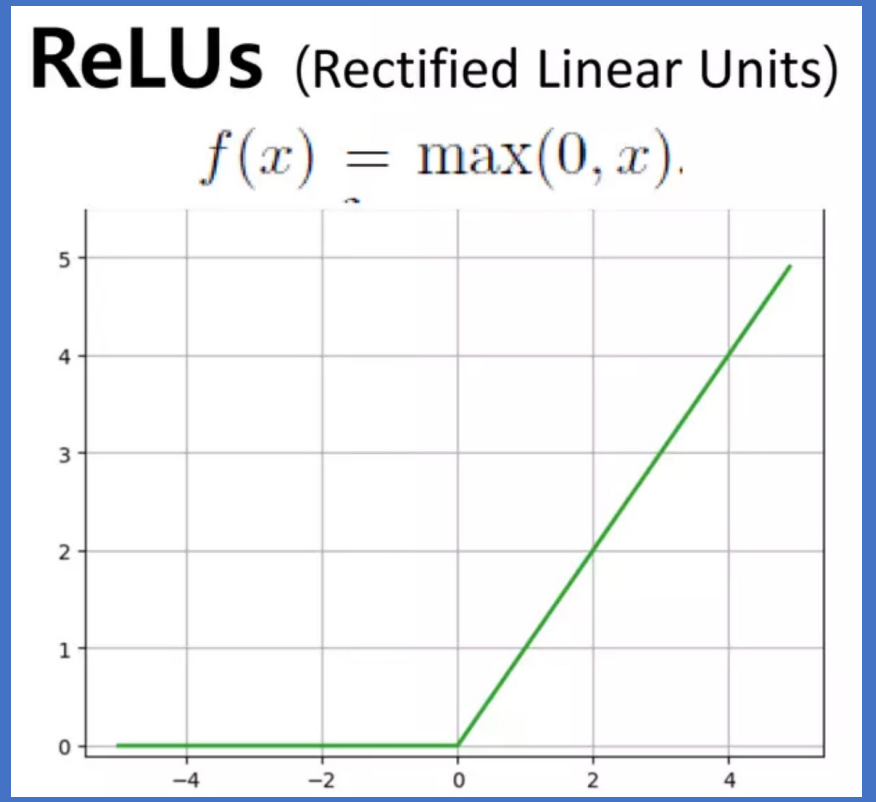
# The Dataset

- 1000개의 이미지 카테고리 각각에 약 1000개의 이미지
- Test Set이 라벨링된 ILSVRC-2010 데이터를 주로 사용
- 120만개의 Train Set, 50,000개의 Validation Set, 150,000개의 Test Set 사용
- 256 × 256의 고정 해상도로 다운 샘플링을 수행
- 직사각형 이미지는 scaling 후 중앙 256x256 패치로 잘라냄

# The Architecture (ReLU Nonlinearity)

**Traditional Activation Fuction**

- Tanh
- Sigmoid



# The Architecture (ReLU Nonlinearity)

## ReLUs VS Tanh

- 둘다 정규화X
- Learing Rate는 각자 최적으로 맞춤

## ReLUs가 약 6배 빨리 수렴

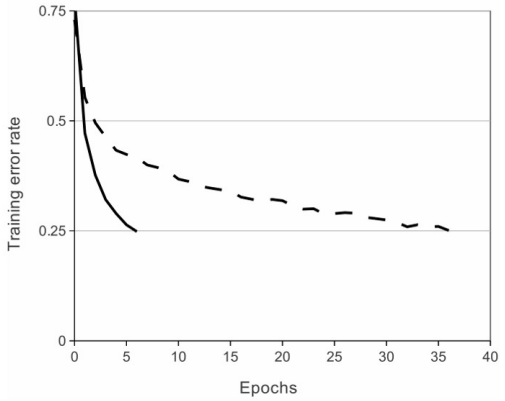
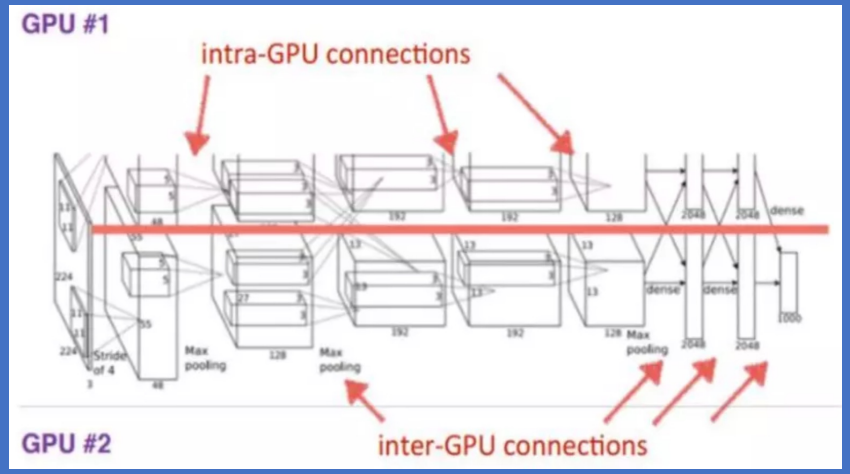



Figure 1: A four-layer convolutional neural network with ReLUs **(solid line)** reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with tanh neurons **(dashed line)**. The learning rates for each network were chosen independently to make training as fast as possible. No regularization of any kind was employed. The magnitude of the effect demonstrated here varies with network architecture, but networks with ReLUs consistently learn several times faster than equivalents with saturating neurons.

error rate가 0.25에 도달하기 위한 epoch 수

# The Architecture (Training on Multiple GPUs)

- GPU 메모리 제한과 느린 학습 속도 개선을 위한 병렬학습 방법 제안




- 네트워크를 분할하여 서로 다른 GPU에서 병렬적으로 연산을 수행(특정 레이어에서만)

- Top-1/Top-5 error rate를 1.7%/1.2% 절감

# The Architecture (Local Response Normalization)

- 실제 뇌세포의 증상인 Lateral Inhibition(강한 뉴런의 활성화가 다른 뉴런의 활동 억제) 이용

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

- CNN에서 인접한 필터를 사용하여 normalization을 진행
  (논문에서는, k=2, n=5, α=10^(-4), β=0.75 사용)

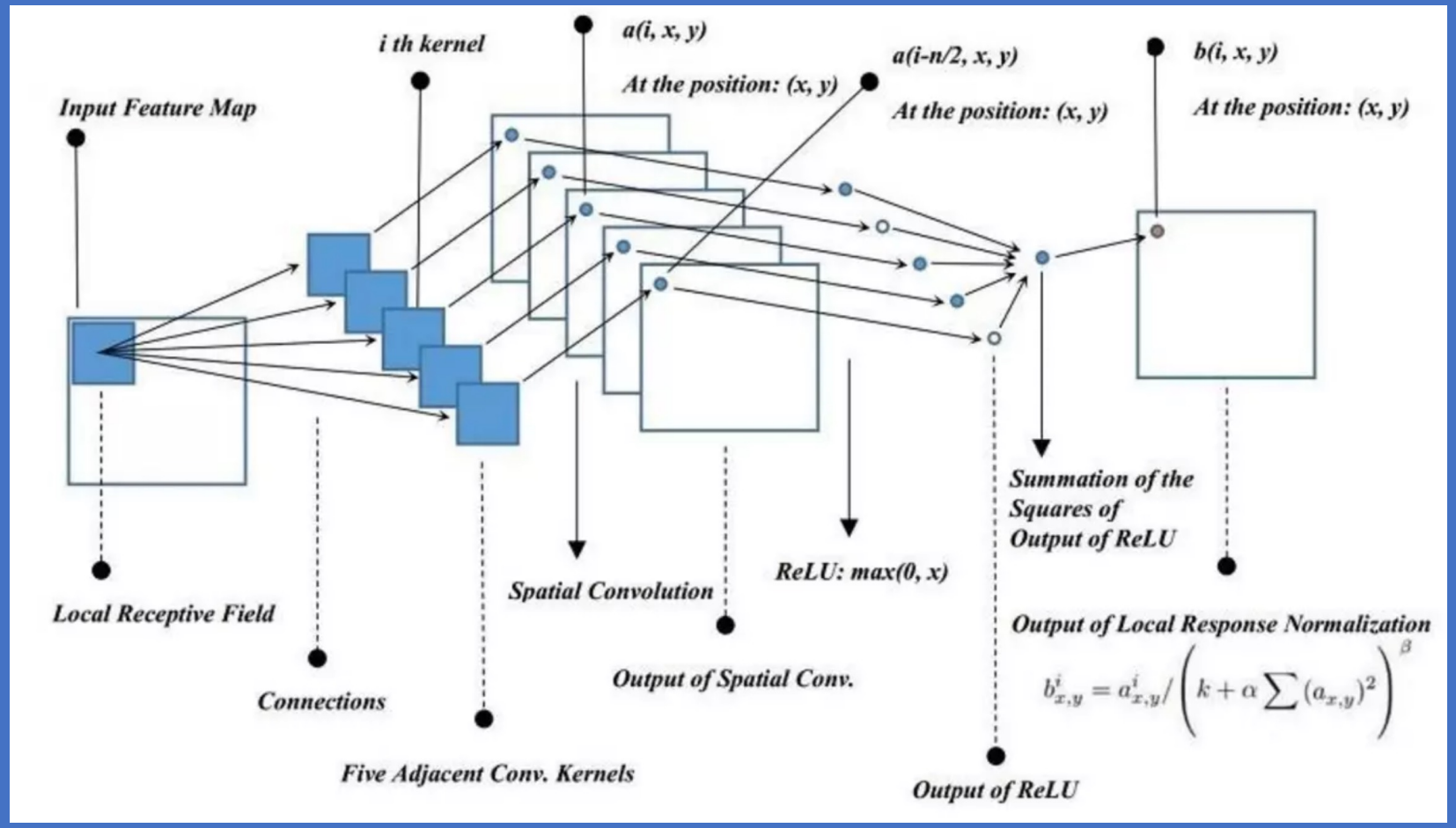
- Top-1/Top-5 error rate를 1.4%/1.2% 개선

**LRN을 사용하는 이유**

- ReLU는 양수 방향으로 입력값을 그대로 사용함

- Convolution이나 Pooling시, 매우 높은 하나의 픽셀값이 주변의 픽셀에 영향을 미침

# The Architecture (Local Response Normalization)




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

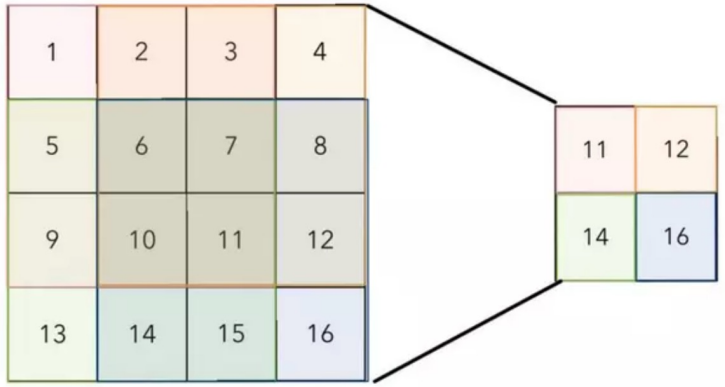
현재 10번째 커널이라면,

8번째 커널부터 12번째 커널까지의 제곱합으로 나누며 정규화를 진행

# The Architecture (Overlapping Pooling)

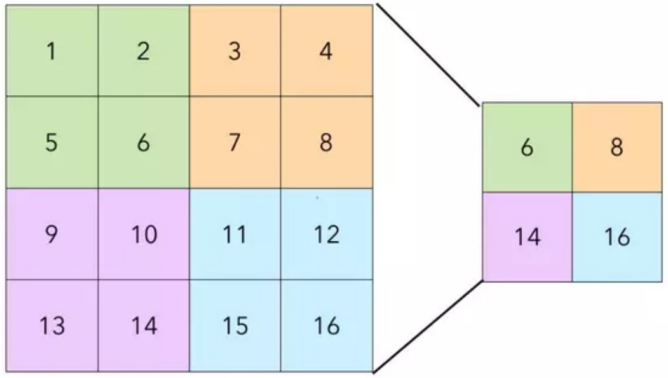
- **Overlapping pooling**

  Pooling window 의 크기 > stride의 크기




- **Traditional pooling**

  Pooling window 의 크기 = stride의 크기




- Top-1/Top-5 error rate를 0.4%/0.3% 개선

# The Architecture (Overall Architecture)

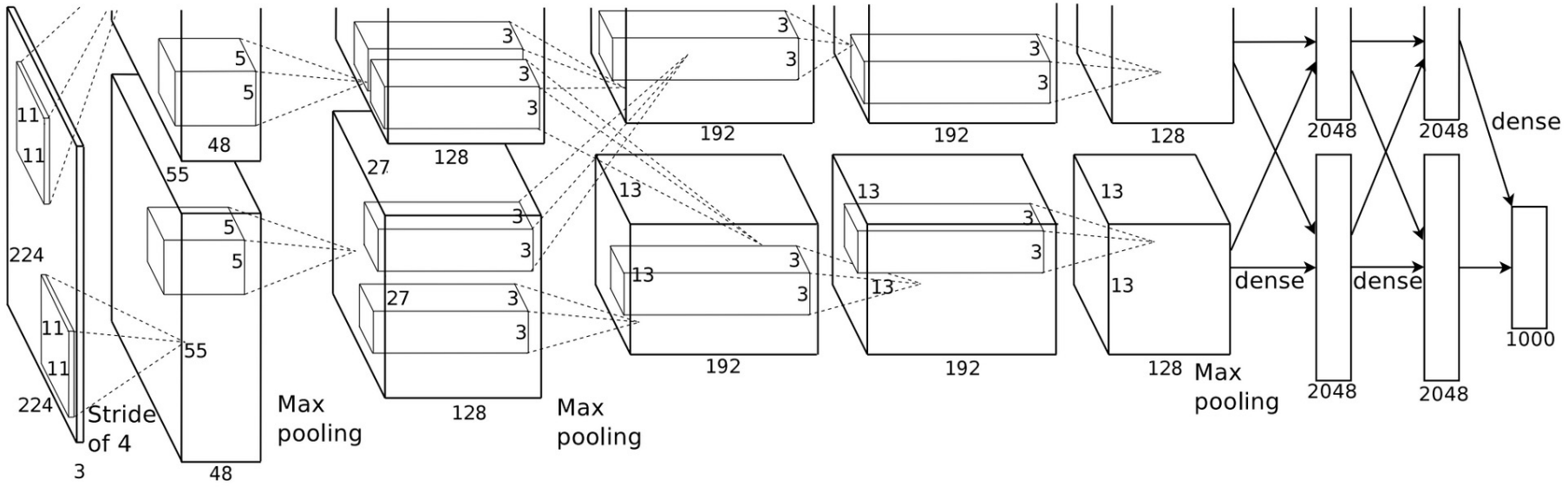
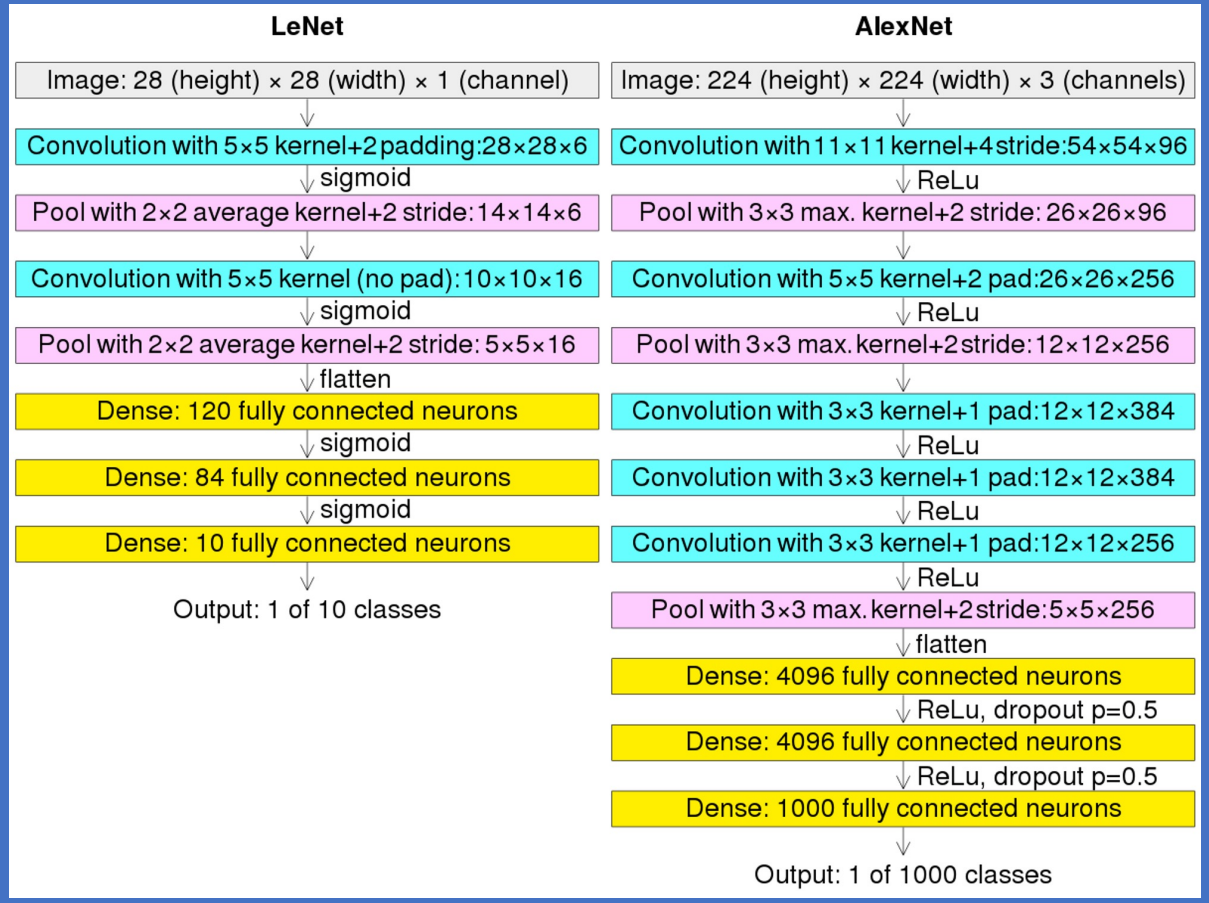



Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

# The Architecture (Overall Architecture)



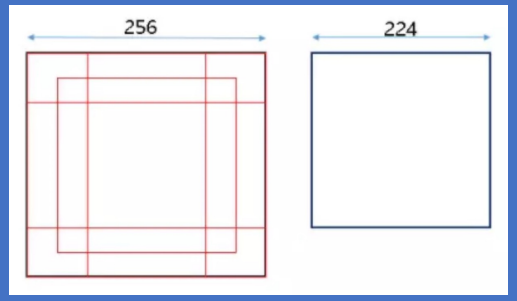
# Reducing Overfitting (Data Augmantation)

**(1) 256 × 256 이미지에서 224 × 224 패치를 추출하고, 좌우반전**

- 좌우반전하여 이미지 양을 2배로 증가
- 256 x 256 이미지를 랜덤으로 잘라 224 x 224로 만들어 1024배 증가
- 기존 데이터 셋의 2048배[(256 – 224) * (256 - 224) *2] 확장

- 실제

5개의 224 × 224 패치 (4 개의 코너 패치 및 중앙 패치)와
수평 반사를 수행한 10개의 패치 사용



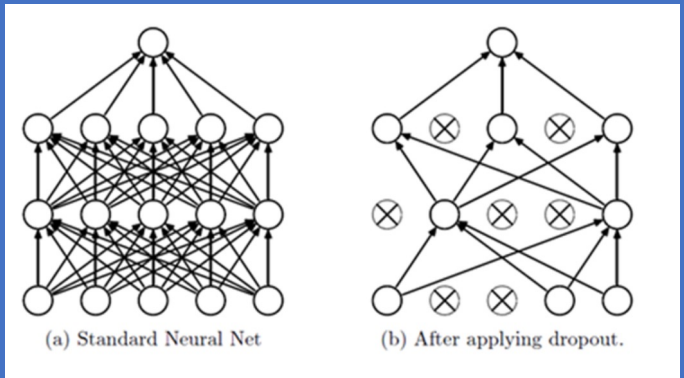

# Reducing Overfitting (Data Augmantation)

**(2) RGB 채널 강도 조정**

- 학습 데이터셋의 픽셀값으로 PCA 를 수행
- PCA eigenvector 에 N(0,0.1) 인 정규분포에 추출한 랜덤값을 곱해 색상을 조정
- Top-1 error rate가 1% 감소

# Reducing Overfitting (DropOut)

- 0.5의 확률로 Hidden Neuron의 값을 0으로 바꿔줌
- (a)와 같이, 망에 있는 모든 Layer에 대해 학습을 수행하는 것이 아님
- (b)와 같이, 망에 있는 일부 뉴런을 생략(Drop Out)하고 줄어든 신경망을 통해 학습 수행




- 3개의 Fully-Connected Layer 중 앞의 2개에만 적용
- 실제: DropOut 적용X, 대신 0.5를 곱해줌

# Details of Learning

- Batch Size = 128
- Momentum = 0.9
- Weight Decay = 0.0005
  - Training Error를 감소시킴
  - N(0, 0.01)에서 랜덤 추출해 가중치를 초기화
- Learning Rate = 0.01
  - Validation Error가 계속 변하지 않으면 0.1을 곱함
  - 실험 중 3번 바뀜
- Epochs = 90

$$v_{i+1} := 0.9 \cdot v_i - 0.0005 \cdot \epsilon \cdot w_i - \epsilon \cdot \left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i}$$

$$w_{i+1} := w_i + v_{i+1}$$

where $i$ is the iteration index, $v$ is the momentum variable, $\epsilon$ is the learning rate, and $\left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i}$ is the average over the $i$th batch $D_i$ of the derivative of the objective with respect to $w$, evaluated at $w_i$.

the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

# Results

| Model | Top-1 | Top-5 |
|---|---|---|
| *Sparse coding [2]* | *47.1%* | *28.2%* |
| *SIFT + FVs [24]* | *45.7%* | *25.7%* |
| CNN | **37.5%** | **17.0%** |

Table 1: Comparison of results on ILSVRC-2010 test set. In *italics* are best results achieved by others.

| Model | Top-1 (val) | Top-5 (val) | Top-5 (test) |
|---|---|---|---|
| *SIFT + FVs [7]* | — | — | *26.2%* |
| 1 CNN | 40.7% | 18.2% | — |
| 5 CNNs | 38.1% | 16.4% | **16.4%** |
| 1 CNN* | 39.0% | 16.6% | — |
| 7 CNNs* | 36.7% | 15.4% | **15.3%** |

Table 2: Comparison of error rates on ILSVRC-2012 validation and test sets. In *italics* are best results achieved by others. Models with an asterisk* were "pre-trained" to classify the entire ImageNet 2011 Fall release. See Section 6 for details.

# Results (Qualitative Evaluations)

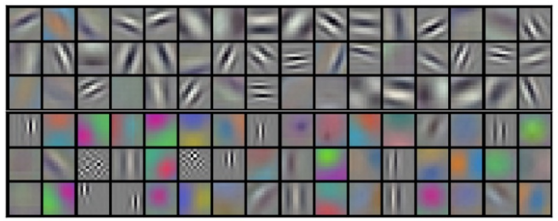


Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

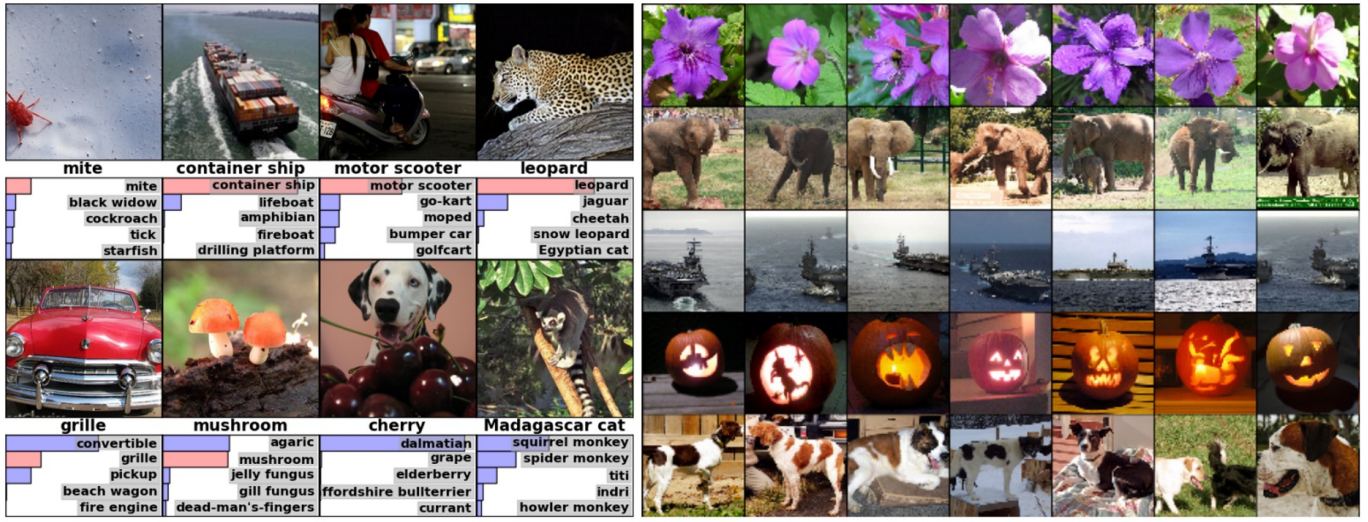



Figure 4: **(Left)** Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). **(Right)** Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.