

# VGG

## 0. Abstract

1.  $3 \times 3$  (작은 filter) ConvNet +  $\frac{2}{3} \uparrow = 2\text{倍} \uparrow$   
(11, 13, 16, 19 실험)
2. 2014: ImageNet challenge  $\rightarrow 2^{nd}$  16, 19가 가장 성능↑

## 1. Introduction

1. AlexNet 구조 틀   
 (2012) zeiler & fergus 2013, overfeat 2014  
→ ILSVRC-2013

: AlexNet의 11번 Conv의 receptive field 크기와 stride가  
해서 이미지를 한번에 너무 축소시키는 문제 ↳ 2013  
Sermanet 2014, Howard 2014  
: 미리 이미지의 어떤 정보를 흐트려 넣을 수 있는지를 multi-scale 적용 → Multi-scale input 모양을 dense하게 보여주면  
= Multi-scale (HxW)

↳

	resize	multi-scale	crop
Sermanet 2014	○	×	○ (위치 전달, 크기는 고정) × (대신 슬라이딩 윈도우)
	○	○	
Howard 2014	○	×	○ (위치 전달, 크기는 고정)
	○	○	○ (크기, 위치 고정 고려)

ex)  $600 \times 900$

↓  
Resize(480) (rescaling: 224, 284 ...)  
↳ 작은 변 (600)을 480으로 하되, 종횡비 유지  
↓  
Crop

⇒ 다른 기법: "  $\frac{2}{3} \uparrow + 3 \times 3$  (작은 필터)" (VGG)

## 2. ConvNet Configuration

1. AlexNet 예측 구조, 가중치 등

### 2.1 Architecture

1. Img

$\begin{bmatrix} \text{size: } 224 \times 224 \\ \text{Channel: } 3(\text{RGB}) \\ \text{kernel: } ? \text{ 디멘지션은 평균 RGB 값인가?} \end{bmatrix}$

2. CONV 1<sub>64</sub> - receptive field:  $\frac{9 \times 9}{1} \rightarrow$  9x9 (1x1은 224가 됨)

- stride: 1 → 모든 퍼셉트론이 conv 적용 가능

-  $3 \times 3$ 은 padding = 1 ( $\text{output\_height} = \text{input\_height} + 2 \times \text{padding} - \text{kernel\_size} + 1$ ):  $\frac{(224-3+2 \times 1)}{1} + 1 = 224$

3. Max-pooling 예측: 일부 Conv 뒤에 (5x5)

kernel-size: 2, stride: 2

4. 37M fc → 4096 + softmax

- 4096 → 4096

- 4096 → 1000  
(num\_classes)

5. activation func: ReLU

6. LRN (Local Response Normalization): AlexNet 설정 참고

⇒ “모두 공통: Net 깊이만 다름”

### 2.2 Configurations

Table 2: Number of parameters (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

→ 얕고 넓은 (receptive field = 필드 크기 ↑) net 뒷부분 터무니 ↓

### 2.3 Discussion

1.  $3 \times 3$  커널 수  $\frac{224}{1} \rightarrow (224-3+1=222 \rightarrow 220 \rightarrow 218)$

$$= 1 \times 1 \quad 17n \quad // \quad (\frac{224-3}{1}+1=218)$$

↳  $3 \times 3$  커널 수  $\frac{224}{1}$

1) 비선형성 증가: 각 3개 계층 후 ReLU

2) 터무니 수(가중치) 감소:  $37M \times (3^2 \times C^2) = 27C^2 \ll 17n \times (1^2 \times C^2) = 49C^2$

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
conv3-64	conv3-64	conv3-64	conv3-64	conv3-64	conv3-64
LRN	conv3-64	conv3-64	conv3-64	conv3-64	conv3-64
maxpool					
conv3-128	conv3-128	conv3-128	conv3-128	conv3-128	conv3-128
conv3-128	conv3-128	conv3-128	conv3-128	conv3-128	conv3-128
maxpool					
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
conv1-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
maxpool					
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
conv1-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
maxpool					
FC-4096	FC-4096	FC-4096	FC-4096	FC-4096	FC-4096
FC-4096	FC-4096	FC-4096	FC-4096	FC-4096	FC-4096
FC-1000	FC-1000	FC-1000	FC-1000	FC-1000	FC-1000
soft-max	soft-max	soft-max	soft-max	soft-max	soft-max

$$\hookrightarrow \text{공식: } C_{\text{out}} \times C_{\text{in}} \times \frac{k_h}{T} \times k_w$$

↓  
kernel size

3) 구조적 제약 = 규칙화

→ f.m. 구현과 같은 것을 가져야 만들어짐

→ "규칙화"; 과적합 감소 가능

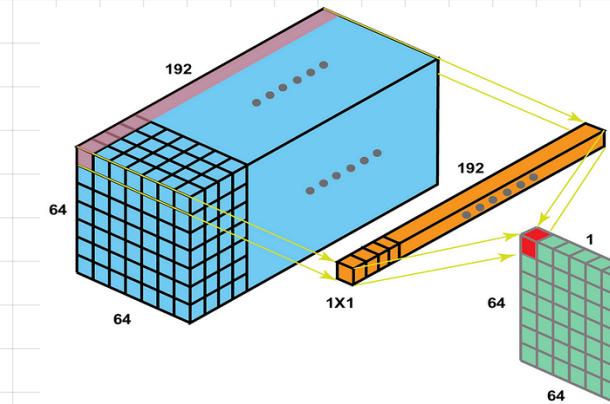
## 2. $1 \times 1$ 층

$$\rightarrow y_{i,j} = \underbrace{W x_{i,j} + b}_{= \text{필터}} \quad (\text{선형변환})$$

→  $1 \times 1$  층은 CONV 계산은 흥, 이때  $H \times W$ 는 고려로 이고, 차별만 설정

→ + 1d 선형성 ( $1 \times 1$  자체는 선형화(ReLU))

3. GoogLeNet은 보다 복잡, 계산량 줄이기 위해 초기에  $H \times W$  차원 감소



## 3. Classification Framework

### 3.1 Training

1. Loss: multinomial Logistic Regression

= logits of softmax activation func  $\approx$ ただし  $w$ 는 계수들의 Cross-Entropy를 최소화

$$\hookrightarrow \sum_{i=1}^N \sum_{k=1}^K y_{ik} \ln \frac{e^{z_{ik}}}{\sum_{j=1}^K e^{z_{ij}}}$$

logit:  $z_{ik} = w_k^T x_i + b_k$

softmax:  $P_\theta(y=k|x_i) = \frac{e^{z_{ik}}}{\sum_{j=1}^K e^{z_{ij}}} = q_{ik}$

Loss:  $L(\theta) = -\sum_{i=1}^N \sum_{k=1}^K y_{ik} z_{ik} + \sum_{i=1}^N \log \sum_{j=1}^K e^{z_{ij}}$  = 대푯값  $(x_i, y_i)$ 의 NLL (CE)

$\hookrightarrow$  cross-entropy loss (NLL):  $CE = -\sum_{i=1}^N \sum_{k=1}^K y_{ik} \log q_{ik}$  (정답 확률, 예측 확률)

$$CE = -\sum_{i=1}^N \sum_{k=1}^K y_{ik} \left[ z_{ik} - \log \sum_{j=1}^K e^{z_{ij}} \right]$$

$$= -\sum_{i,k} y_{ik} z_{ik} + \sum_{i,k} y_{ik} \cdot \log \sum_j e^{z_{ij}}$$

$$= \sum_i \left[ \sum_k y_{ik} \cdot \log \sum_j e^{z_{ij}} \right] / \sum_i y_{ik}$$

$$= \sum_i \left[ 1 \cdot \log \sum_j e^{z_{ij}} \right] / \sum_i y_{ik}$$

$$= \sum_i \log \sum_{j=1}^K e^{z_{ij}}$$

$$= -\sum_{i=1}^N \sum_{k=1}^K y_{ik} z_{ik} + \sum_{i=1}^N \log \sum_{j=1}^K e^{z_{ij}}$$

## 2. Optimizer: mini-batch gradient descent

batch 평균  $\bar{w}_k \leftarrow$   
 $\text{Momentum}(\beta=0.9)$   
 $(\text{epoch } n)$   
 $+ \text{weight decay} (= L_2 \text{ penalty}, 5 \times 10^{-4})$

$\Rightarrow \text{optim.SGD}(\text{lr}=0.01, \text{momentum}=0.9, \text{weight\_decay}=5e-4)$

3. batch size: 256

4. dropout(0.5): FC 초기 27n

5. lr:  $10^{-2} \rightarrow 10\text{번} 3\text{번} \downarrow$

6. epochs: 74 (AlexNet 보다 훨씬 더 깊이↑ but, 평균↓(정규화) + 가중치 사전 초기화)

7. 가중치 초기화: 누적 gradient 평균

$\rightarrow \text{Net} \rightarrow \text{가중치 초기화 후} \rightarrow \text{연결 수} \downarrow \text{가중치 학습}$

$\rightarrow$  각각의 Conv, 3fc 가중치는 더 깊은 Net에서 초기 가중치로 설정 (계속 미세 조정 / 누적되는 가중치)  
 $M=0, \sqrt{\frac{1}{M}}=10^{-2} \rightarrow$  초기화 평균, 평균 = 중앙값 = 평균값

$\frac{\text{누적}}{\text{가중치}} = \frac{\text{평균}}{\text{가중치}}$

$\frac{\text{누적}}{\text{가중치}} = \frac{\text{평균}}{\text{가중치}} = \frac{\text{평균}}{\text{가중치}} = \frac{\text{평균}}{\text{가중치}}$

8. img size  $224 \times 224$

↓  
 training Rescaling: 원본( $S=224$ )을  $S$ 로 하고 정비유지,  $S \geq 224$ )

$\hookrightarrow S=224 \rightarrow S$ 로 정비유지,  $S > 224 \rightarrow S$ 로 정비유지,  $S < 224 \rightarrow S$ 로 정비유지

random crop (224 고정)

(SGD iteration, 즉 loader for문 DCTC img = transform(img))

+

Crop된 img [ RandomHorizontalFlip(p)  
 ColorJitter() ]

$\rightarrow S$  설정

1) single scale

$\rightarrow S=256$  (256, 384 이용)

$\rightarrow S=256$  학습 후  $S=384$ 는 학습 속도를 옵티마

$S=256$ 의 가중치를 초기값으로 하고  $lr=1e-3$

2)  $S=384$  DCTC

2) multi-scale

$\rightarrow [256, 384]$  random rescaling

$\rightarrow 384$  기반 가중치 초기값으로 하고 학습하는 경우 더 좋음

## \* weight initialization

$\text{fan}_{\text{in}} = C_{\text{in}} \times k_{\text{in}}^2 = N_{\text{in}} = \text{입력 뉴런 수}$  (FC:  $k_{\text{in}}=k_{\text{out}}=1$ )  
 $\text{fan}_{\text{out}} = C_{\text{out}} \times k_{\text{out}}^2 = N_{\text{out}} = \text{출력 뉴런 수}$

### 1) Gaussian

$\rightarrow$  스케일 조정  $\times$  작은 분산 + 단순 모작위

$\rightarrow$  소신/폭포 위험

### 2) Xavier

$\rightarrow \text{sigmoid, tanh} \text{ 평균}$

$\rightarrow$  전등분포:  $W \sim U[-\sqrt{\frac{6}{n_{\text{in}}+n_{\text{out}}}}, \sqrt{\frac{6}{n_{\text{in}}+n_{\text{out}}}}]$

$\rightarrow$  정규분포:  $W \sim N(0, \frac{2}{n_{\text{in}}+n_{\text{out}}})$   
 $\downarrow M \quad \frac{1}{\sqrt{2}}$

### 3) He/Kaiming

$\rightarrow \text{ReLU}$

$\rightarrow$  전등분포:  $W \sim U[-\sqrt{\frac{6}{n_{\text{in}}}}, \sqrt{\frac{6}{n_{\text{in}}}}]$

$\rightarrow$  정규분포:  $W \sim N(0, \frac{2}{n_{\text{in}}})$   
 $\downarrow M \quad \frac{1}{\sqrt{2}}$

## 3.2 Testing

0. 원본 + 수평 대칭 ( $\frac{1}{2}$ )

1. 짧은 변을 Q로 하는 등방성 리스케일링 (정비유지)

2. densely net 적용 (학습의 crop 대신)

$\rightarrow$  원본 이미지에 fully-convolutional net

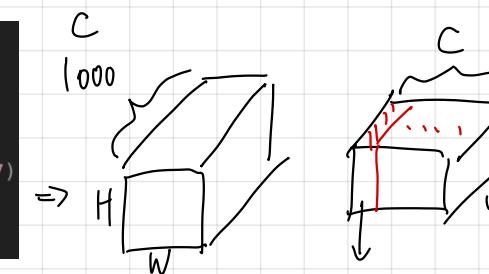
$fcl \rightarrow 1 \times 1 \text{ conv}$

$fcl, 3 \rightarrow 1 \times 1 \text{ conv}$

↓  
 짧은 변을 map (채널 수 = 층 수, 즉 2nd fc처럼 logits)

$\rightarrow$  글로벌 합성 "batch"  $\rightarrow \text{softmax}$

```
nn.Linear(512*7*7, 4096)
nn.Linear(4096, 4096)
nn.Linear(4096, 1000)
↓
nn.Conv2d(512, 4096, kernel_size=7)
nn.Conv2d(4096, 4096, 1)
nn.Conv2d(4096, 1000, 1)
```



↑ 짧은 변에 대한 "Spatially averaged (sum-pooled)"  
 $\frac{1}{H} \times \frac{1}{W}$

→ AlexNet이 ILSVRC-2012 test set에 대한 성능을 보여주는 예제 네트워크 실행 X

↳ AlexNet: multi-crop testing vs Vgg: dense evaluation

↓

Crop 방식으로 이미지 처리  
특정 부분을 사용한 평균  
→ 한 번 훈련

가운데 이미지에서 Conv3  
슬라이딩 윈도우  
→ 이미지 쪽을 사용 → 수용 영역 (receptive field↑)

↓

## 4. Classification Experiments

1. ILSVRC-2012 (class 1000, train 1.2M, Val 50k, test 100k)

→ Top-1 error: 잘못 분류된 이미지 비율

Top-5 error: 예측된 상위 5개 클래스 내 정답 클래스가 포함 X 비율

### 4.1 Single scale Evaluation

2<sup>nd</sup> ( $S=Q$ ) / Scale Jittering:  $S \in [S_{\min}, S_{\max}]$  일 때,  
 $Q = 0.5(S_{\min} + S_{\max})$

↳ 사용 확률은 평균으로

Table 3: ConvNet performance at a single test scale.

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train ( $S$ )	test ( $Q$ )		
A	256	256	29.6	10.4
A-LRN	256	256	29.7	10.5
B	256	256	28.7	9.9
C (1x1 nn)	256	256	28.1	9.4
	384	384	28.1	9.3
	[256;512]	384	27.3	8.8
D (3x3 nn)	256	256	27.0	8.8
	384	384	26.8	8.7
	[256;512]	384	25.6	8.1
E	256	256	27.3	9.0
	384	384	26.9	8.7
	[256;512]	384	<b>25.5</b>	<b>8.0</b>

1. LRN 효과 (LRN) : X (A > A-LRN) → B ~ E의 LRN X

2. A → E : 깊이 짧아 오류 ↓

3. C < D : 동일한 깊이 → 비슷한 속도로 유의미한 수용영역 필요  
중간단계 문제 없음

4. Scale Jittering: 성능 ↑ (: 학습 증강)

④ B: 3x3 2m → 5x5 1m 단위 시 error ↑

## 4.2 Multi scale Evaluation

Table 4: ConvNet performance at multiple test scales.

ConvNet config. (Table 1)	smallest image side train ( $S$ )	test ( $Q$ )	top-1 val. error (%)	top-5 val. error (%)
B	256	224,256,288	28.2	9.6
	256	224,256,288	27.7	9.2
C	384	352,384,416	27.8	9.2
	[256; 512]	256,384,512	26.3	8.2
D	256	224,256,288	26.6	8.6
	384	352,384,416	26.5	8.6
E	[256; 512]	256,384,512	<b>24.8</b>	<b>7.5</b>
	256	224,256,288	26.9	8.7
F	384	352,384,416	26.7	8.6
	[256; 512]	256,384,512	<b>24.8</b>	<b>7.5</b>

1. 향습 size 증가 시 test val. scale jittering step 증가 ↓

$$\rightarrow Q = S-32, S, S+32$$

2. 향습 시 jittering :  $[S_{min}, S_{max}] \rightarrow \text{test: } (S_{min}, 0.5(S_{min}+S_{max}), S_{max})$

⇒ 4.1 Table 3 를 참고 (Test scale Jittering 예시)

## 4.3 Multi-CROP Evaluation

Table 5: ConvNet evaluation techniques comparison. In all experiments the training scale  $S$  was sampled from [256; 512], and three test scales  $Q$  were considered: {256, 384, 512}.

ConvNet config. (Table 1)	Evaluation method	top-1 val. error (%)	top-5 val. error (%)
D	dense	24.8	7.5
	multi-crop	24.6	7.5
	multi-crop & dense	<b>24.4</b>	<b>7.2</b>
E	dense	24.8	7.5
	multi-crop	24.6	7.4
	multi-crop & dense	<b>24.4</b>	<b>7.1</b>

↳ multi-crop eval vs dense eval

평가: >  $\downarrow$  ← dense & crop(0.0%) 때 자연스러운 유행 추세를 갖지만  $M_2$  높은  
제작  $M_2$  ←  $M_1$  높은

## 4.4 ConvNet Fusion

Table 6: Multiple ConvNet fusion results.

Combined ConvNet models	Error		
	top-1 val	top-5 val	top-5 test
<b>ILSVRC submission</b>			
(D/256/224,256,288), (D/384/352,384,416), (D/[256;512]/256,384,512) (C/256/224,256,288), (C/384/352,384,416) (E/256/224,256,288), (E/384/352,384,416)	24.7	7.5	7.3
<b>post-submission</b>			
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), dense eval. (D/[256;512]/256,384,512), (E/[256;512]/256,384,512), multi-crop (D/[256;512]/256,384,512), (E/[256;512]/256,384,512), multi-crop & dense eval.	24.0	7.1	7.0
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), multi-crop & dense eval.	23.9	7.2	-
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), multi-crop & dense eval.	<b>23.7</b>	<b>6.8</b>	<b>6.8</b>

↳ 여러 모델 평균화

## 4.5 Comparison with the state of the art

Table 7: Comparison with the state of the art in ILSVRC classification. Our method is denoted as "VGG". Only the results obtained without outside training data are reported.

Method	top-1 val. error (%)	top-5 val. error (%)	top-5 test error (%)
VGG (2 nets, multi-crop & dense eval.)	<b>23.7</b>	<b>6.8</b>	<b>6.8</b>
VGG (1 net, multi-crop & dense eval.)	24.4	7.1	7.0
VGG (ILSVRC submission, 7 nets, dense eval.)	24.7	7.5	7.3
GoogLeNet (Szegedy et al., 2014) (1 net)	-	-	7.9
GoogLeNet (Szegedy et al., 2014) (7 nets)	-	-	<b>6.7</b>
MSRA (He et al., 2014) (11 nets)	-	-	8.1
MSRA (He et al., 2014) (1 net)	27.9	9.1	9.1
Clarifai (Russakovsky et al., 2014) (multiple nets)	-	-	11.7
Clarifai (Russakovsky et al., 2014) (1 net)	-	-	12.5
Zeiler & Fergus (Zeiler & Fergus, 2013) (6 nets)	36.0	14.7	14.8
Zeiler & Fergus (Zeiler & Fergus, 2013) (1 net)	37.5	16.0	16.1
OverFeat (Sermanet et al., 2014) (7 nets)	34.0	13.2	13.6
OverFeat (Sermanet et al., 2014) (1 net)	35.7	14.2	-
Krizhevsky et al. (Krizhevsky et al., 2012) (5 nets)	38.1	16.4	16.4
Krizhevsky et al. (Krizhevsky et al., 2012) (1 net)	40.7	18.2	-

## 5. Conclusion

→ 210 + 1% filter