

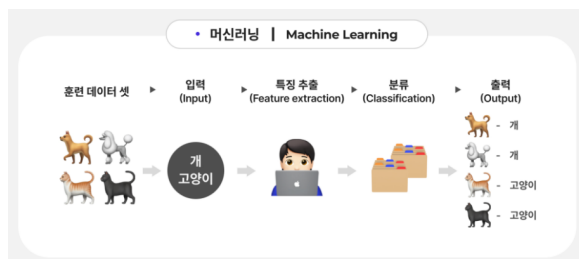


## SEMES - 볼트 이상 진단 AI 모델 개발

이미지 분류 AI 모델 (정상 / 파손 / 유실 분류용)

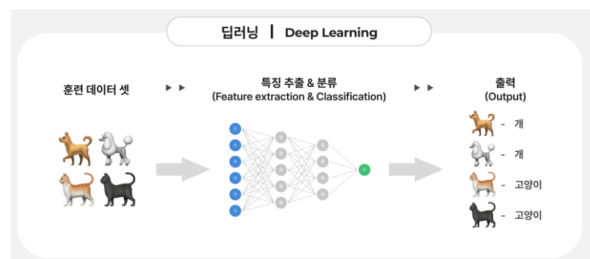
### 머신러닝 vs 딥러닝

#### 머신러닝



- 머신러닝은 주어진 데이터를 인간이 먼저 처리
1. 이미지 데이터의 클래스를 정의하여 데이터 셋을 만들어 준다
  2. 사람이 먼저 컴퓨터에 특정 패턴을 추출하는 방법을 지시
  3. 그 이후 컴퓨터가 스스로 데이터의 특징을 분석하고 축적
  4. 이미지를 특징에 맞게 평가하여 분류된 결과를 얻는다

#### 딥러닝



- 딥러닝은 머신러닝에서 사람이 하던 패턴 추출 작업이 생략
1. 이미지 데이터의 클래스를 정의하여 데이터 셋을 만들어 준다
  2. 컴퓨터가 스스로 데이터를 기반으로 학습할 수 있도록 정해진 신경망을 컴퓨터에게 줌
  3. 어린아이가 학습하는 것처럼 경험 중심으로 학습을 수행
  4. 이미지를 인공 신경망에 평가하여 분류된 결과를 얻는다

#### ⇒ 딥러닝 방식 선정

- ⇒ 정답지가 있는 학습 방법인 지도 학습과 정답지가 없는 학습 방법인 비지도 학습
- ⇒ 시간 단축을 하고, 작은 데이터셋으로도 성능 향상을 위해 지도학습 방식 선정

- 모델 선정 : Pytorch / Tensorflow에서 지원하는 모델 중 선정
  - 모델의 정의 : 여러 가지 방법론의 구현체

#### Pytorch

- ▼ 지원모델
  - AlexNet
  - ConvNeXt
  - DenseNet

#### Tensorflow

- ▼ 지원모델
  - EfficientNetB7
  - EfficientNetB6
  - EfficientNetB5

- [EfficientNet](#)
- [EfficientNetV2](#)
- [GoogLeNet](#)
- [Inception V3](#)
- [MaxVit](#)
- [MNASNet](#)
- [MobileNet V2](#)
- [MobileNet V3](#)
- [RegNet](#)
- [ResNet](#)
- [ResNeXt](#)
- [ShuffleNet V2](#)
- [SqueezeNet](#)
- [SwinTransformer](#)
- [VGG](#)
- [VisionTransformer](#)
- [Wide ResNet](#)

- [EfficientNetB4](#)
- [NASNetLarge](#)
- [EfficientNetB3](#)
- [InceptionResNetV2](#)
- [EfficientNetB2](#)
- [EfficientNetB1](#)
- [Xception](#)
- [ResNet152V2](#)
- [InceptionV3](#)
- [DenseNet201](#)
- [ResNet101V2](#)
- [EfficientNetB0](#)
- [ResNet152](#)
- [ResNet101](#)
- [DenseNet169](#)
- [ResNet50V2](#)
- [DenseNet121](#)
- [ResNet50](#)

▼ 정확도 - Pytorch

Weight	Acc@1	Model	Acc@5	Params	GFLOPS	Size
<a href="#">ViT_H_14_Weights.IMAGENET1K_SWAG_E2E_V1</a>	88.552		98.694	633.5M	1016.72	
<a href="#">RegNet_Y_128GF_Weights.IMAGENET1K_SWAG_E2E_V1</a>	88.228		98.682	644.9M	944.57	
<a href="#">ViT_L_16_Weights.IMAGENET1K_SWAG_E2E_V1</a>	88.064		98.512	305.2M	361.99	
<a href="#">RegNet_Y_32GF_Weights.IMAGENET1K_SWAG_E2E_V1</a>	86.838	<a href="#">EfficientNetB7</a>	98.362	145.0M	97.00%	256
<a href="#">RegNet_Y_128GF_Weights.IMAGENET1K_SWAG_LINEAR_V1</a>	86.068	<a href="#">EfficientNetB6</a>	97.844	644.8M	96.80%	166
<a href="#">RegNet_Y_16GF_Weights.IMAGENET1K_SWAG_E2E_V1</a>	86.012	<a href="#">EfficientNetB5</a>	98.054	83.6M	96.70%	118
<a href="#">EfficientNet_V2_L_Weights.IMAGENET1K_V1</a>	85.808	<a href="#">EfficientNetB4</a>	97.788	118.5M	96.40%	75
<a href="#">ViT_H_14_Weights.IMAGENET1K_SWAG_LINEAR_V1</a>	85.708	<a href="#">NASNetLarge</a>	97.73	632.0M	96.00%	343
<a href="#">ViT_B_16_Weights.IMAGENET1K_SWAG_E2E_V1</a>	85.304	<a href="#">EfficientNetB3</a>	97.65	86.9M	95.70%	48
<a href="#">ViT_L_16_Weights.IMAGENET1K_SWAG_LINEAR_V1</a>	85.146	<a href="#">InceptionResNetV2</a>	97.422	304.3M	95.30%	215
<a href="#">EfficientNet_V2_M_Weights.IMAGENET1K_V1</a>	85.112	<a href="#">EfficientNetB2</a>	97.156	54.1M	94.90%	36
<a href="#">RegNet_Y_32GF_Weights.IMAGENET1K_SWAG_LINEAR_V1</a>	84.622	<a href="#">EfficientNetB1</a>	97.48	145.0M	94.40%	31
<a href="#">ConvNeXt_Large_Weights.IMAGENET1K_V1</a>	84.414	<a href="#">Xception</a>	96.976	197.8M	94.50%	88
<a href="#">EfficientNet_V2_S_Weights.IMAGENET1K_V1</a>	84.228	<a href="#">ResNet152V2</a>	96.878	21.5M	94.20%	232
<a href="#">EfficientNet_B7_Weights.IMAGENET1K_V1</a>	84.122	<a href="#">InceptionV3</a>	96.908	66.3M	93.70%	92
<a href="#">Swin_V2_B_Weights.IMAGENET1K_V1</a>	84.112	<a href="#">DenseNet201</a>	96.864	87.9M	93.60%	80
<a href="#">ConvNeXt_Base_Weights.IMAGENET1K_V1</a>	84.062	<a href="#">ResNet101V2</a>	96.87	88.6M	93.80%	171
<a href="#">EfficientNet_B6_Weights.IMAGENET1K_V1</a>	84.008	<a href="#">EfficientNetB0</a>	96.916	43.0M	93.30%	29
<a href="#">RegNet_Y_16GF_Weights.IMAGENET1K_SWAG_LINEAR_V1</a>	83.976	<a href="#">ResNet152</a>	97.244	83.6M	93.10%	232
		<a href="#">ResNet101</a>			92.80%	171
		<a href="#">DenseNet169</a>			93.20%	57
		<a href="#">ResNet50V2</a>			93.00%	98
		<a href="#">DenseNet121</a>			92.30%	33
		<a href="#">ResNet50</a>			92.10%	98

• 정확도가 높은 모델

▼ ViT(Vision Transformer)

- 특징: Transformer 기반 비전 모델, CNN 대신에 self-attention 기술 사용, 이미지 패치를 벡터로 변환한 후 Transformer 구조에 입력

• 정확도가 높은 모델

▼ EfficientNet

- 장점: 기존 CNN보다 높은 성능을 보이며, input size가 큰 이미지에도 적용 가능, 대규모 데이터셋에서 다른 모델보다 우수한 성능을 보임
- 단점: 많은 데이터와 연산량이 필요, 속도가 느리고 복잡한 구조로 인해 모델 크기가 큼

#### ▼ RegNet(Regularization Network)

- 특징: 네트워크 구조를 효율적으로 구성하는 알고리즘을 개발한 ResNet의 발전형 모델, 모듈(Module) 구조를 조합하여 생성
- 장점: 모듈 구조의 조합으로 인해 모델 크기를 줄이면서도 성능을 유지, 작은 모델에서도 큰 모델과 비슷한 성능을 보임
- 단점: 다른 모델과 마찬가지로 학습 데이터가 적을 때 성능이 떨어짐

#### ▼ EfficientNet

- 특징: 네트워크 깊이, 너비, 해상도를 조정하여 모델의 성능과 효율성을 극대화한 모델
- 장점: 더 적은 연산량과 파라미터로도 높은 성능을 보이며, 다양한 해상도 입력 이미지에 대해 일관된 성능을 보임
- 단점: 다른 모델보다 뛰어난 성능을 보이는 만큼, 모델의 구조가 복잡하고 학습에 시간이 많이 걸림. 특정 문제에 최적화된 모델보다는 일반화 성능이 더 뛰어나기 때문에 특정 문제에 대한 최적화가 필요할 수 있음.

- 특징: 네트워크 깊이, 너비, 해상도를 조정하여 모델의 성능과 효율성을 극대화한 모델
- 장점: 더 적은 연산량과 파라미터로도 높은 성능을 보이며, 다양한 해상도 입력 이미지에 대해 일관된 성능을 보임
- 단점: 다른 모델보다 뛰어난 성능을 보이는 만큼, 모델의 구조가 복잡하고 학습에 시간이 많이 걸림. 특정 문제에 최적화된 모델보다는 일반화 성능이 더 뛰어나기 때문에 특정 문제에 대한 최적화가 필요할 수 있음.

#### ▼ NASNet

- 특징: 자동으로 최적의 딥러닝 모델을 탐색하는 Neural Architecture Search 기술을 이용한 모델, 높은 성능과 일반화 능력을 가짐
- 장점: 자동으로 최적의 구조를 탐색하여 최적화된 모델을 생성, 대규모 데이터셋에서 높은 성능을 보임
- 단점: 모델의 구조가 복잡하여 속도가 느릴 수 있음, 모델 설계가 어려워 인간의 도움이 필요함

#### ▼ InceptionResNet

- 특징: Inception 모듈과 ResNet 모듈을 결합하여 생성한 모델, 더 나은 성능과 효율성을 제공
- 장점: Inception 모듈의 다양한 필터 크기와 ResNet 모듈의 스킵 연결을 활용하여 성능을 향상, 작은 모델에서도 큰 모델과 비슷한 성능을 보임
- 단점: 모델의 구조가 복잡하여 속도가 느릴 수 있음

#### ▼ Xception

- 특징: Inception 모델의 구조를 개선하여 생성한 모델, Depthwise Separable Convolution을 이용하여 더 효율적인 모델을 만들
- 장점: Inception 모델보다 더 높은 성능과 효율성을 제공하며, 작은 모델에서도 큰 모델과 비슷한 성능을 보임
- 단점: 모델의 구조가 복잡하여 속도가 느릴 수 있음, 일부 문제에서 다른 모델보다 성능이 떨어질 수 있음.

#### ▼ ResNet

- 특징: 신경망의 깊이가 증가하면서 발생하는 그레디언트 소실 문제(gradient vanishing problem)를 해결하기 위해 스킵 연결(skip connection)을 사용한 모델
- 장점: 깊은 모델에서도 높은 정확도를 보임, 스킵 연결을 통해 기울기가 손실되지 않고 정보가 흐를 수 있도록 함
- 단점: 깊은 모델에서는 성능이 좋지만, 모델의 깊이가 적을 때는 다른 모델과 성능 차이가 크지 않을 수 있음

#### • 추천 모델

##### 1. RegNet

- Pytorch만 지원
- 페이스북 AI 연구팀에서 발표한 모델
- 모델 구조를 효율적으로 설계하고, regularization 기법을 사용하여 overfitting을 방지하고 성능을 개선
- 모델의 구조가 단순하여 파라미터 수가 적어 상대적으로 모델의 크기가 작고, 학습 시간이 짧음
- 모델의 너비와 깊이를 자유롭게 조절

## 2. EfficientNet

- Pytorch와 Tensorflow 모두 지원
- google에서 발표한 모델
- 31~256MB까지 다양한 크기에서 선택 가능(크기와 정확도는 비례)
- 파라미터 수와 연산량이 효율적으로 조정되어 있어 더 높은 정확도와 더 적은 연산량
- 네트워크의 깊이, 너비, 해상도(scale)를 동시에 최적화하여 높은 정확도
- 모델의 특징 추출 부분인 Backbone Network(BN)과 분류기(Classifier)를 따로 학습할 수 있도록 모듈화