SqueezeNet Review

Reference

- SQUEEZENET: ALEXNET-LEVEL ACCURACY WITH 50X FEWER PARAMETERS AND <0.5MB MODEL SIZE

Introduction

CNN 연구의 흐름은 비교적 간단한 아키텍쳐를 유지하며 높은 정확도를 내는 모델을 찾는 연구가 주를 이루고 있다. 아키텍쳐가 간단하다는 것은 파라미터가 적은 모델을 의미하며 이는 곧 학습할 weight가 적다는 뜻이다. 이러한 모델은 아래와 같은 장점을 가진다고 한다.

- More efficient distributed training.
- Less overhead when exporting new models to clients.
- Feasible FPGA and embedded deployment.

이러한 장점을 살리기 위해 SqueezeNet 연구가 진행되었다. SqueezeNet은 요약하면 AlexNet과 비슷한 성능을 내지만 파라미터 수는 50배 적은 모델이다. 따라서 정확도는 유지하며 모델의 아키텍쳐는 매우 간단하다고 할 수 있다.

Architectural Design Strategies

SqueezeNet은 높은 정확도는 유지하며 동시에 파라미터의 수를 줄이기 위해 아래 전략을 사용한다.

- Strategy 1

  3×3 filters을 1×1 filters로 바꾼다. 1×1 conv filter는 차원 축소의 효과, FC의 효과 등의 이점을 얻을 수 있음을 여기에서 살펴보았다.

- Strategy 2

  3×3 filters로 가는 input channels의 수를 줄인다. 3×3 filters로 구성되어 있는 conv layer을 생각해보자. 이 layer의 파라미터의 총 갯수는 # of input channels * # of filters * (3×3)이다. 따라서 파라미터의 수를 줄이기 위해서는 3×3 filters을 줄이는 것 뿐만 아니라 input channels의 수를 줄이는 것도 중요하다.

- Strategy 3

  Downsample을 늦게 해서 conv layers가 large activation maps을 가지게 한다. 대부분 downsampling은 CNN 아키텍쳐에서 1보다 큰 stride을 지정하거나 pooling layers에서 일어난다. 초기 layers가 큰 strides을 가진다면 그 후에 나오는 대부분의 layers는 작은 actiation maps을 가지게 된다. 반대로 대부분의 layers의 stride가 1이고 네트워크의 마지막 부분에 1보다 큰 strides가 집중되어 있다면 많은 layer의 activation maps이 클 것이다. activation maps이 크다면 분류 정확도를 높인다고 알려져 있으므로, 이를 squeezenet에 이용한다.

The Fire Module

fire module은 *squeeze* conv layer와 *expand* conv layer로 구성된다.

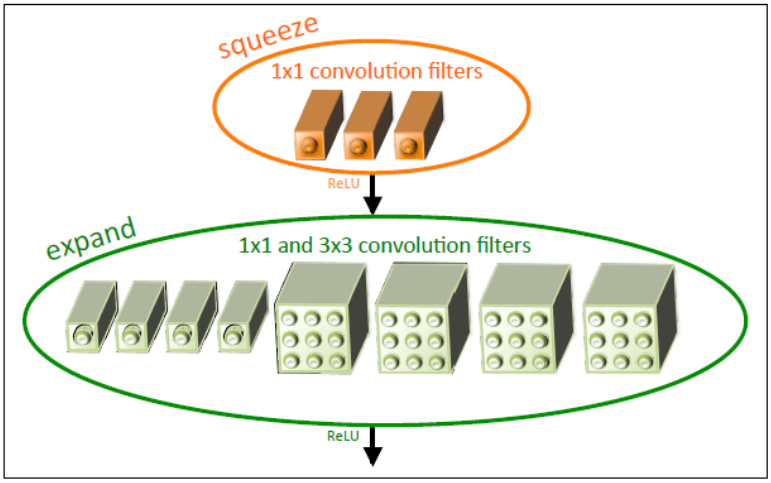


Figure 1: Microarchitectural view: Organization of convolution filters in the **Fire module**. In this example, $s_{1x1} = 3$, $e_{1x1} = 4$, and $e_{3x3} = 4$. We illustrate the convolution filters but not the activations.

squeeze conv layer은 1×1 conv filters만 있고 *expand* conv layer은 $1 \times 1, 3 \times 3$ conv filters로 구성된다. *squeeze* conv layer은 strategy 1을 실현했다고 보면 된다.

fire module에서는 조정해야 할 세 개의 hyperparameters가 있다. $s_{1x1}, e_{1x1}, e_{3x3}$ 인데, 각각 *squeeze* layer에서 1×1 conv filters의 수, *expand* layer에서 $1 \times 1, 3 \times 3$ conv filters의 수이다. 이들을 정할 때 $s_{1x1} < e_{1x1} + e_{3x3}$를 맞추며 조정한다고 한다. 이는 곧, strategy 2, input channels의 수를 제한하기 위함이다.

The SqueezeNet Architecture

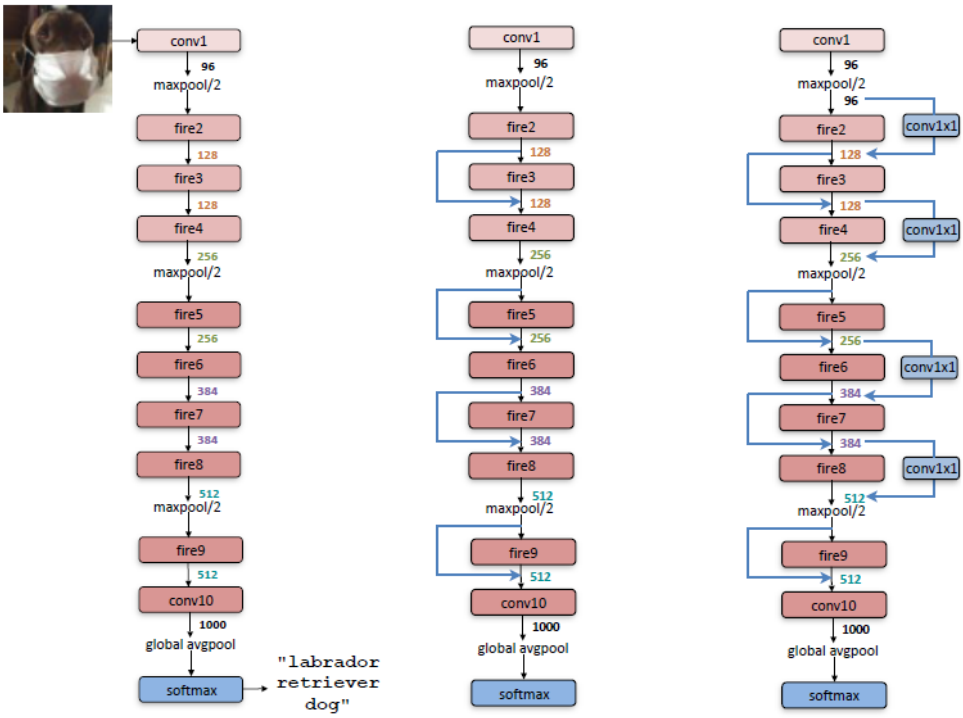


Figure 2: Macroarchitectural view of our SqueezeNet architecture. Left: SqueezeNet (Section 3.3); Middle: SqueezeNet with simple bypass (Section 6); Right: SqueezeNet with complex bypass (Section 6).

위 그림은 squeezenet 아키텍쳐이다. 맨 처음에 일반적인 conv layer와 maxpool이 있다. 그 후 8개의 fire module이 나오고 마지막으로 conv layer와 global avgpool이 있다. conv1, fire4, fire8, conv10후에 stride 2 maxpooling을 하는데, 이는 strategy 3을 실현하기 위함이다.

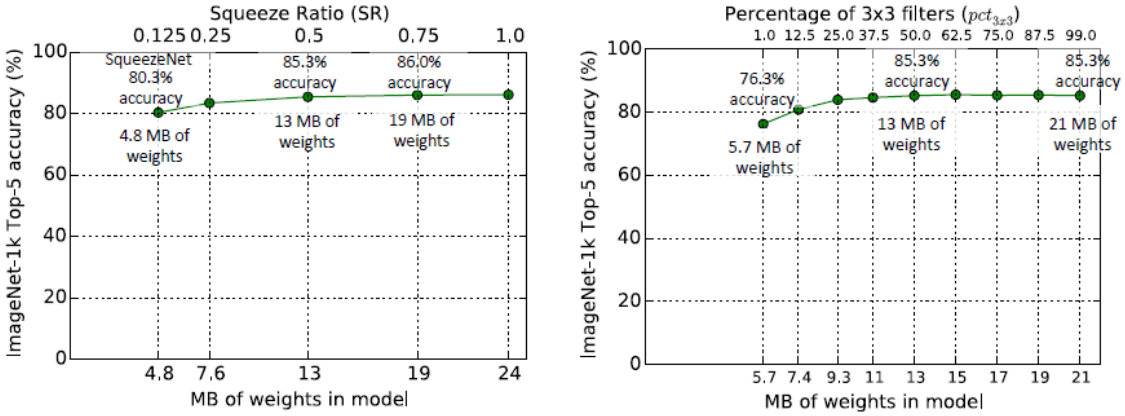
CNN MicroArchitecture Design Space Exploration

SquezeNet의 구조를 module layer 단위에서 살펴보자. squeezenet의 module layer는 위에서 언급한 세 개의 hyperparameters, $s_{1x1}, e_{1x1}, e_{3x3}$에 의해서 결정된다. 이 세 모수는 각 fire module마다 정의되므로 총 24개의 hyperparameters가 있다. 이들의 변화에 따른 CNN 모델의 정확도를 살펴보기 위해, 다음의 *metaparameters*을 정의하여 fire modules의 dimensions을 조정해보자.

$base_e$는 첫 fire module의 *expand* filters의 갯수이다. 매 *freq* fire module 후에, *expand* filters의 수를 $incr_e$만큼 늘린다. 즉, i번째 fire module의 *expand* filters의 수는 $e_i = base_e + (incr_e \times [\frac{i}{freq}])$. *expand* layer에서 $1 \times 1, 3 \times 3$ filters가 있으므로 $e_i = e_{i,1x1} + e_{i,3x3}$로 정의하고 모든 fire modules에서 공유되는 pct_{3x3}을 이용하여 $e_{i,3x3} = e_i \times pct_{3x3}, e_{i,1x1} = e_i \times (1 - pct_{3x3})$로 정의한다. 마지막으로 *squeeze* lyer의 filter 수를 *squeeze ratio (SR)*을 이용하여 정의한다.

$$s_{i,1x1} = SR \times e_i = SR \times (e_{i,1x1} + e_{i,3x3})$$

이제 SR 값이 변함에 따라 모델의 크기와 정확도가 어떻게 변화하는지 알아보자. starting point로 위 그림 2의 SqueezeNet 모델을 사용하였다. 이 모델의 metaparameters는
$base_e = 128, incr_e = 128, pct_{3x3} = 0.5, freq = 2$이다.



(a) Exploring the impact of the squeeze ratio (SR) on model size and accuracy.

(b) Exploring the impact of the ratio of 3x3 filters in expand layers (pct_{3x3}) on model size and accuracy.

Figure 3: Microarchitectural design space exploration.

그림 3-(a)로부터 SR이 0.125보다 커지기 시작하면 정확도를 0.86까지 올릴 수 있음을 알 수 있다.

또한 3×3 filters를 1×1 filters로 바꿔 파라미터 수를 줄이는 것을 제안했다. 여기서는 이에 대한 근거를 확보하기 위해 두 filters의 비율에 따라 변화하는 모델의 크기와 정확도를 관찰한다.
$base_e = incr_e = 128, freq = 2, SR = 0.5$로 고정하고 pct_{3x3}을 0.01에서 0.99까지 변화시켰다. 즉, 각 fire module의 *expand* layer을 대부분 1×1 conv filters로 구성하게 하거나, 또는 3×3 conv filters로 구성하게 조정했다. 그림 3-(b)를 보면, pct_{3x3}을 증가시키면 어느 정도까지는 정확도가 증가하지만 그 이후로는 모델의 크기만 증가하고 정확도는 정체되는 모습을 보인다. 즉, 3×3 conv filters의 갯수를 적절히 조절한다면 모델의 크기는 제한하면서 정확도를 최대로 끌어 올릴 수 있다는 뜻이다.

CNN MacroArchitecture Design Space Exploration

fire module의 layer에 대해서 살펴보았으니 이제 전체적인 아키텍쳐를 살펴보자. 그림 2를 보면 총 세 개의 아키텍쳐가 나온다.

- Vanilla SqueezeNet
- SqueezeNet with simple bypass connections between some Fire modules
- SqueezeNet with complex bypass connections between the remaining Fire modules

simple bypss 아키텍쳐는 ResNet의 residual 학습 아이디어를 빌려왔다. 3, 5, 7, 9번째 fire module에 bypass connection을 추가한다. 예를 들어, fire module4의 input으로 fire 2, fire 3의 output을 준다.

하지만 그림 2에서 볼 수 있듯이, intput channels와 output channels의 수가 상이한 fire가 있기 때문에, 모든 fire에서 residual 학습이 일어나지 않는다. 이러한 이유로 1×1 con filters로 차원을 맞춰주고 residual 학습을 하는 complex bypass가 제안되었다.

이러한 residual 학습은 또 다른 이점이 있다. SqueezeNet에서 SR이 0.125이고 이는 곧 squeeze layer 아 expand layer보다 8배 더 적은 output channels을 가짐을 의미한다. 이러한 심각한 차원 축소 때문에 한정된 정보가 squeeze layers로 갈 수 있다. 하지만 residual 학습을 통해 더 많은 정보를 squeeze layer에 보낼 수 있을 것이다.

Table 3: SqueezeNet accuracy and model size using different macroarchitecture configurations

Architecture	Top-1 Accuracy	Top-5 Accuracy	Model Size
Vanilla SqueezeNet	57.5%	80.3%	4.8MB
SqueezeNet + Simple Bypass	**60.4%**	**82.5%**	4.8MB
SqueezeNet + Complex Bypass	58.8%	82.0%	7.7MB

흥미롭게도, complex bypass보다 simple bypass가 성능이 더 좋다고 한다.