20231128

14.0 합성곱 신경망을 사용한 컴퓨터 비전

-대뇌의 시각피질 연구에서 출발, 1980년 이미지인식 분야에서 사용
-일부 복잡한 이미지 처리문제에서 사람을 능가하는 성능 달성.
-활용 예제: 이미지검색 서비스, 자율주행 자동차, 영상 자동분류 시스템, 음성인식, 자연어 처리

*주요 내용
-CNN의 구성요소
-TF와 케라스를 이용한 CNN의 구현
-가장 뛰어난 성능의 CNN 구조 살펴보기
-활용예시: 객체탐지,의미분할

14.1 시각 피질 구조

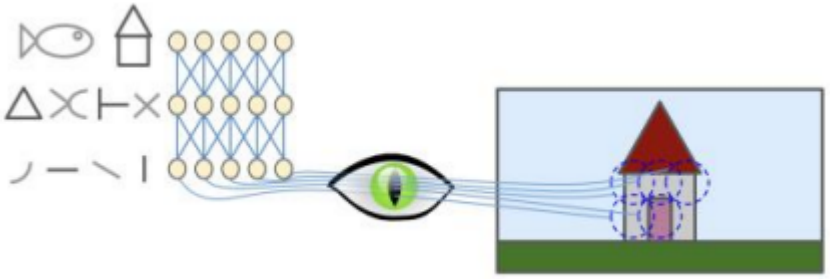
▪ 합성곱 신경망(CNN)은 대뇌 시각 피질 연구에서 시작
▪ 데이비드 허블과 토르스텐 비셀, 시각 피질의 구조에 대한 연구(1958)

*고양이를 이용한 연구

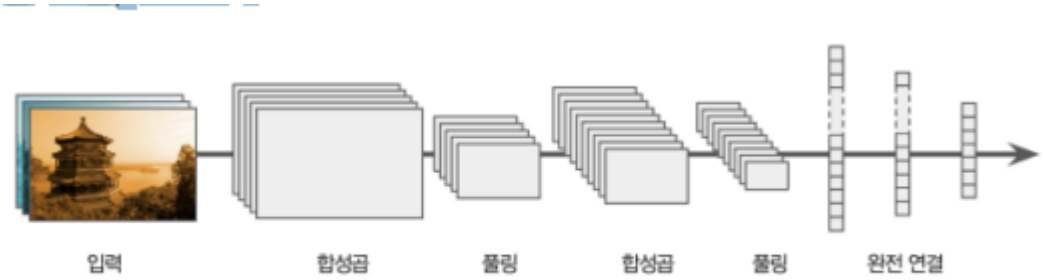
시각 피질 속 많은 뉴런 : 작은 국부수용장(local receptive)가짐 발견

→ 뉴런, 시야의 일부 범위 안에 있는 시각 자극에만 반응
ex)A뉴런 수평선의 이미지에만 반응,B 뉴런은 다른 각도의 선분에 반응, C뉴런은 (큰 수용장을 가짐) 저수준 패턴 조합된 복잡한 패턴에 반응
→idea 도출 : 고수준 뉴런 =이웃한 저수준 뉴런의 출력에 기반

*국부 수용장 모델을 모방



합성곱신경망(CNN)으로 발전
→합성곱 층(convolution layer) ,풀링 층(pooling layer)
→1998년 얀 르쿤, LeNet-5

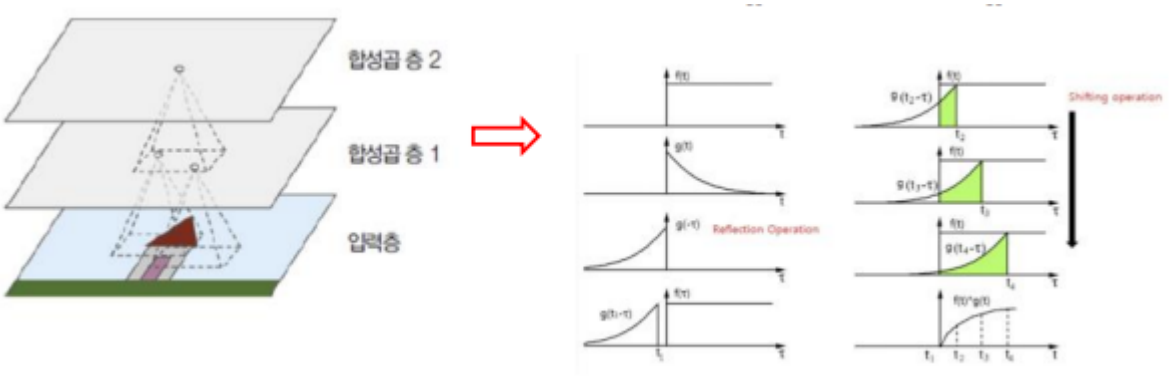


14.2 합성곱 층

*CNN의 가장 중요한 구성 요소는 합성곱 층
-첫 번째 합성곱 층의 뉴런: 입력 이미지의 모든 픽셀에 연결X, 합성곱 층 뉴런의 수용장속 픽셀에만 연결
-두 번째 합성곱 층에 있는 각 뉴런:첫 번째 층의 작은 사각 영역 안에 위치한 뉴런에 연결

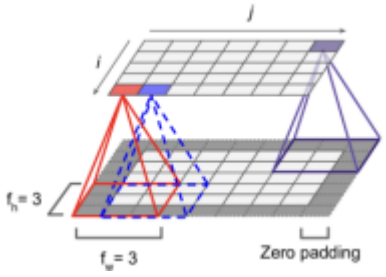
-스트라이드 : 한 수용장과 다음 수용장 사이의 간격
-사각 형태의 국부수용장을 모방한 CNN 층

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau = \int_{-\infty}^{\infty} f(t-\tau)g(\tau)d\tau$$



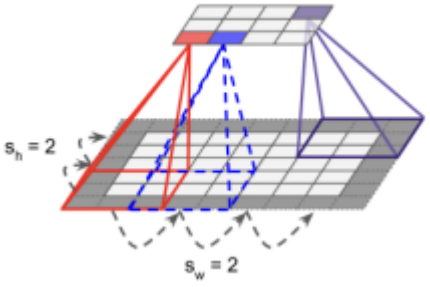
14.2합성곱 층(1)

*층과 제로패딩 사이의 연결 → 크기 안줄음



*보폭(스트라이드)2를 사용한 차원축소 지원 CNN층

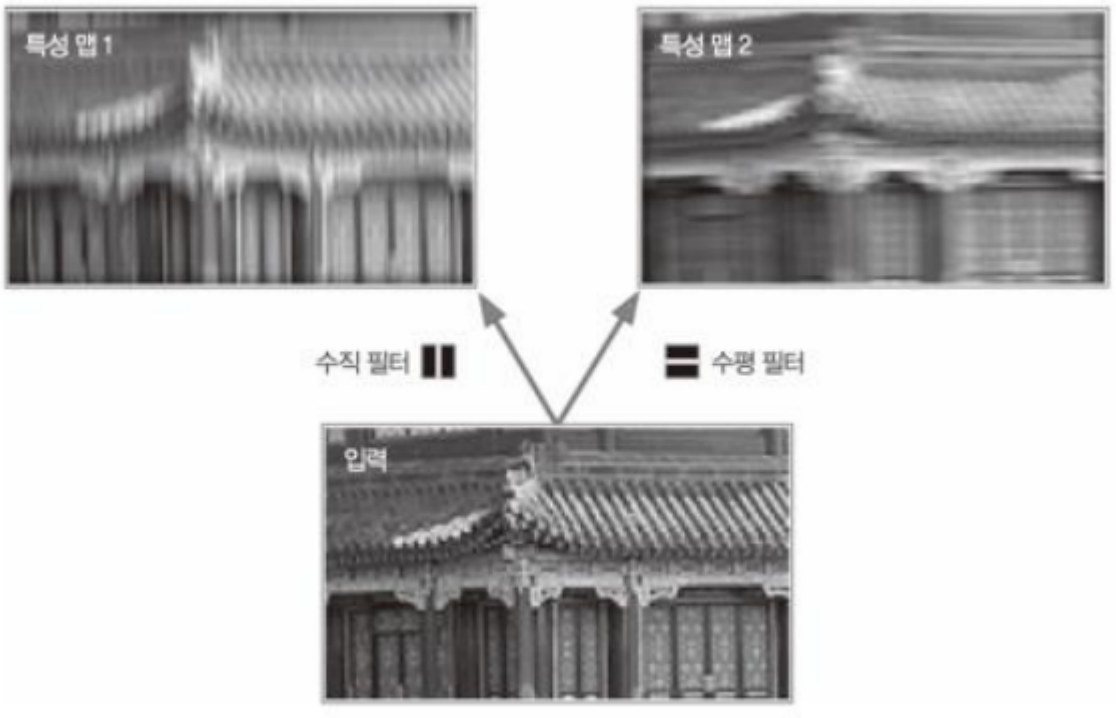
→스트라이드: 수용장과 다음 수용장 사이의 간격



14.2.1 필터 (합성곱 커널)

*입력뉴런에 사용될 가중치 역할 수행
필터의 모양과 크기= 국부수용장의 모양과 크기
→필터는 넘파이 어레이로 지정됨
→ 다양한 필터 사용

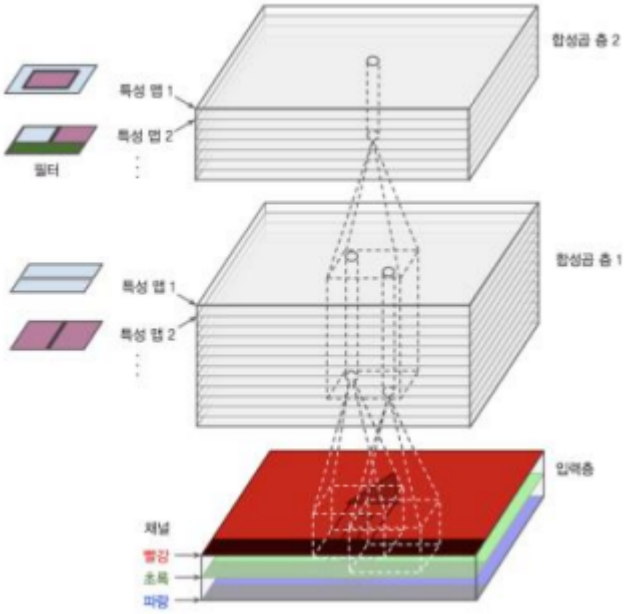
→필터 수는 하이퍼파라미터로 지정.
ex)



14.2.2 여러 가지 특성 맵 쌓기

*특성맵(Feature Map)
-특성맵: 필터 각각을 사용하여 생성된 출력값, 각 특성맵의 픽셀= 하나의 뉴런
-수십, 수백 개의 필터를 사용
-필터에 포함된 모든 뉴런은 동일한 가중치,편향 사용
-필터마다 사용되는 가중치와 편향은 다름.

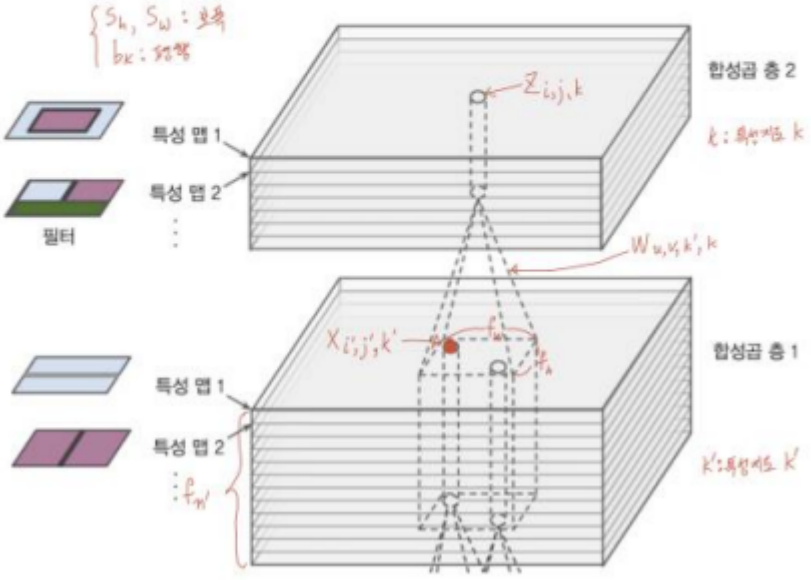
*컬러채널(Color Channel)
-이미지를 대상으로 하는 합성곱 층은 3차원으로 표현 가능
-입력 이미지가 컬러인 경우 R, G, B 세 개의 채널,
흑백인 경우 하나의 채널 사용



14.2.2 여러 가지 특성 맵 쌓기 (1)

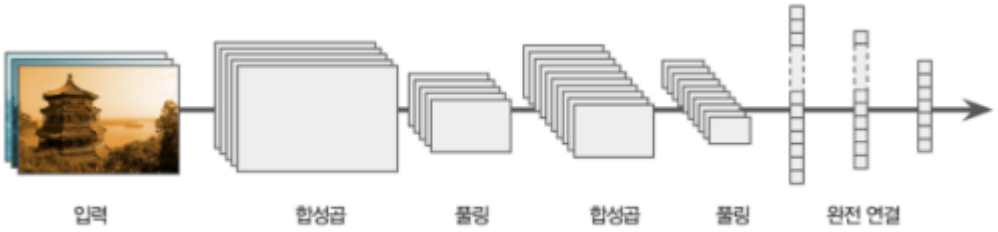
각 뉴런의 출력값: 입력에 대한 가중치의 합에 편향을 더한 값

$$z_{i,j,k} = b_k + \sum_{u=0}^{f_h-1}\sum_{v=0}^{f_w-1}\sum_{k'=0}^{f_{n'}-1} x_{i',j',k'} \times w_{u,v,k',k} \quad \text{여기서} \begin{cases} i' = i \times s_h + u \\ j' = j \times s_w + v \end{cases}$$



14.4 CNN 구조

*전형적인 CNN 구조
-네트워크를 통과하여 진행할수록 이미지는 점점 작아지지만, 합성곱 층 때문에 일반적으로 점점 더 깊어짐 (즉, 더 많은 특성 맵을 가지게 됨)
- 이미지 인식 에서 완전 연결 층의 심층 신경망을 사용X 이유 = 파라미터가 너무 많아짐, CNN: 층 부분적으로 연결 &가중치 공유
-합성곱 층에 사용하는 커널 크기
:작은 커널= 파라미터와 계산량이 적고 더 나은 성능



14.4 CNN 구조 (1)

*케라스 활용: 패션 MNIST
-패션 MNIST 데이터셋 문제를 해결하기 위한 간단한 CNN
-아래 합성곱 모델이 92% 정도의 정확도 성능 발휘

-10장의 밀집 네트워크보다 좋은 성능임.

```
model = keras.models.Sequential([
    DefaultConv2D(filters=64, kernel_size=7, input_shape=[28, 28, 1]),
    keras.layers.MaxPooling2D(pool_size=2),
    DefaultConv2D(filters=128),
    DefaultConv2D(filters=128),
    keras.layers.MaxPooling2D(pool_size=2),
    DefaultConv2D(filters=256),
    DefaultConv2D(filters=256),
    keras.layers.MaxPooling2D(pool_size=2),
    keras.layers.Flatten(),
    keras.layers.Dense(units=128, activation='relu'),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(units=64, activation='relu'),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(units=10, activation='softmax'),
])
```

14.2.3 텐서플로 구현

```
import numpy as np
from sklearn.datasets import load_sample_image

# 샘플 이미지를 로드합니다.
china = load_sample_image("china.jpg") / 255
flower = load_sample_image("flower.jpg") / 255
images = np.array([china, flower])
batch_size, height, width, channels = images.shape

# 2개의 필터를 만듭니다.
filters = np.zeros(shape=(7, 7, channels, 2), dtype=np.float32)
filters[:, 3, :, 0] = 1  # 수직선
filters[3, :, :, 1] = 1  # 수평선

outputs = tf.nn.conv2d(images, filters, strides=1, padding="SAME")

plt.imshow(outputs[0, :, :, 1], cmap="gray") # 첫 번째 이미지의 두 번째 특성맵을 그립니다.
plt.axis("off") # 책에는 없습니다.
plt.show()
```

14.3 풀링 층

계산량과 메모리 사용량을 줄이면서 과대적합의 위험도를 줄여주는 용도로 사용됨.

- 풀링 층 뉴런은 가중치가 없음.
- 보폭(stride)를 사용하여 차원을 축소시키는 기능 수행
- 최대 풀링층(max pooling layer)
- 2x2 크기의 풀링 커널(pooling kernel)
- 네 개의 셀에 속한 값들 중에서 가장 큰 값만 상위 층으로 전달됨.
- 보폭(stride): 2
- 하위 입력층에서 두 칸씩 건너 뛰며 풀링커널 적용. 상위층의 뉴런 수가 1/4 로 줄어듦.
- 패딩 없음(padding="valid")
- 보폭에 따라 일부 행과 열이 무시될 수 있음.

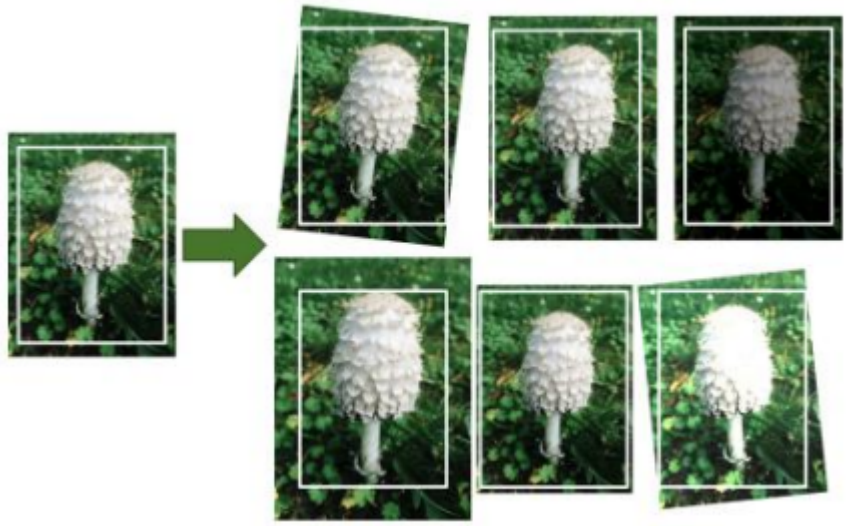
14.4.1 LeNet-5

*LeNet-5
-가장 널리 알려진 CNN 구조
-1998년 얀르쿤이 만듬. 손글씨 숫자 인식(MNIST)에 사용
-구조

층	종류	특성 맵	크기	커널 크기	스트라이드	활성화 함수
출력	완전 연결	-	10	-	-	RBF
F6	완전 연결	-	84	-	-	tanh
C5	합성곱	120	1×1	5×5	1	tanh
S4	평균 풀링	16	5×5	2×2	2	tanh
C3	합성곱	16	10×10	5×5	1	tanh
S2	평균 풀링	6	14×14	2×2	2	tanh
C1	합성곱	6	28×28	5×5	1	tanh
입력	입력	1	32×32	-	-	-

14.4.2 AlexNet

*AlexNet
-2012년 이미지넷 대회에서 우승
-구조: LeNet-5와 비슷하지만 크고 깊음, 처음으로 합성곱 층 위에 풀링 층을 쌓지 않고 바로 합성곱 층끼리 쌓음.
-특징
-과대 적합을 줄이기 위해 두 가지 규제 기법: 드롭 아웃을 50% 비율로 적용. (F9와 F10의 출력),데이터 증식 수행
•-데이터증식: 훈련샘플을 인공적으로 생성 기법(수평 뒤집기, 간격 이동, 조명 변경 등)



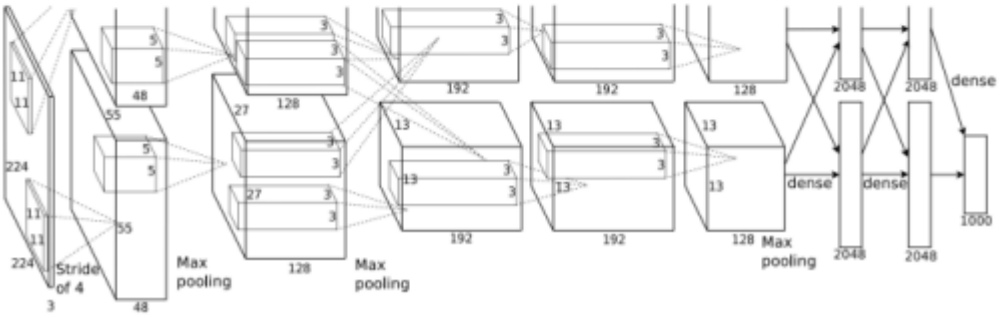
-정규화: LRN(local response normalization)

• 뉴런의 출력값을 보다 경쟁적으로 만드는 정규화 기법적인 정규화 단계 사용

• 각각의 특성맵을 보다 특별하게 만들어서 보다 다양한 특성을 탐색할 수 있도록 도와줌.

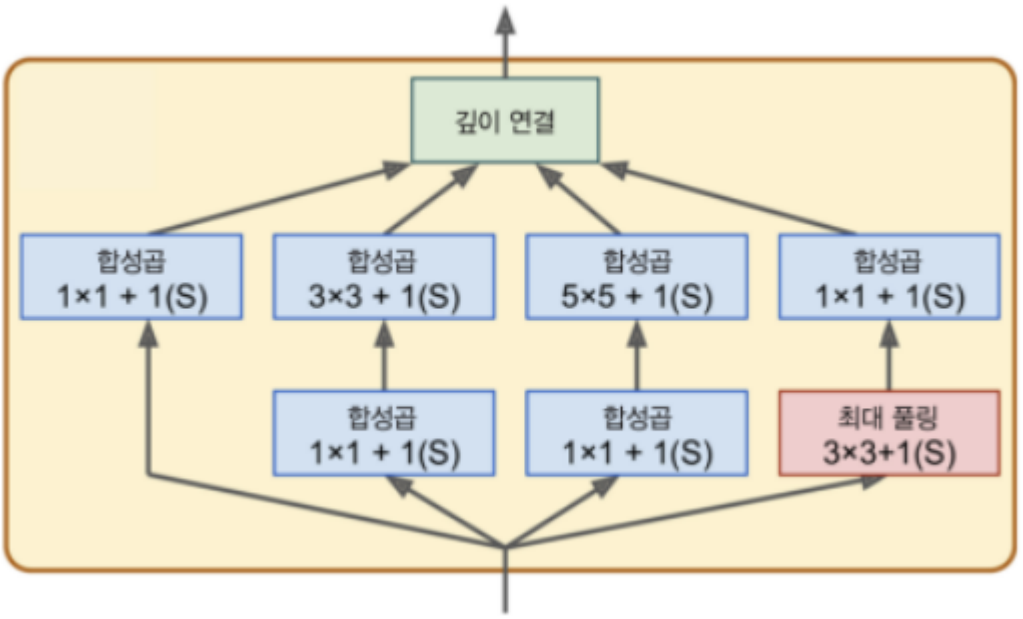
14.4.2 AlexNet (2)

층	종류	특성 맵	크기	커널 크기	스트라이드	패딩	활성화 함수
출력	완전 연결	-	1,000	-	-	-	Softmax
F10	완전 연결	-	4,096	-	-	-	ReLU
F9	완전 연결	-	4,096	-	-	-	ReLU
F8	최대 풀링	256	6×6	3×3	2	valid	-
C7	합성곱	256	13×13	3×3	1	same	ReLU
C6	합성곱	384	13×13	3×3	1	same	ReLU
C5	합성곱	384	13×13	3×3	1	same	ReLU
S4	최대 풀링	256	13×13	3×3	2	valid	-
C3	합성곱	256	27×27	5×5	1	same	ReLU
S2	최대 풀링	96	27×27	3×3	2	valid	-
C1	합성곱	96	55×55	11×11	4	valid	ReLU
입력	입력	3 (RGB)	227×227	-	-	-	-

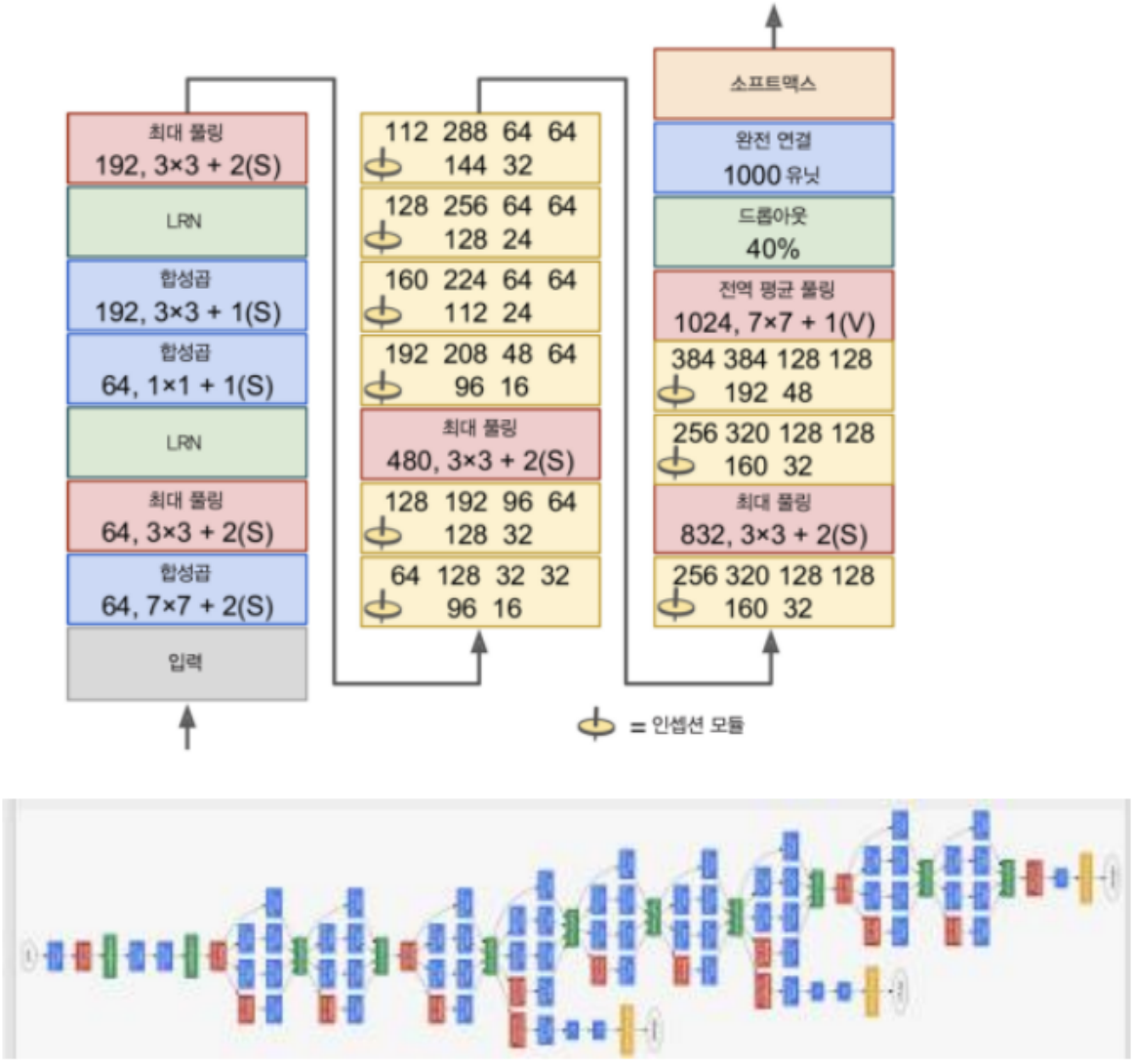


14.4.3 GoogLeNet

*GoogLeNet
-2014년 ILSVRC 이미지넷 대회에서 우승
-톱-5 에러율: 7% 이하로 낮춤 -> 이전 CNN보다 훨씬 더 깊기 때문
-인셉션 모듈이라는 서브 네트워크 사용
-이전의 구조보다 훨씬 효과적으로 파마미터를 사용
-GoogLeNet은 AlexNet 보다 10배나 적은 파라미터를 가짐. (6천만개 -> 6백만개)
*인셉션 모듈

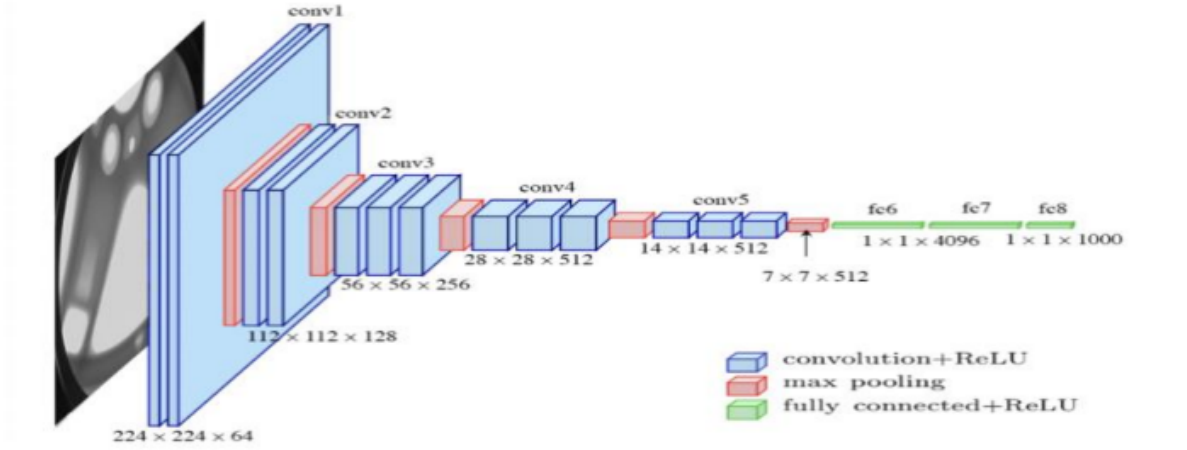


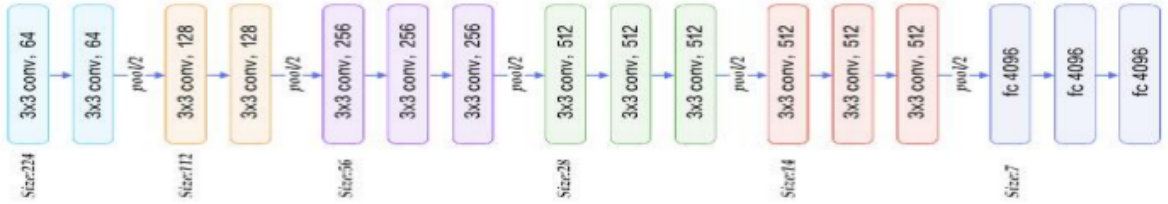
14.4.3 GoogLeNet (2)



14.4.4 VGGNet

VGGNet 구조
-ILSVRC 2014년 대회 2등
-톱-5 에러율: 8~10%
-합성곱 계층과 풀링 계층으로 구성되는 기본적인 CNN
-합성곱 계층, 완전연결 계층을 모두 16층(혹은 19층)으로 심화함.
-특징은 많은 개수의 필터를 사용하지만, 3x3 필터만을 사용함.

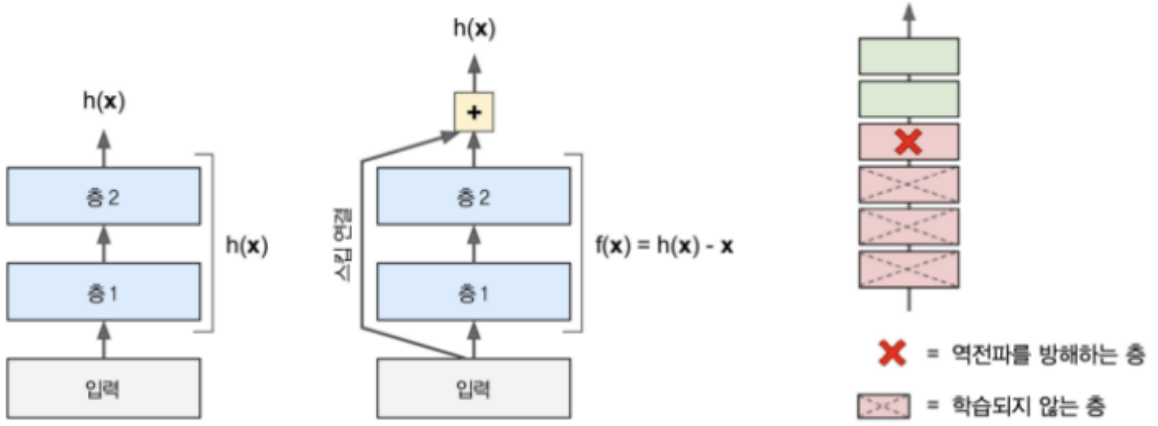




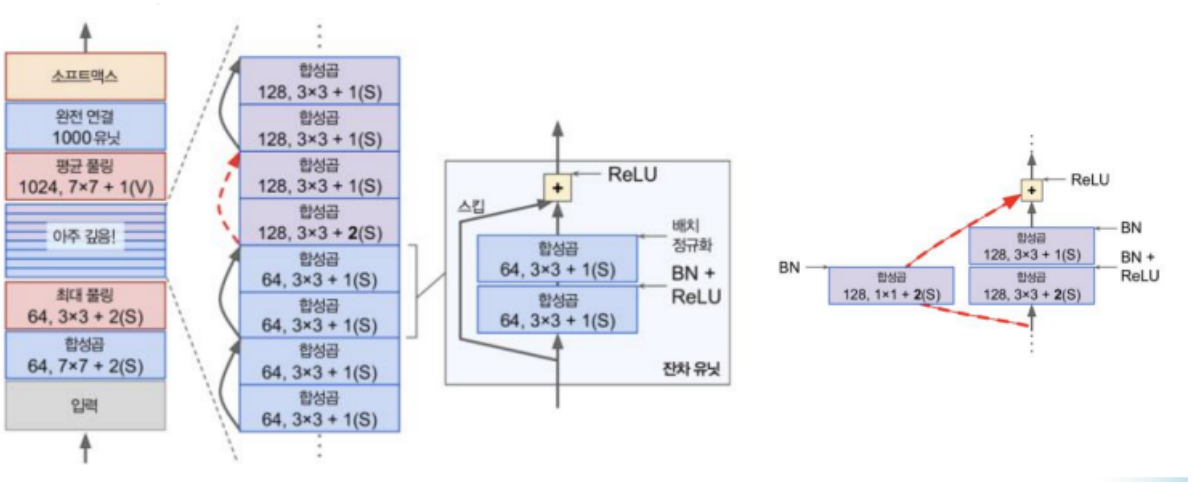
14.4.5 ResNet

*ResNet
-2015년 ILSVRC 이미지넷 대회에서 우승
-톱-5 에러율: 3.6% 이하
-잔차 네트워크(Residual Network) 사용

*잔차 유닛(residual unit, RU):인한 많은 계산을 줄이기 위해
-RU를 활용한 잔차학습(residual learning) 효과: 스킵 연결로 인한 보다 수월한 학습 가능



14.4.5 ResNet (1)



14.4.6 Xception

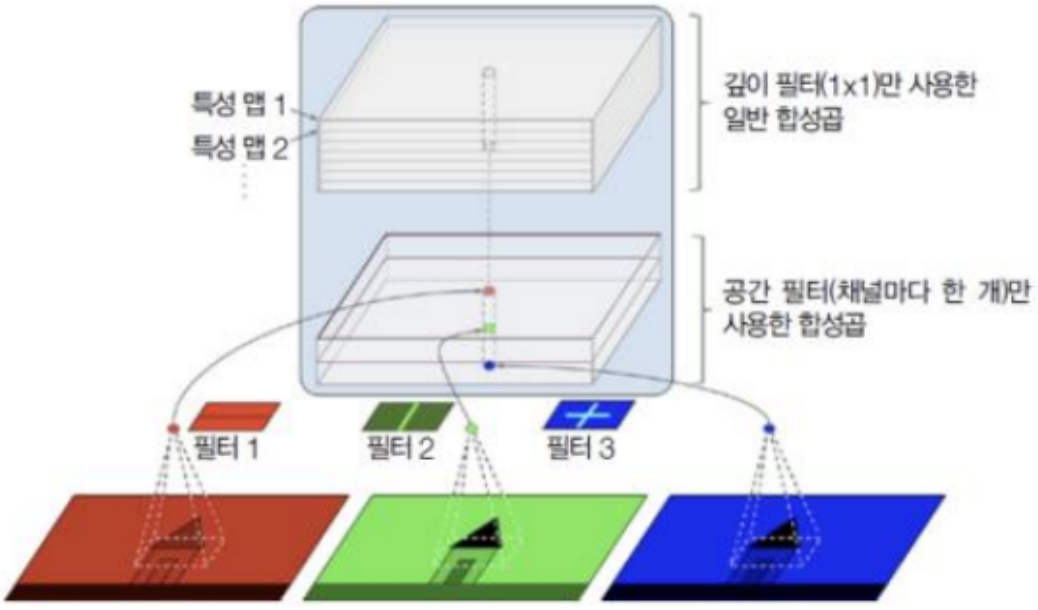
-2016년에 소개된 GoogLeNet 과 ResNet 모델의 합성 버전

-톱-5 에러율: 3% 정도

-GoogLeNet의 인셉션 모듈 대신 깊이별 분리합성곱 층 사용

*분리 합성곱 층:계산량 줄음

-공간별 패턴인식 합성곱 층과 깊이별 패턴인식 합성곱 층을 분리하여 연속적으로 적용



14.4.6 Xception (1)

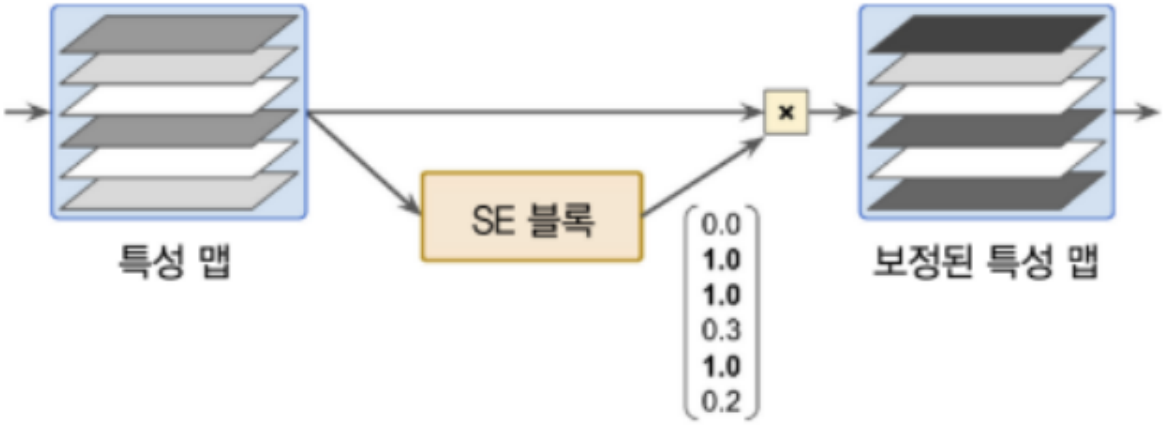
*Xception 구조
• 보통 두 개 정도의 정상적인 합성곱 층으로 시작한 후에 깊이별 분리합성곱 층 적용

14.4.7 SENet

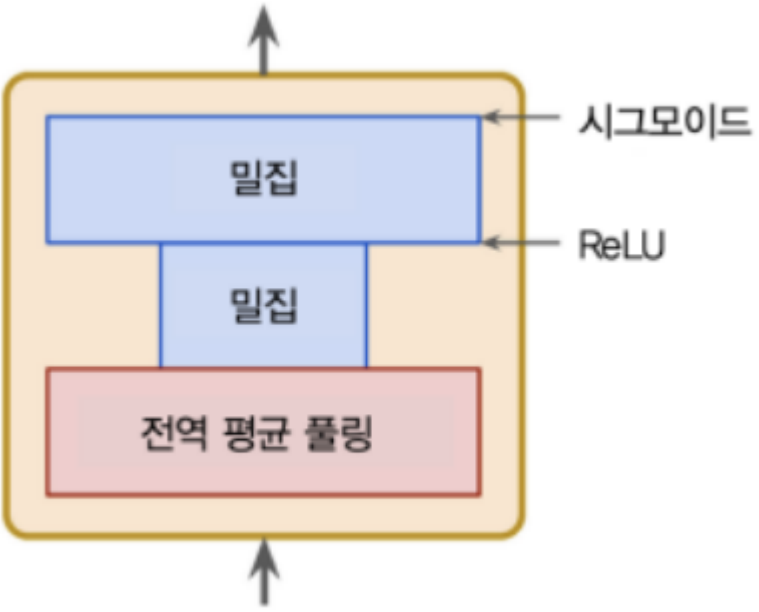
-2017년 ILSVRC 이미지넷 대회에서 우승
-톱-5 에러율: 2.25%
-GoogLeNet의 인셉션 모듈과 ResNet의 잔차유닛(RU)에 SE block을 추가하여 보다 좋은 성능 발휘

14.4.7 SENet (1)

SE block 기능
-입력된 특성맵을 대상으로 깊이별 패턴 특성 분석
-패턴 특성들을 파악한 후 출력값 보정
EX) 코와 입의 패턴이 보일 때 눈의 특성지도를 강화시킴.



SE block 구조
•-학습된 연관성을 이용하여 입력 특성지도를 보정할 가중치 출력



14.4.8 Clould 에서 Mobile로

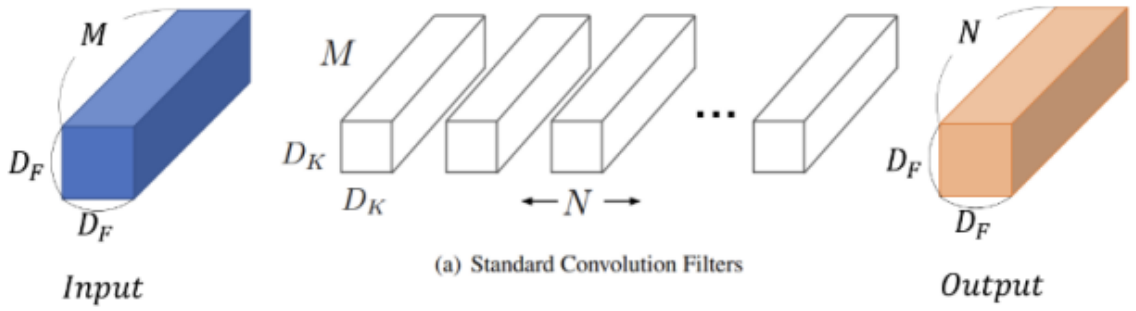
	ResNet Cloud AI	MobileNet Mobile AI	MCUNet Tiny AI
Memory (Activation)	16GB	4GB ← 13,000x smaller	320kB
Storage (Weights)	~TB/PB	256GB	1MB

14.4.8 딥러닝 모델 사이즈

→ 대부분 상당한 크기

14.4.8 MobileNet

-스마트폰 및 기타 모바일 장치와 같이 리소스가 제한된 환경에서 효율적인 계산을 위해 설계된 경량 심층신경망으로 2017년 구글에서 개발
- MobileNet은 모바일 기기에서 돌아갈 수 있을만큼 경량한 구조를 설계하는데 집중

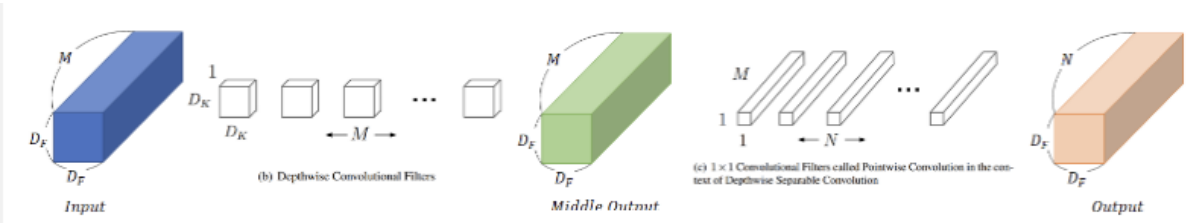


$$G_{convolution} = D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F$$

-Depthwise Separable Convolution의 연산량

→분리해서 연산

- 공간에 대한 분석인 depthwise convolution 연산량
- 채널에 대한 분석인 pointwise convolution 연산량
- 커널 크기가 3인 경우를 예로 들면연산량이 약 8~9배나 줄어듬.



$$G_{dsc} = D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F$$

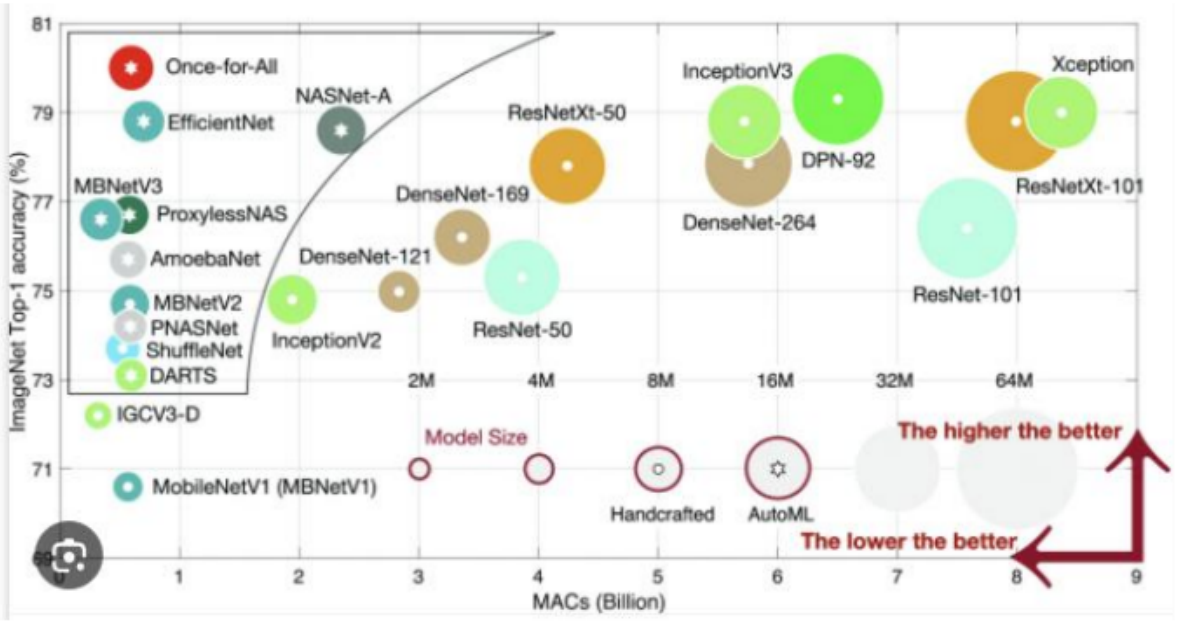
14.4.8 경량알고리즘 연구

	접근방법	연구 방향
경량 알고리즘 연구	모델 구조 변경	잔여 블록, 병목 구조, 밀집 블록 등 다양한 신규 계층 구조를 이용하여 파라미터 축소 및 모델 성능을 개선하는 연구(ResNet, DenseNet, SqueezeNet)
	합성곱 필터 변경	합성곱 신경망의 가장 큰 계산량을 요구하는 합성곱 필터의 연산을 효율적으로 줄이는 연구(MobileNet, ShuffleNet)
	자동 모델 탐색	특정 요소(지연시간, 에너지 소모 등)가 주어진 경우, 강화 학습을 통해 최적 모델을 자동 탐색하는 연구(NetAdapt, MNasNet)
알고리즘 경량화 연구	모델 압축	가중치 가지치기, 양자화/이진화, 가중치 공유 기법을 통해 파라미터의 불필요한 표현력을 줄이는 연구(Deep Compression, XNOR-Net)
	지식 증류	학습된 기본 모델을 통해 새로운 모델의 생성 시 파라미터값을 활용하여 학습시간을 줄이는 연구(Knowledge Distillation, Transfer Learning)
	하드웨어 가속화	모바일 기기를 중심으로 뉴럴 프로세싱 유닛(NPU)을 통해 추론 속도를 향상시키는 연구
	모델 압축 자동 탐색	알고리즘 경량화 연구 중 일반적인 모델 압축 기법을 적용한 강화 학습 기반의 최적 모델 자동 탐색 연구(PocketFlow, AMC)

Model	Size in h5 (MB)	Size in tflite (MB)
MobileNetV2	26	8.54
EfficientNet-B0	47	15.3
EfficientNet-B1	76	24.85
EfficientNet-B2	90	29.37
EfficientNet-B3	124	40.78
EfficientNet-B4	203	66.87

14.4.8 AutoMLs

→사람제작 vs 모델이 제작


ImageNet Top-1 accuracy (%)
MACs (Billion)
Once-for-All
EfficientNet
NASNet-A
MBNetV3
ProxylessNAS
AmoebaNet
MBNetV2
PNASNet
ShuffleNet
DARTS
IGCV3-D
MobileNetV1 (MBNetV1)
InceptionV2
DenseNet-121
DenseNet-169
ResNet-50
ResNetXt-50
InceptionV3
DPN-92
DenseNet-264
Xception
ResNetXt-101
ResNet-101
Model Size
Handcrafted
AutoML
2M
4M
8M
16M
32M
64M
The higher the better
The lower the better