

GoogLeNet

0. Abstract, 1. Introduction

1. ILSVRC 2014

2. 차원 효율 (깊이, 폭 증가시)

→ Hebbian principle + multi-scale

→ 축소시 multiply-add 대신 더하기

3. Model: Inception

→ 22 layers

→ AlexNet 1/4 데드레이션 대비 ↓

2. Related Work

1. 기존

→ 여러 conv + fc 뒤에 max + layer 크기, 수 ↑ + dropout + max pool

2. Gabor filter: 2 layers 고정 필터 (multi-scale)

↳ 고정 필터 [3x3 만 모든 레이어 : multi-scale X]

[3x3, 5x5, 7x7 : multi-scale O (Gabor)]

⇒ 디신 필터로 학습해서 치환

중간 영상 ($H \times W$) 손실 우려는 있음

3. Network-in-Network: 1×1 conv

→ 차원 축소 (깊이 폭↑ 때, 병목↓)

↳ 네트워크 폭, 깊이 ↑

↓ layer 개수

"on layer 마다 채널 수 = 뉴런 수 = 필터 수"

3. Motivation and High Level Considerations

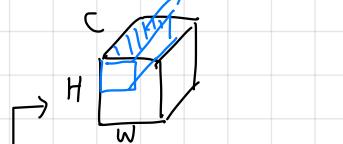
1. 네트워크 폭, 깊이 ↑ → 성능 ↑ → 과적합 (: 디버깅 ↑)

(준밀하게 규모 커으면 연산량 ↑ (+ 가중치 ≈ 0 : 더 놀라운...))

→ CONV 시에 C_1, \dots, C_{in} 에 대한 각 가능치가 있음

2. ⇒ 차원 효율 높임

↳ "회소 연결 (sparsity)" ~ (Hebbian)



→ CONV 경우 공간 ($H \times W$)은 이미 회소 (local) 연결 → channel 축은 fc보다 훨씬 적음

→ fc 경우 연결 자체를 fc → sparsity

↪ but, dense 행렬 공간이 우리 / (비교의 흐름은 수치 계산 비효율
ConvΣ 구현은 dense 연산)

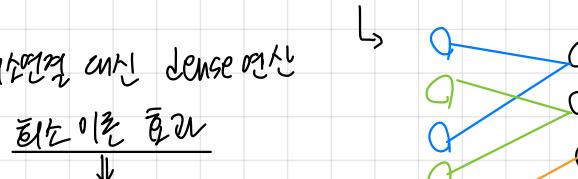
⇒ 흐름(이론)은 가능하되, 계산은 dense matrix로 처리
↪ "稠密행렬을 상대적으로 dense한 벡터로 Clustering 하기 가능"

4. Architectural Details

1. Atrous: layer-by-layer

→ 흔히나 유닛별 성능을 계산 후 높은 유닛끼리 Clustering → 다음 층의 각 유닛들은 조심스럽게 연결만 받음 ⇒ 흐름연결

↪ Inception: Atrous처럼 전파 흐름연결 대신 dense연산
(cohv)으로 해결하되, 흐름 이론 허용



Conver 단위 fc 연결 대신

연결 흐름 (1x1/3x3/5x5/pooling)

방법 018 → 이들을 Concat

(+우선 1x1로 차원축소로 018까지 ↓)

CONV 흐름연결 대신

차원 축소 연산 (018까지↓)

을 예로 사용

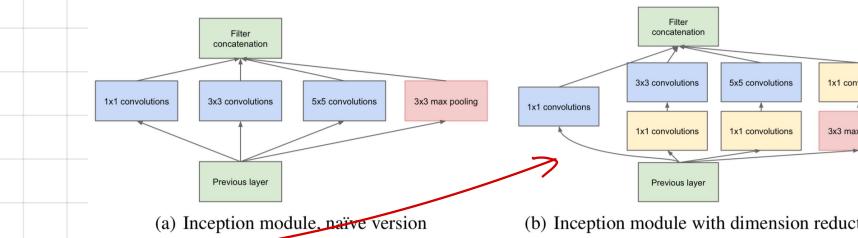


Figure 2: Inception module

↑ 문제, 01.. (다 합친/조심스럽게 누적→큰 키워드)
하루종일: 주소영역 충돌 (파일, 딜레이)
(연결 경로 충돌 문제)

2. $5 \times 5 \uparrow$, $1 \times 1 \rightarrow 018 \uparrow$

⇒ $3 \times 3, 5 \times 5$ 쪽에는 1×1 (dim reduction & Activation(RELU))
($H \times W \downarrow$ 하면서 maxpooling (3×3 , stride 2))

5. GoogLeNet

1. Inception 아키텍처

→ 표 1과 동일한 net + 1번의 width 넓힌 net

한번의 넓힘, 마지막 순서 차이

2. ReLU

입력 이미지: RGB, 224x224, 정규화(0~1정규화)

3. 27 layers (5번 Pooling, 7층까지 27000 22700)

4. fc 단계 avgpooling

| type | patch size/ stride | output size | depth | #1x1 | #3x3 reduce | #3x3 | #5x5 reduce | #5x5 | pool proj | params | ops |
|-------------------|-----------------------|----------------|-------|------|----------------|------|----------------|------|--------------|--------|-------|
| 1 convolution | 7x7/2 | 112x112x64 | 1 | | | | | | | 2.7K | 34M |
| 2 max pool | 3x3/2 | 56x56x64 | 0 | | | | | | | | |
| 3 convolution | 3x3/1 | 56x56x192 | 2 | 1 | 64 | 192 | | | | 112K | 360M |
| 4 max pool | 3x3/2 | 28x28x192 | 0 | | | | | | | | |
| 5 inception (3a) | 28x28x256 | 2 | 64 | 2 | 96 | 128 | 1 | 16 | 32 | 32 | 159K |
| 6 inception (3b) | 28x28x480 | 2 | 128 | 3 | 128 | 192 | 2 | 32 | 96 | 64 | 380K |
| 7 max pool | 3x3/2 | 14x14x480 | 0 | | | | | | | | |
| 8 inception (4a) | 14x14x512 | 2 | 192 | 4 | 96 | 208 | 3 | 16 | 48 | 64 | 364K |
| 9 inception (4b) | 14x14x512 | 2 | 160 | 5 | 112 | 224 | 4 | 24 | 64 | 64 | 437K |
| 10 inception (4c) | 14x14x512 | 2 | 128 | 6 | 128 | 256 | 5 | 24 | 64 | 64 | 463K |
| 11 inception (4d) | 14x14x528 | 2 | 112 | 7 | 144 | 288 | 6 | 32 | 64 | 64 | 580K |
| 12 inception (4e) | 14x14x832 | 2 | 256 | 8 | 160 | 320 | 7 | 32 | 128 | 128 | 840K |
| 13 max pool | 3x3/2 | 7x7x832 | 0 | | | | | | | | |
| 14 inception (5a) | 7x7x832 | 2 | 256 | 9 | 160 | 320 | 8 | 32 | 128 | 128 | 1072K |
| 15 inception (5b) | 7x7x1024 | 2 | 384 | 10 | 192 | 384 | 9 | 48 | 128 | 128 | 1388K |
| 16 avg pool | 7x7/1 | 1x1x1024 | 0 | | | | | | | | |
| 17 dropout (40%) | | 1x1x1024 | 0 | | | | | | | | |
| 18 linear | | 1x1x1000 | 1 | | | | | | | 1000K | 1M |
| 19 softmax | | 1x1x1000 | 0 | | | | | | | | |

Table 1: GoogLeNet incarnation of the Inception architecture

↑ 표 1 |X|
Pooling →
Projection (1x1)의 출력 개수

5. $\text{pool} \rightarrow \text{auxiliary classifier}$ (보조 분류기)

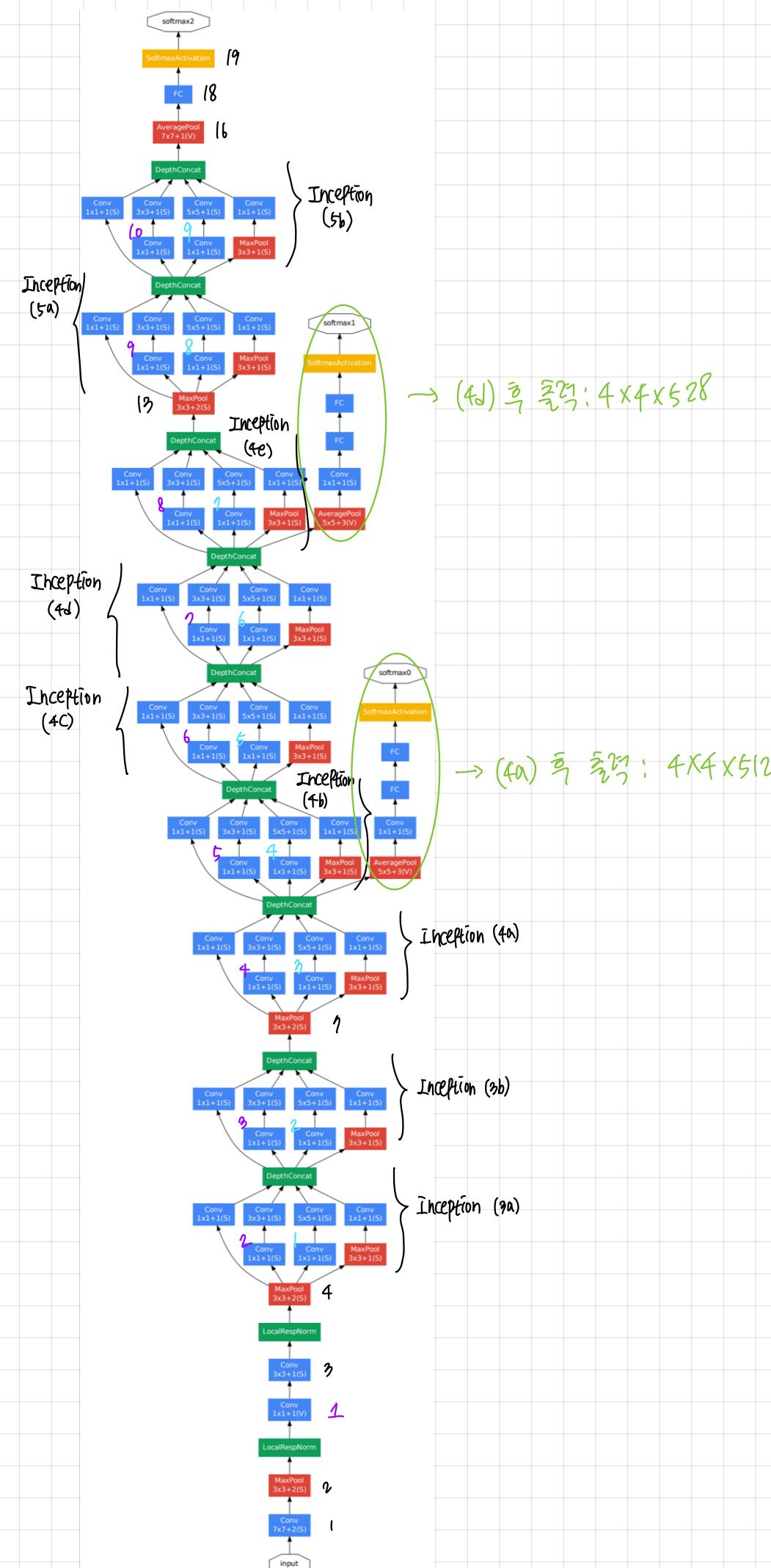
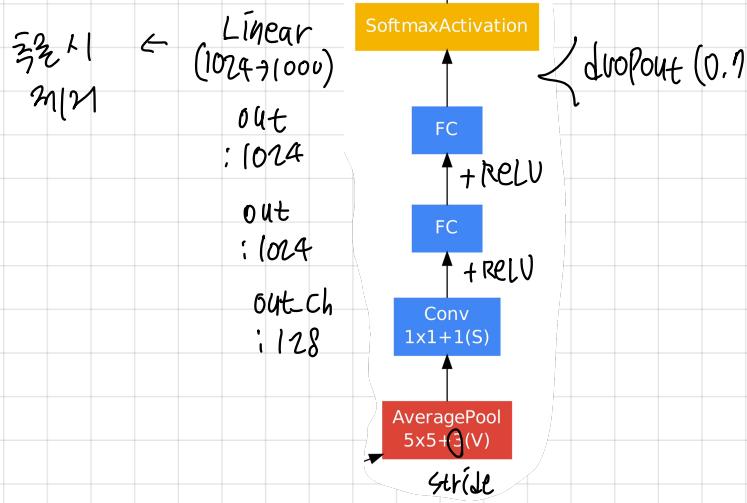
$\rightarrow (4a), (4d)$ 등이

\rightarrow 손실 계산 0, 2 차원으로 줄어듬

\rightarrow 흐름 시계는 아니

\rightarrow gradient vanishing 문제, gradient X, 미지수

↳ 보조 분류기 22



6. Training Methodology

1. momentum: 0.9
2. 비동기식 SGD
3. lr scheduler: 8 epochs 마다 4%씩 감소
4. 주로 시 터보레이트: 학습의 터보레이트를 정지

7. ILSVRC 2014 Classification Challenge setup and Results

1. Class 1000
(train 120만장, val 50만장, test 10만장)
2. Top1-error
5
3. Test 성능 향상
→ 77m 향상을
→ aggressive cropping
 - ↳ 4-scales: 256, 288, 320, 352
 - ↳ 77m (128가 77m); 왼, 중, 오 정사각형 → 256x256의 4모드, 288은 224x224 3모드
+
352: 상, 중, 하 → 224x224 rescale
 - ⇒ 4 scales x 3 local x 6 crops x 2 mirrors = "144 crops"
- Crops는 softmax layer avg 향상

4. 결과

| Team | Year | Place | Error (top-5) | Uses external data |
|-------------|------|-------|---------------|--------------------|
| SuperVision | 2012 | 1st | 16.4% | no |
| SuperVision | 2012 | 1st | 15.3% | Imagenet 22k |
| Clarifai | 2013 | 1st | 11.7% | no |
| Clarifai | 2013 | 1st | 11.2% | Imagenet 22k |
| MSRA | 2014 | 3rd | 7.35% | no |
| VGG | 2014 | 2nd | 7.32% | no |
| GoogLeNet | 2014 | 1st | 6.67% | no |

Table 2: Classification performance

| Number of model | Number of Crops | Cost | Top-5 error | compared to base |
|-----------------|-----------------|------|-------------|------------------|
| 1 | 1 | 1 | 10.07% | base |
| 1 | 10 | 10 | 9.15% | -0.92% |
| 1 | 144 | 144 | 7.89% | -2.18% |
| 7 | 1 | 7 | 8.09% | -1.98% |
| 7 | 10 | 70 | 7.62% | -2.45% |
| 7 | 144 | 1008 | 6.67% | -3.45% |

Table 3: GoogLeNet classification performance break down

8. ILSVRC 2014 Detection Challenge Setup and Results

(논문 참고)

9. Conclusions

1. 최적 회소 구조를 딸기 빙당 날록으로 조사
2. 각각의 경로 : (다음에 확인)