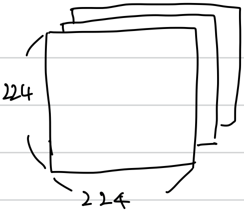
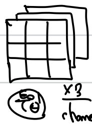
* VGGnet 16



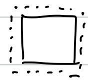
chanel = 3 →

After
COV 3x3 x 64개



⇒ { Stride 1
padding 1

⇒ padding 1



→ output 차원
224x224 유지
⇒ 3x3 conv + padding 1로 유지

① 3x224x224

Step 1 –
② Con3 x 64 통일 : 64 x 3x3 x 3 (chanel)
⇒ 64 x 224 x 224
⇒ ③ conv3 x 64 : 64 x 3x3 x 64 (chanel)
64 x 224 x 224
④ Maxpooling (2) → ex) ⊞ → ◻ , ▦ → ⊞
64 x 112 x 112 → kernal : 2x2 , stride 2 . padding : 0 ⇒

Step 2 –
① conv3 - 128 : 128 x 3x3 x 64 (chanel)
⇒ 128 x 112 x 112
② conV3 -128
③ maxpool (2) : 128 x 56x56

· · 256 . 256 . 256 (M) 512 . 512 . 512 . (M) 512 512 512 (M)

Last: 512 (chanel) x 7 by 7

· FC Layer

512x7x7

① Flatten
1x1 x 512x7x7 = 25088 ⇒ ② Layer 1x1x4096

(25088x4096 Weight) → "Limitation" VGGnet의 60% 비중

→ ③ FC4096 ⋮ ⇒ ④ FC1000 (1000개 분류)

⇓

Back propagation ⇐
① FC Loss ↓
② Kernel (Filter) 최적화
⇒ model 완성!

Conv 3 set { Conv Layer
Batch Normalization
ReLU } → Maxpooling → … → Classification

필요하다면 Adaptive Average Pooling (7,7) ✓

Classification –
chanel x dim (N by N) x 4096
ReLU
Dropout
4096 x 4096
ReLU
Dropout
4096 x class 분류 수