VGGNet

작성일시	@December 27, 2021 2:47 PM
강의 번호	2015년
유형	리뷰
자료	https://arxiv.org/abs/1409.1556
복습	☑

1. Introduction

- VGGNet 논문에서는 Convent 구조에서 깊이와 관련된 부분에 집중하였다. 그렇기 때문에 논문에서는 구조의 다른 파라미터들은 고정시켜 놓고 convolution 층을 증가시켜 가는 것을 확인할 수 있었다
- convolution 층을 증가시키는 방법은 모든 층에 3*3의 작은 convolution 필터를 적용하였기때문에 가능함

2. ConvNet Configurations

1. arcitecture
    - ConvNets의 input값은 224*224 RGB 로 고정되어있는데 논문에서 사용된 사전처리는 각 픽셀에서 훈련 세트에서 계산된 평균 RGB 값을 빼는 것
    - 3*3 receptive field를 사용함(stride는 1 고정)
        - 위/아래/왼쪽/오른쪽/중앙을 인식하기 위한 가장 작은 크기
        - conv 층의 spatial padding은 이미지 외곽부분을 extended(?)하게 넓은 범위로 반영되게함으로 image vanishing을 방지함
    - 논문에서 사용된 FC layers는 4096개의 채널이 있는 레이어 2개 1000개의 채널을 포함한 레이어 1개

- 해당 논문에서는 Local Response Normalization(LRN)을 포함하지 않았음
  - 오히려 성능을 개선하지않고 메모리 소비량을 증가시키기 때문

    ⇒ Batch_Normalization을 적용하면 어떨까? 라는 궁금증이 생김 다음에 실험해보도록 하자

2. configurations

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
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 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
maxpool					
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 conv3-256 **conv3-256**
maxpool					
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 conv3-512 **conv3-512**
maxpool					
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 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 1

- 네트워크 A의 11개 가중치 계층 ~ 네트워크 E의 19개 가중치 계층까지로 구성됨.
- channel의 수는 64로 시작하여 각각의 maxpool 층을 지날때마다 2배씩 증가하여 512개의 channel 수 까지 증가하게 됨
- Conv1층과 Conv3층 사용 차이와 Layer층이 깊어짐에 따라 paramerter 수도 많아진다.

Table 2: **Number of parameters** (in millions).

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

3. discussion
    - 상대적으로 작은 3*3 receptive field를 사용하였음
        - 2개의 3*3 conv층을 사용하면 5*5의 receptive field의 성능
        - 3개의 3*3 conv층을 사용하면 7*7의 receptive field의 성능
    - 왜 7*7 층보다 3개의 3*3층을 사용하는걸까?
        - 단일 수정 계층 대신 3개의 비선형 수정 계층을 통합하여 의사 결정 기능을 더욱 차별화
            - 하나의 두꺼운 층을 선택하는것보다 얇은 3개의 층을 선택하면 더욱 다양한 의사결정 가능
        - 파라미터 수를 줄임
            - 1*1 convolution의 통합은 convolution layer의 수용 영역에 영향을 주지 않고 의사결정 기능의 비선형성을 증가시키는 방법
            - 1*1 convolution은 동일한 차원의 공간에 대한 선형 투영이지만, 수정 함수에 의해 추가 비선형성이 도입됨

3. Classification Framework

1. Training
    - batch_size = 256, momentum = 0.9 dropout은 첫 두 FC-Layer에서 일어난다
    - learning_rate = 10^-2두고 learning_rate가 3번 감소하고 370K(74epochs)번의 반복이후에 학습이 멈춤

      ⇒VGGNet의 많은 파라미터와 깊은 층에 비교하여 적은 epoch을 요구

      ⇒깊은 층과 작은 컨볼루션 필터사이즈에 의해 시행되는 암시된 정규화와 몇몇층에서 시행되는 사전 초기화 때문

      > 학습 이미지 크기

- 학습 데이터를 다양한 크기로 변환하고 그 중 일부분을 샘플링해 사용함으로써 몇 가지 효과를 얻을 수 있음
  - 한정적인 데이터의 수를 늘릴수있음(Data augmentation)
  - 하나의 오브젝트에 대한 다양한 측면을 학습 시 반영시킬 수 있다.

    ⇒두 가지 모두 Overfitting을 방지하는 데 도움이 됨

2. Testing

- Input image는 사전에 정의된 smallest image side로 isotropically rescale 되며 test scale Q로 표시됨
  - test scale Q가 training scale S와 같을 필요없음
- Fully Connected layer가 conv layer로 변환됨
  - resulting Fully Convolutional network가 전체 image에 적용
  - 결과
    - class의 개수와 동일한 개수의 channel을 갖는 class score map
    - Input image size에 따라 변하는 spatial resolution = input image size의 제약이 사라짐
    - 하나의 image를 다양한 scale로 ㄷ상용한 결과를 조합해 image classification accuracy 개선가능
  - implementaion details
    - 4-GPU 시스템에서 단일 GPU 시스템보다 속도가 3.75배 향상

4. Classification Experiments

4.1 Single scale evaluation

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
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 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
maxpool					
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 conv3-256 **conv3-256**
maxpool					
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 conv3-512 **conv3-512**
maxpool					
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 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 2: **Number of parameters** (in millions).

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

Table 1 & 2

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	256	256	28.1	9.4
	384	384	28.1	9.3
	[256;512]	384	27.3	8.8
D	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	**25.5**	**8.0**

Table3

- 테스트 이미지의 크기는 고정된 S(=Q)와 jitter를 사용한 S가 있고 그 성능평가의 결과는 table 3

$$Q = 0.5(S_{min} + S_{max}) \text{ for jittered } S \in [S_{min}, S_{max}].$$

- LRN을 사용하는 것이 Normalization 계층이 없는 모델 A에서 개선되지않는다.
  - 그래서 B~E에서는 Normalization 채택X
- ConvNet의 깊이가 증가함에 따라 분류 오류가 감소함
  - 같은 깊이에도 불구하고 3개의 1*1 Conv층을 가진 C가 네트워크 전체적으로 3*3 Conv 층을 사용하는 B보다 성능이 안좋았습니다. 이는 추가 비선형성이 도움이 되지만(C가 B보다 낫지만) 사소한 receptive 필드를 가진 컨볼루션 필터를 사용하여 공간 맥락을 포착하는 것도 중요하다는 것을 나타낸다(D가 C보다 낫다).
- 논문에서 설계한 모델의 구조는 깊이가 19층에 도달했을때 수렴함

4.2Multi scale evaluation

Table 4: **ConvNet performance at multiple test scales.**

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
B	256	224,256,288	28.2	9.6
C	256	224,256,288	27.7	9.2
	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
	[256; 512]	256,384,512	**24.8**	**7.5**
E	256	224,256,288	26.9	8.7
	384	352,384,416	26.7	8.6
	[256; 512]	256,384,512	**24.8**	**7.5**

Table 4

- 테스트시 scale jittering을 사용한 것이 고정된 test scale을 가질 경우 보다 더 좋은 성능을 발휘

4.3 Multi-crop evaluation

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	**24.4**	**7.2**
E	dense	24.8	7.5
	multi-crop	24.6	7.4
	multi-crop & dense	**24.4**	**7.1**

Table 5

- multi-crop 을 사용하는 것이 dense 평가를 사용하는 것보다 약간의 좋은 성능을 보이지만 두 가지 방식을 조합하였을때 상호보완적이라는 것을 알 수 있음.

4.4 CONVNet fusion

Table 6: **Multiple ConvNet fusion results.**

Combined ConvNet models	Error		
	top-1 val	top-5 val	top-5 test
ILSVRC submission			
(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
post-submission			
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), dense eval.	24.0	7.1	7.0
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), multi-crop	23.9	7.2	-
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), multi-crop & dense eval.	**23.7**	**6.8**	**6.8**

Table 6

- Soft-max class posterior 을 평균화하여 여러 모델의 output을 조합하고 이를 통해 모델의 보완성으로 성능을 향상 시킴

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.)	**23.7**	**6.8**	**6.8**
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)	-	**6.7**	
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	-

Table 7

- LVRC-2014 챌린지 분류 과제(Russakovsky et al., 2014)에서 "VGG" 팀은 7개의 모델을 앙상블하여 7.3%의 test error로 2위를 차지하였고, 제출 후에는 두개의 모델을 앙상블하여 error를 6.8%까지 줄였다고 함.

5. Conclusion

- 논문에서 실험 결과는 모델의 깊이가 깊어질수록 성능이 더 좋아지는 것을 확인할 수 있었고 모델구조를 모르고 keras 에서 import 만해서 VGG16, VGG19 를 사용하였을때 왜 19까지가 최대인지 궁금했던 부분을 확인할 수 있었음
- 첫 논문리뷰이지만 아직 모르는 부분이 많아 여러 리뷰 글을 구글링하며 이해하고 리뷰해보니 다음 모델 분석부터는 어떤 방법으로 시작할지 그리고 마쳐야할지 감이 오는거 같다 다음 리뷰도 시작해보자