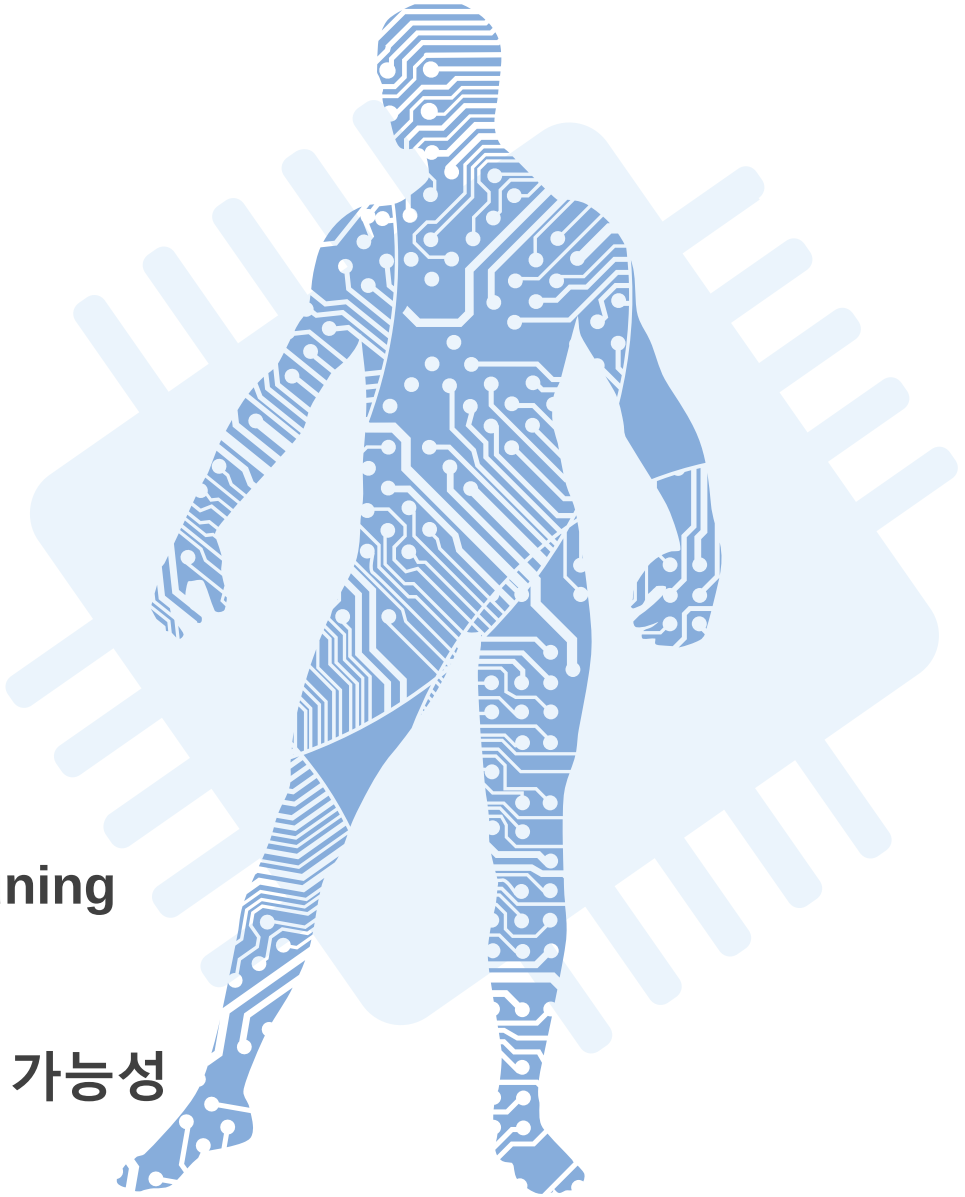
AI Intro

Deep Learning AI

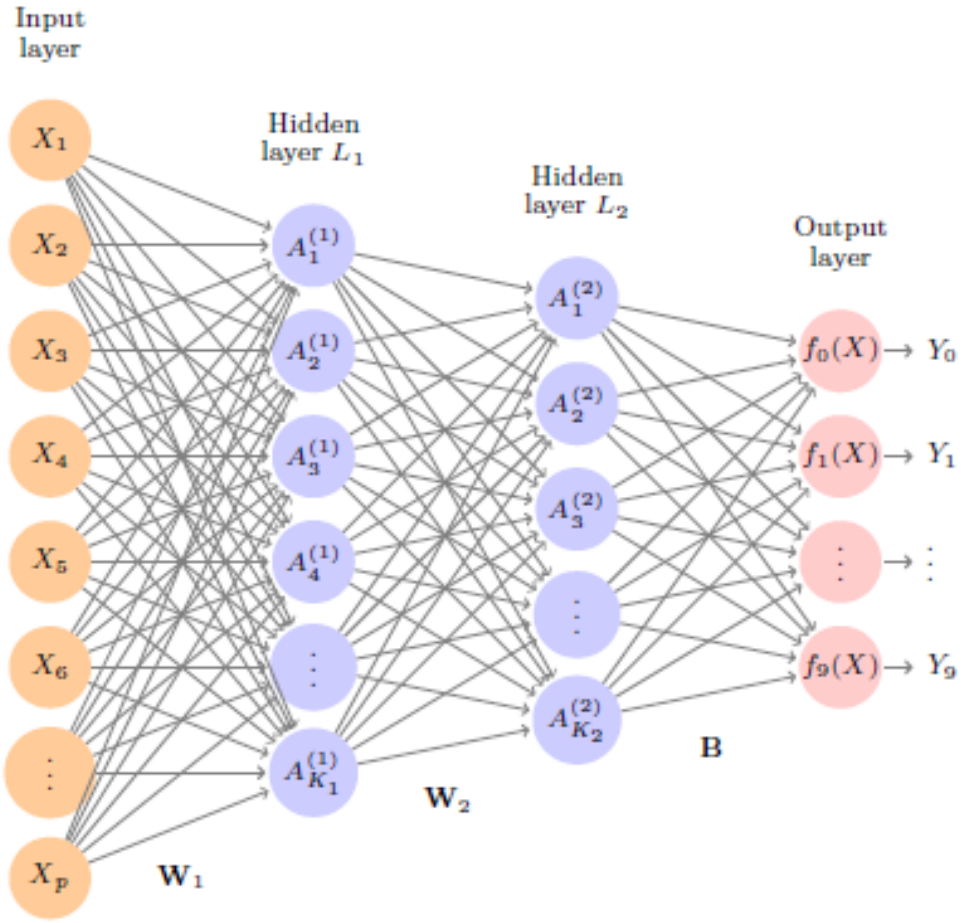
Deep Learning AI와 적용방안

발표자: 고동희 차장

Table of Contents



Deep Learning AI란 무엇인가?



- 딥러닝AI는 기계학습의 하위 분야이며, 데이터에서 표현 혹은 패턴(representation or patterns)을 학습하기 위한 수학적 프레임워크라고 정의할 수 있음
- 딥러닝에서 "Deep"은 모델의 더 깊은 이해를 의미하는 것이 아님. 딥러닝 모델은 기여하는 레이어의 수에 따라 모델의 깊이(Depth)가 결정되며 모델에 사용되는 레이어가 많을수록 모델이 깊다(Deep)는 것을 의미함
- 따라서 딥러닝의 더 적절한 이름은 계층표현학습(layered representations learning)또는 위계구조적 표현학습(hierarchical representations learning)임
- 일반적으로 딥러닝 모델은 입력층, 여러 개의 은닉층(Hidden layer), 그리고 출력층으로 이루어짐
- 딥러닝은 연속적인 은닉계층(hidden layers)을 통해 데이터에서 점점 더 의미있는 표현 혹은 패턴을 학습 하는 모델임.

Deep Learning 알고리즘이 학습하는 표현은 어떻게 생겼을까?

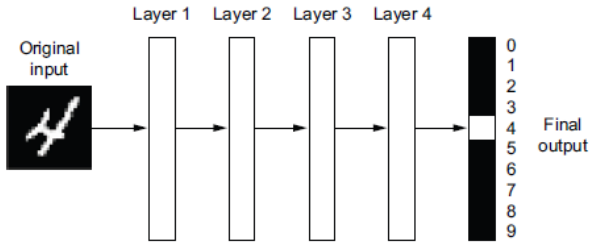


Figure 1.5 A deep neural network for digit classification

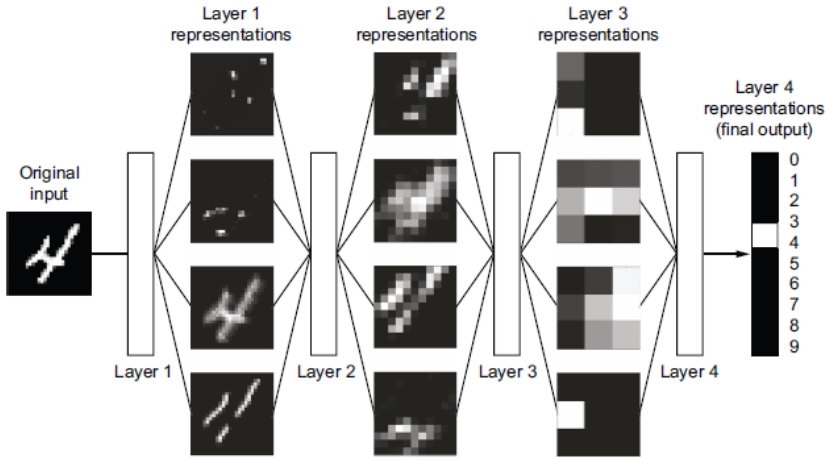
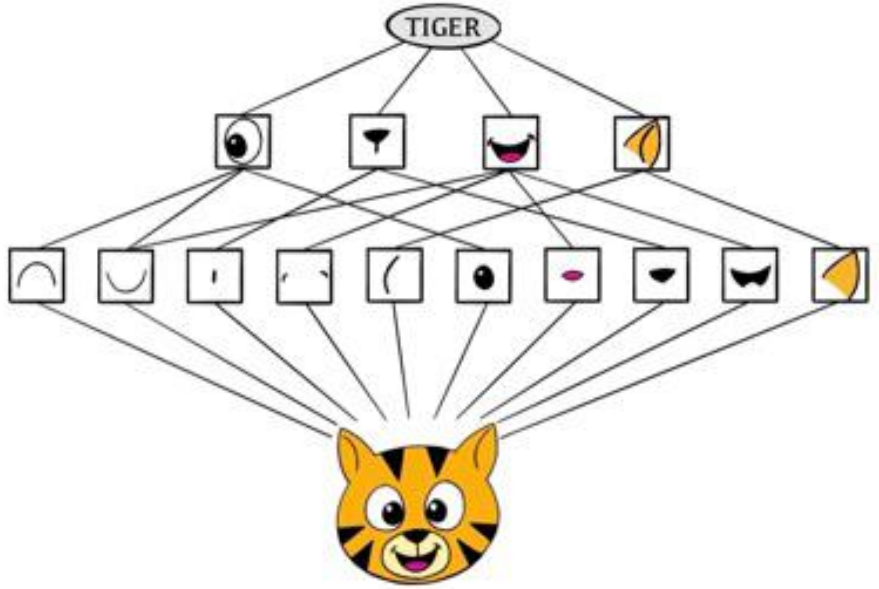


Figure 1.6 Data representations learned by a digit-classification model

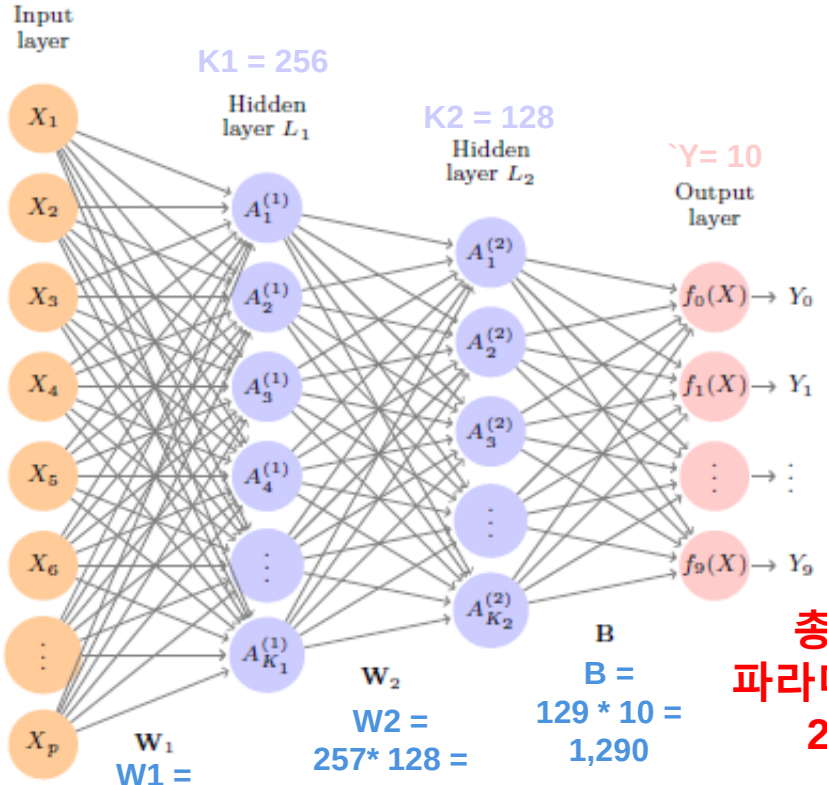


- 위 그림에서 볼 수 있듯이 딥러닝 네트워크는 원본 이미지를 원래의 이미지와는 점점 다르고 최종 결과에 대해 점점 더 많은 정보를 제공하는 표현으로 변환 함.

- 쉽게 말하면, 딥러닝 네크워크는 입력 정보가 연속적인 필터를 거쳐 점점 더 정제되어 나오는 다단계 정보 증류 과정으로 생각할 수 있음

Deep Learning 작동원리(딥러닝 vs 다중회귀분석)

- 각 레이어 계층에서 적용되는 데이터 변환은 가중치에 의해서 결정된다.
- 딥러닝에서 학습은 네트워크 모든 레이어의 가중치를 최적화 시켜 입력데이터를 해당 타겟 테이터에 올바르게 매핑하는 과정을 의미한다.
- 하지만 다른 모델들과 달리 딥러닝 네트워크는 학습시켜야 할 파라미터의 개수가 적게는 수십만개에서 많게는 수백(천)억개에 달하기 때문에 네트워크의 최적화된 가중치를 찾아내는 일은 생각만큼 쉬운 일은 아니다. 그렇다면 딥러닝 알고리즘은 어떻게 가중치를 최적화 할 수 있을까?



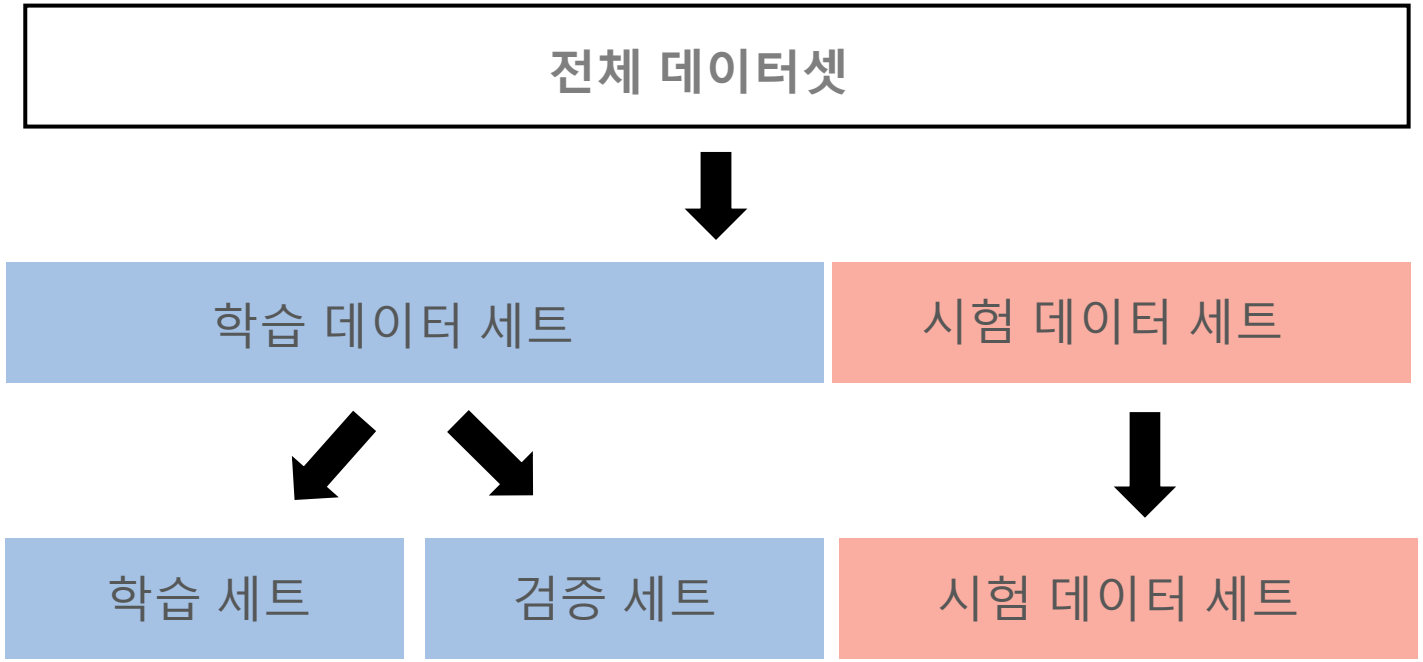
$$Y = \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_4 + \beta_0 + \epsilon$$

총 필요한
파라미터 개수 = 4

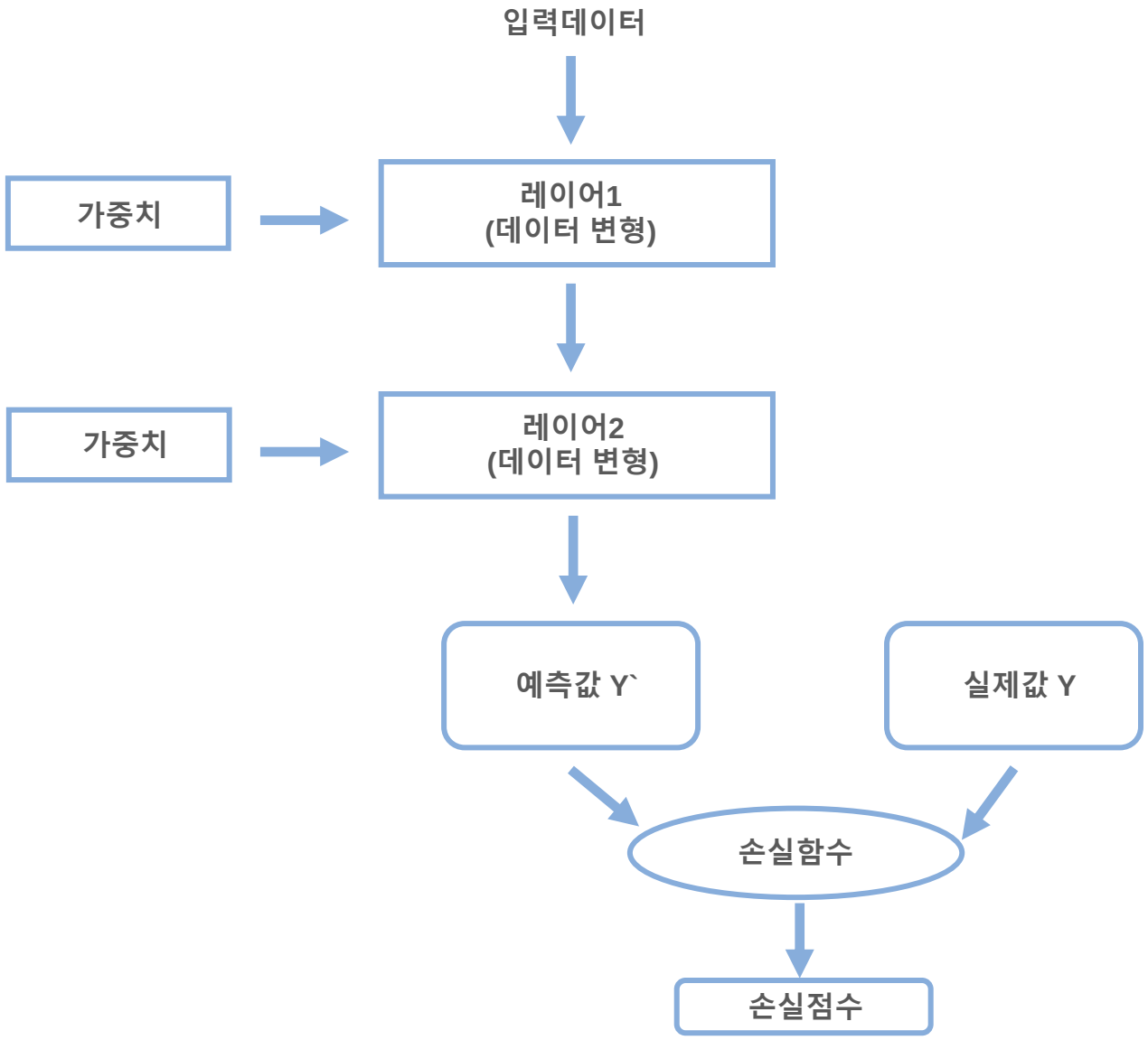
딥러닝 학습데이터 분할

데이터 분할은 딥러닝 모델을 학습시키는 가장 중요한 단계 중 하나로, 데이터는 학습 세트(training set), 검증세트(validation set), 그리고 시험세트(testing set)로 구분한다. 학습세트(training set)는 모델을 실제로 학습시키는데 사용하며, 검증세트(validation set)는 학습 중 모델의 성능을 모니터링하거나 하이퍼파라미터를 조정하는데 사용된다. 시험세트(testing set)는 모델의 최종 성능을 평가하는데 사용되며, 이는 모델이 이전에 본 적이 없는 데이터에 대해 얼마나 일반화되었는지를 평가하는데 사용된다.

1) 딥러닝의 경우 최적화 시켜야 할 가중치의 수가 많다 2) 이를 위해 데이터셋을 분할해서 학습을 진행해야 한다 ➔ 인공지능 학습을 위한 일정 수준 이상의 데이터 확보는 프로젝트의 성공여부를 결정하는 요소 중 하나임.

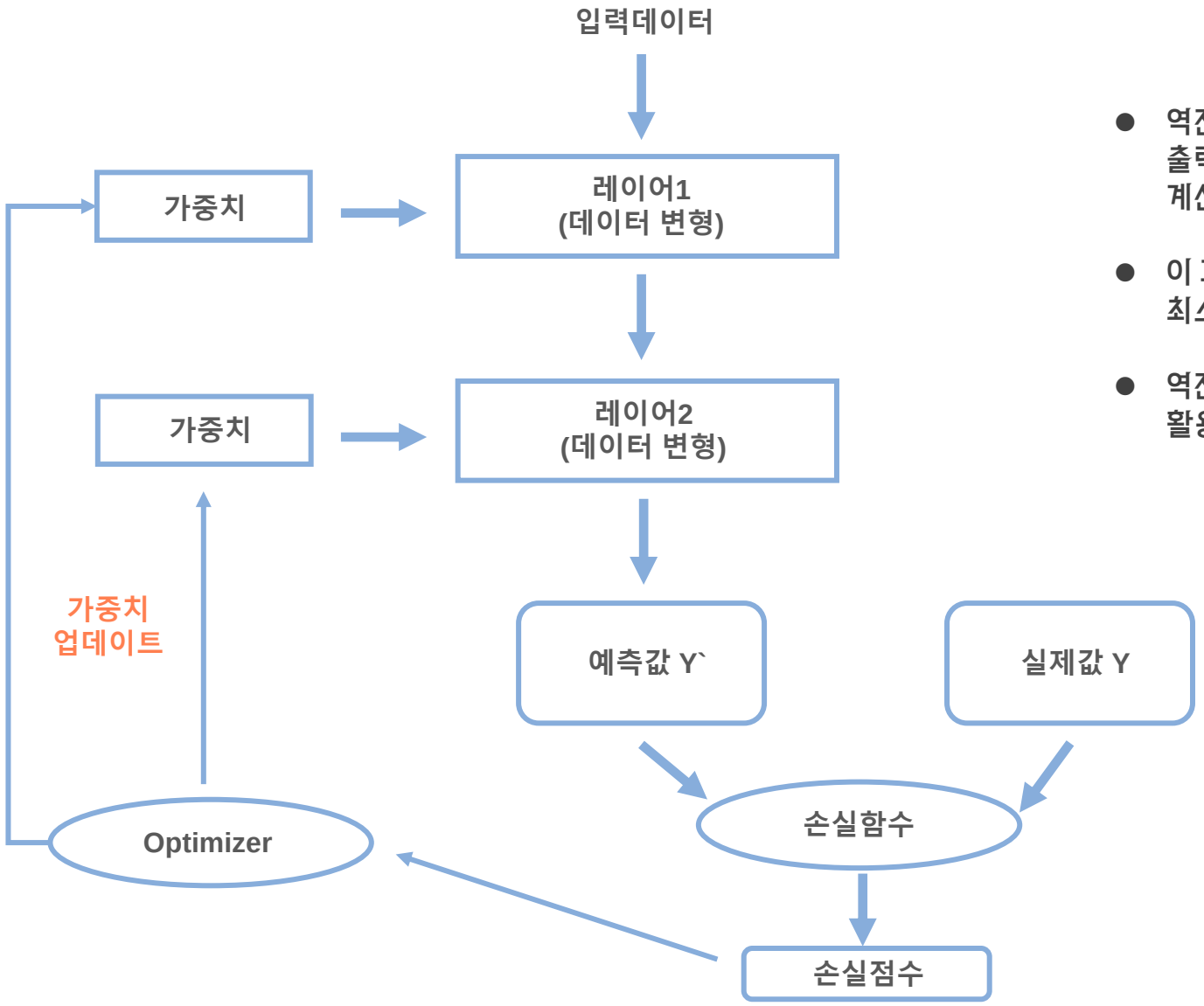


순방향 (Feed Forward) 알고리즘

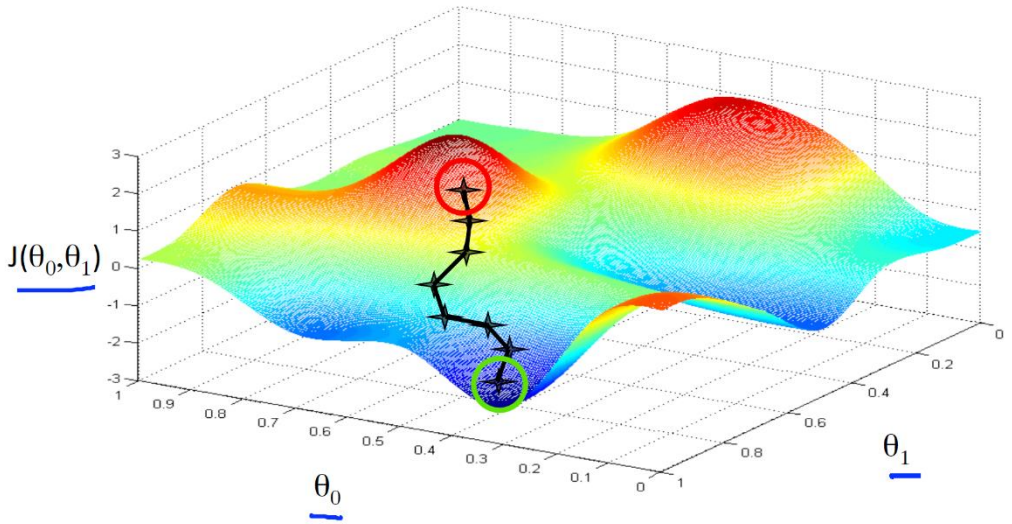


- 먼저 딥러닝 네트워크에서 입력된 데이터는 입력층에서 출력층으로 순방향(feed-forward)이동을 하며 이 때 각 레이어의 출력값은 다음 계층의 입력값이 된다.
- 이 방식으로 데이터는 각 레이어를 통해 변형되고 최종적으로 출력층에서 결과값을 생성한다.
- 딥러닝 모델을 제어하여 최적의 가중치를 계산하기 위해서는 모델의 성능(performance)을 측정할 수 있어야 한다.
- 신경망 모델의 성능은 모델의 출력값과 실제 기대값 사이의 차를 계산하여 측정하며, 이때 사용되는 거리 함수를 손실함수(loss function) 혹은 비용함수(cost function)라 한다.
- 딥러닝 학습의 핵심은 이 손실함수를 피드백(feedback) 시그널로 이용하여 모델의 가중치를 최적화하는 것이며, 이를 위하여 딥러닝 모델은 역전파(back-propagation)알고리즘을 사용한다.

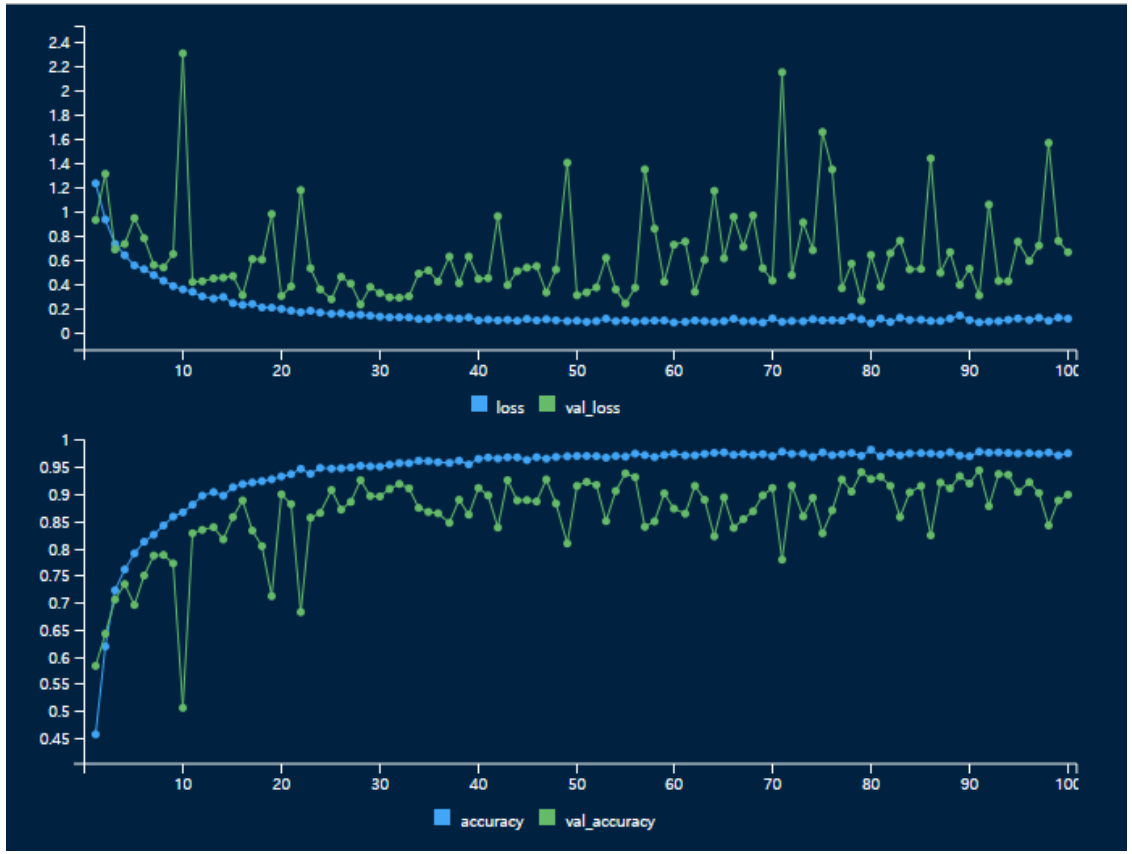
역전파 (Backpropagation)와 경사하강법



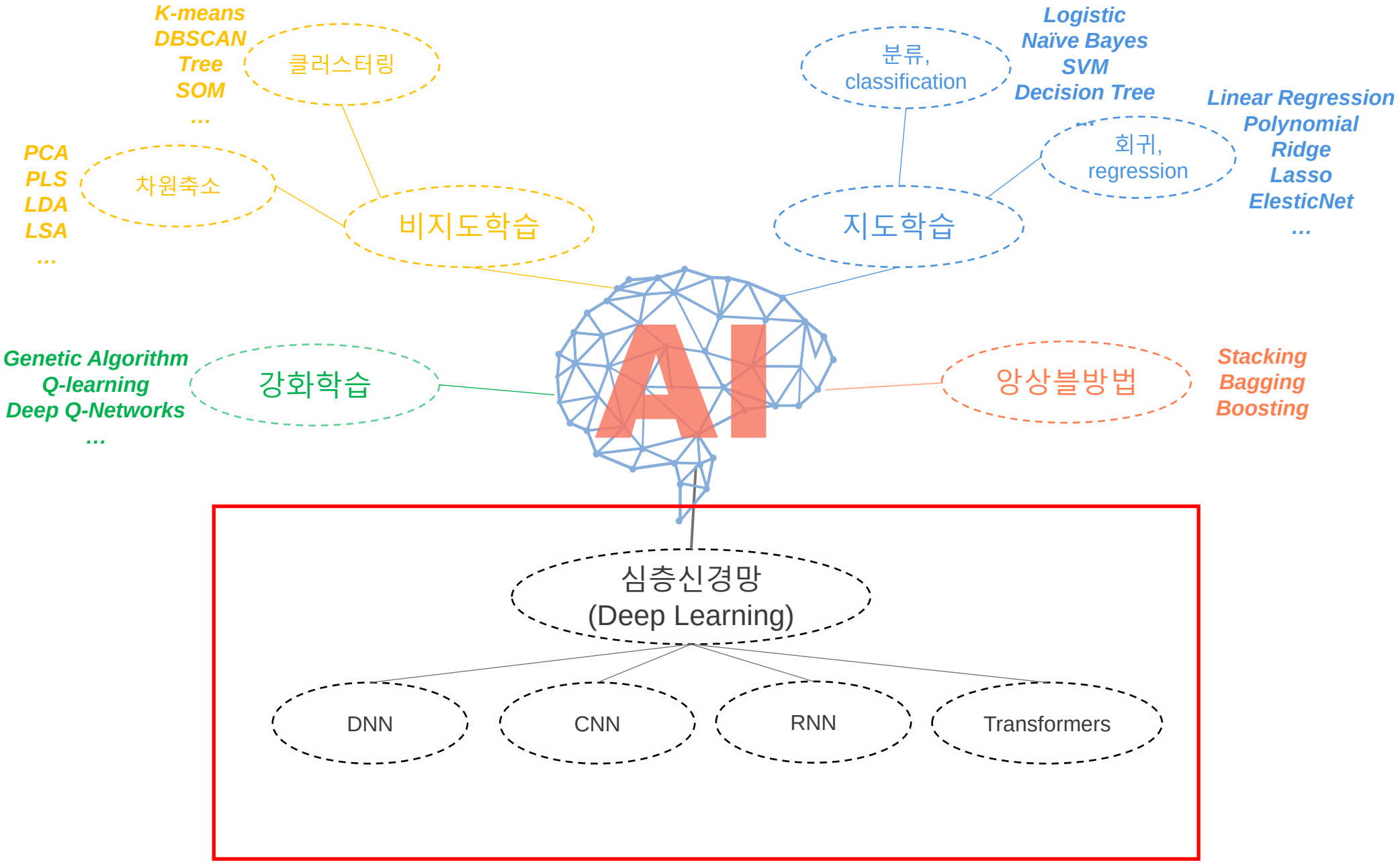
- 역전파 알고리즘은 신경망에서 가중치를 조정하기 위한 핵심 알고리즘으로, 출력층에서 발생한 오차를 다시 입력층으로 전파하여 각 노드의 오차 기여도를 계산하고 가중치를 업데이트 하는 과정이다.
- 이 과정의 반복을 통해 신경망은 예측 값과 실제 목표 값 사이의 차이를 최소화하도록 조정된다.
- 역전파알고리즘은 가중치 최적화를 위해 경사하강법(gradient descent)을 활용한다.



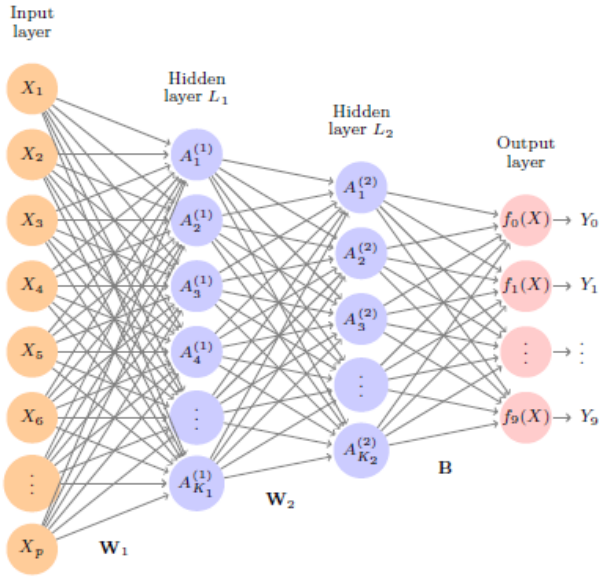
심층신경망 학습 히스토리 로그



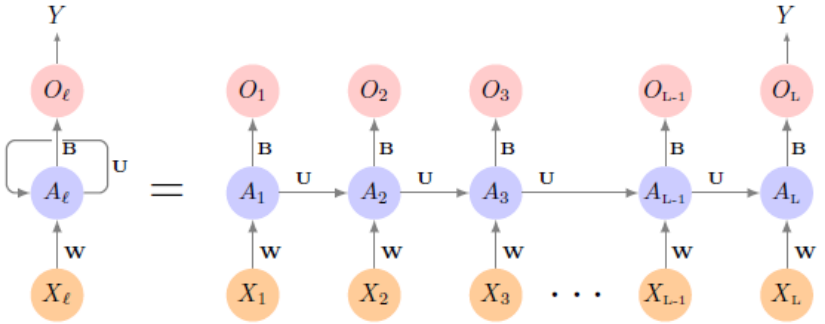

```
Epoch 1/100
113/113 [==============================] - 11s 69ms/step - loss: 1.2693 - accuracy: 0.4419 - val_loss: 1.0651 - val_accuracy: 0.5344
Epoch 2/100
113/113 [==============================] - 8s 67ms/step - loss: 0.9162 - accuracy: 0.6236 - val_loss: 0.8312 - val_accuracy: 0.6181
Epoch 3/100
113/113 [==============================] - 8s 68ms/step - loss: 0.7424 - accuracy: 0.7050 - val_loss: 1.6713 - val_accuracy: 0.4700
Epoch 4/100
113/113 [==============================] - 8s 66ms/step - loss: 0.6590 - accuracy: 0.7511 - val_loss: 0.6682 - val_accuracy: 0.7275
Epoch 5/100
113/113 [==============================] - 8s 67ms/step - loss: 0.5904 - accuracy: 0.7711 - val_loss: 1.8440 - val_accuracy: 0.6419
Epoch 6/100
113/113 [==============================] - 8s 70ms/step - loss: 0.5303 - accuracy: 0.7969 - val_loss: 0.6567 - val_accuracy: 0.7169
Epoch 7/100
113/113 [==============================] - 8s 69ms/step - loss: 0.4820 - accuracy: 0.8214 - val_loss: 0.5548 - val_accuracy: 0.7681
Epoch 8/100
113/113 [==============================] - 8s 67ms/step - loss: 0.4292 - accuracy: 0.8397 - val_loss: 0.8391 - val_accuracy: 0.7025
Epoch 9/100
113/113 [==============================] - 8s 68ms/step - loss: 0.3993 - accuracy: 0.8544 - val_loss: 0.4830 - val_accuracy: 0.7987
Epoch 10/100
113/113 [==============================] - 8s 67ms/step - loss: 0.3753 - accuracy: 0.8644 - val_loss: 0.4436 - val_accuracy: 0.8231
Epoch 11/100
113/113 [==============================] - 8s 68ms/step - loss: 0.3610 - accuracy: 0.8706 - val_loss: 0.4157 - val_accuracy: 0.8281
Epoch 12/100
113/113 [==============================] - 8s 67ms/step - loss: 0.3468 - accuracy: 0.8733 - val_loss: 0.4978 - val_accuracy: 0.8125
Epoch 13/100
113/113 [==============================] - 8s 67ms/step - loss: 0.3075 - accuracy: 0.8875 - val_loss: 1.1487 - val_accuracy: 0.7194
Epoch 14/100
113/113 [==============================] - 8s 68ms/step - loss: 0.2859 - accuracy: 0.8972 - val_loss: 0.5518 - val_accuracy: 0.8062
Epoch 15/100
113/113 [==============================] - 8s 67ms/step - loss: 0.2698 - accuracy: 0.9042 - val_loss: 0.5942 - val_accuracy: 0.7906
Epoch 16/100
113/113 [==============================] - 8s 68ms/step - loss: 0.2652 - accuracy: 0.9086 - val_loss: 1.4939 - val_accuracy: 0.6319
Epoch 17/100
113/113 [==============================] - 8s 68ms/step - loss: 0.2444 - accuracy: 0.9142 - val_loss: 0.5149 - val_accuracy: 0.8288
Epoch 18/100
113/113 [==============================] - 8s 68ms/step - loss: 0.2281 - accuracy: 0.9239 - val_loss: 0.4735 - val_accuracy: 0.8363
```

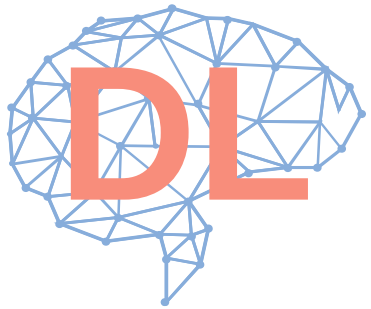

K-means
DBSCAN
Tree
SOM
...
클러스터링
PCA
PLS
LDA
LSA
...
차원축소
비지도학습
분류,
classification
Logistic
Naïve Bayes
SVM
Decision Tree
...
회귀,
regression
Linear Regression
Polynomial
Ridge
Lasso
ElesticNet
...
지도학습
AI
Genetic Algorithm
Q-learning
Deep Q-Networks
...
강화학습
앙상블방법
Stacking
Bagging
Boosting
심층신경망
(Deep Learning)
DNN
CNN
RNN
Transformers

심층 신경망(Deep Neural Network)

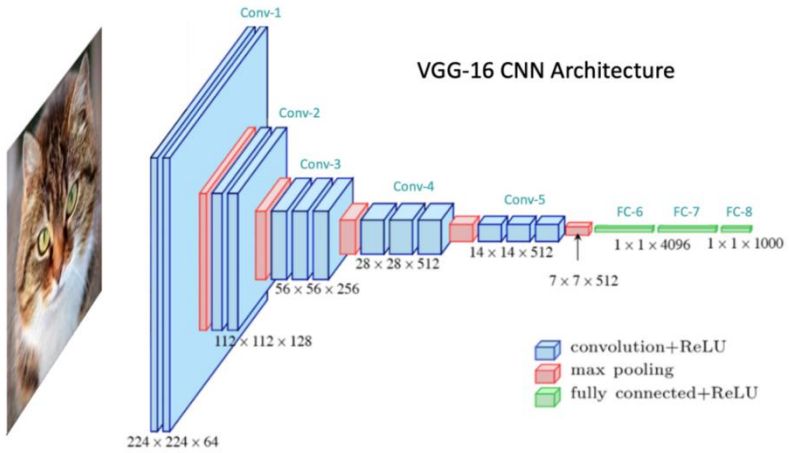


순환 신경망(Recurrent Neural Network)

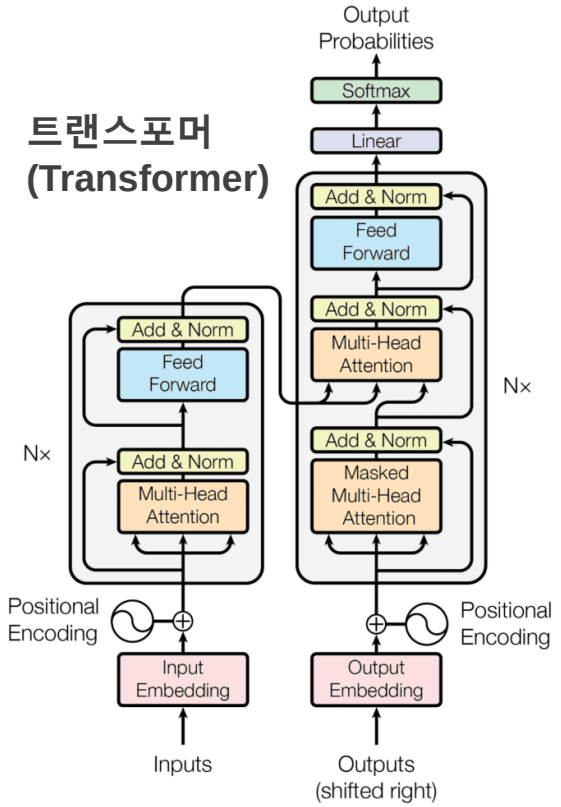




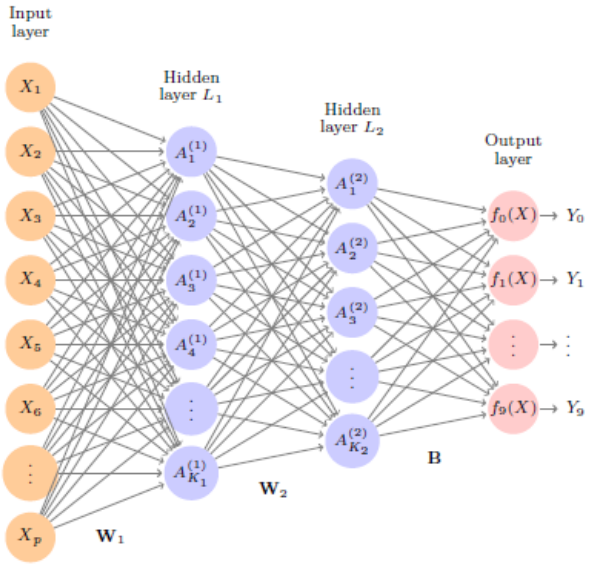
합성곱 신경망(Convolutional Neural Network)



트랜스포머 (Transformer)



심층 신경망 소스코드 예제 & Hyperparameter 튜닝의 중요성




```
model <- keras_model_sequential() %>%
  layer_dense(units = FLAGS$nodes1, activation = "relu",
              input_shape = ncol(ames_DN_train)) %>%
  layer_dropout(rate = FLAGS$dropout1) %>%
  layer_dense(units = FLAGS$nodes2, activation = "relu") %>%
  layer_dropout(rate = FLAGS$dropout2) %>%
  layer_dense(units = 1) %>%
  compile(
    loss = "mse",
    metrics = "mae",
    optimizer = FLAG$optimizer
  ) %>%
  fit(
    x = ames_DN_train,
    y = ames_train_targets,
    epochs = 500,
    batch_size = FLAGS$batch_size,
    validation_split = 0.2,
    callbacks = list(
      callback_early_stopping(patience = 25)
    ),
    verbose = 0
  )

runs <- tuning_run(file = "C:/Users/NET02/Desktop/",
                   flags = list(
                     nodes1 = c(64, 128, 256),
                     nodes2 = c(64, 128, 256),
                     dropout1 = c(0.2, 0.3, 0.4),
                     dropout2 = c(0.2, 0.3, 0.4),
                     optimizer = c("rmsprop", "adam"),
                     batch_size = c(16,32,64,128)
                   ),
                   sample=0.7
                   )
```

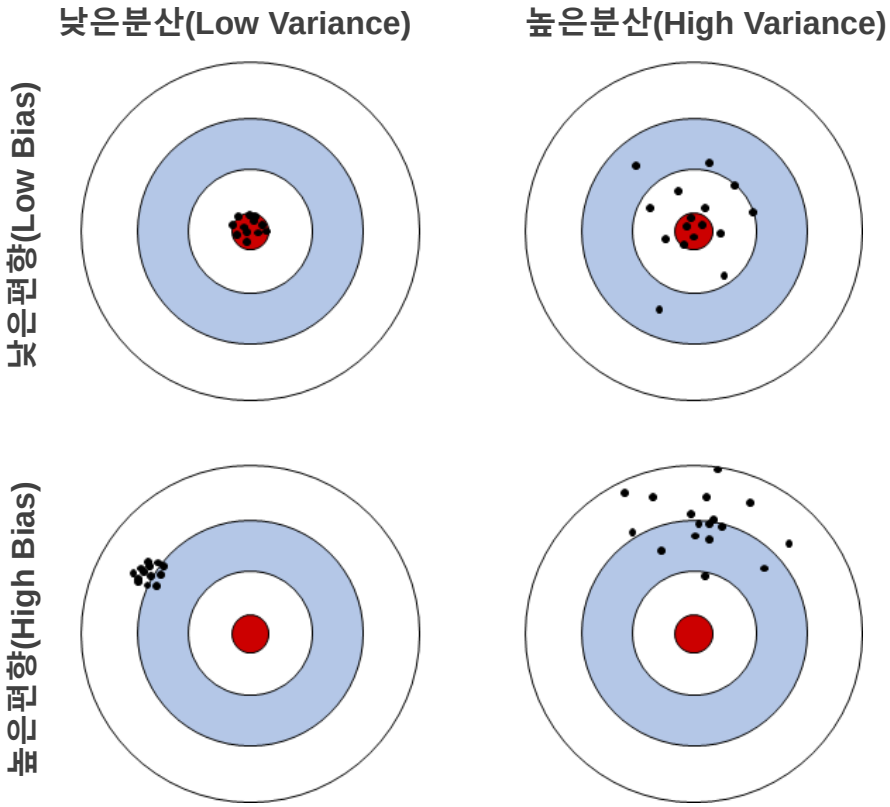
- 인공신경망의 하이퍼 파라미터 튜닝은 신경망의 성능의 향상시키기 위해 하이퍼 파라미터 값들을 최적화하는 과정이다. 하이퍼파라미터는 학습률(learning rate), 배치크기(batch size), 계층수(layer), 뉴런수(neurons), 드랍아웃 비율(dropout rate), 활성화 함수(activation function) 등 학습과정과 네트워크 구조를 결정하는 설정을 의미한다.

- 학습 중에 학습되는 모델 파라미터 값들과 달리, 하이퍼파라미터는 학습 시작 전에 설정되며, 학습 데이터에서 시험 데이터로 일반화하는 네트워크의 능력을 향상시키기 위한 신중한 선택이 필요하다.

- 효과적인 하이퍼 파라미터 튜닝은 그리드탐색(grid search, in this case 648), 랜덤 탐색, 또는 베이지안 최적화와 같은 보다 진보된 방법을 통해 다양한 구성을 체계적으로 테스트 및 평가하여 모델 손실을 (loss function)을 최소화하고 정확도를 극대화(accuracy)하는 최적의 하이퍼파라미터를 찾는 과정을 포함한다.

편향-분산 트레이드오프(Bias-Variance Tradeoff)

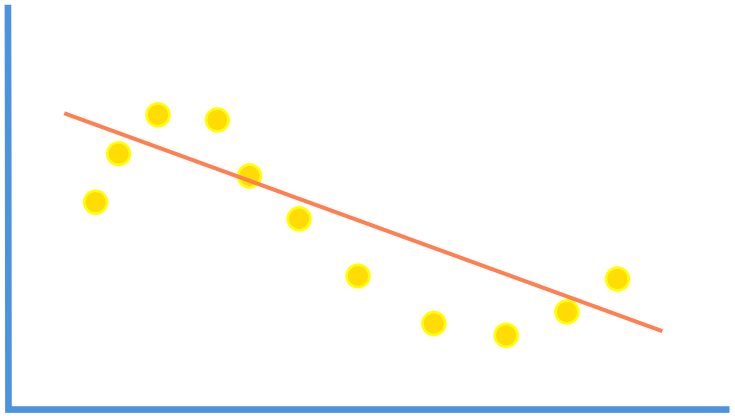
편향-분산(Bias-Variance) 트레이드오프는 머신러닝 모델의 성능을 최적화하는데 중요한 개념으로, 모델이 얼마나 일반화를 잘 할 수 있는지를 평가하는데 사용된다. 편향(Bias)은 모델이 학습데이터의 기본 패턴을 제대로 일반화하지 못할 때 발생하는 오류를 의미하며, 이는 주로 모델이 너무 단순할 때 발생한다. 반면에, 분산은 모델이 학습 데이터에 너무 민감하게 반응하여 발생하는 오류로, 모델이 너무 복잡할 때 발생한다. 따라서 편향을 줄이려고 하면 분산이 증가하고, 분산을 줄이려고 하면 편향이 증가하는 경향이 있다. 이 트레이드오프를 잘 활용하여 적절한 모델 복합성을 찾는 것이 기계학습 모델의 일반화를 최적화하는데 필수적인 요소이다.



편향-분산 트레이드오프(Bias-Variance Tradeoff)

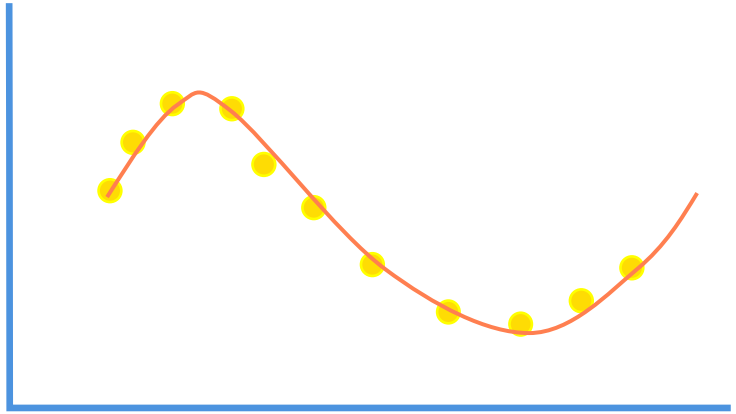
편향(Bias): 모델이 학습데이터의 기본 패턴을 제대로 일반화하지 못할 때 발생하는 오류로, 모델이 너무 단순할 때 발생

분산(Variance): 모델이 학습 데이터에 너무 민감하게 반응하여 발생하는 오류, 모델이 너무 복잡할 때 발생



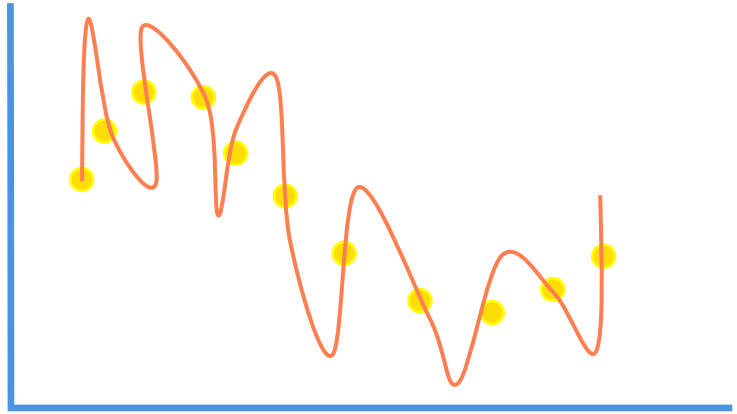
과소적합(Underfitting)
높은편향 – 낮은 분산

$$Y = \beta_1 X_1 + \in$$



최적화된 모델(Balanced)
중간편향 – 중간 분산

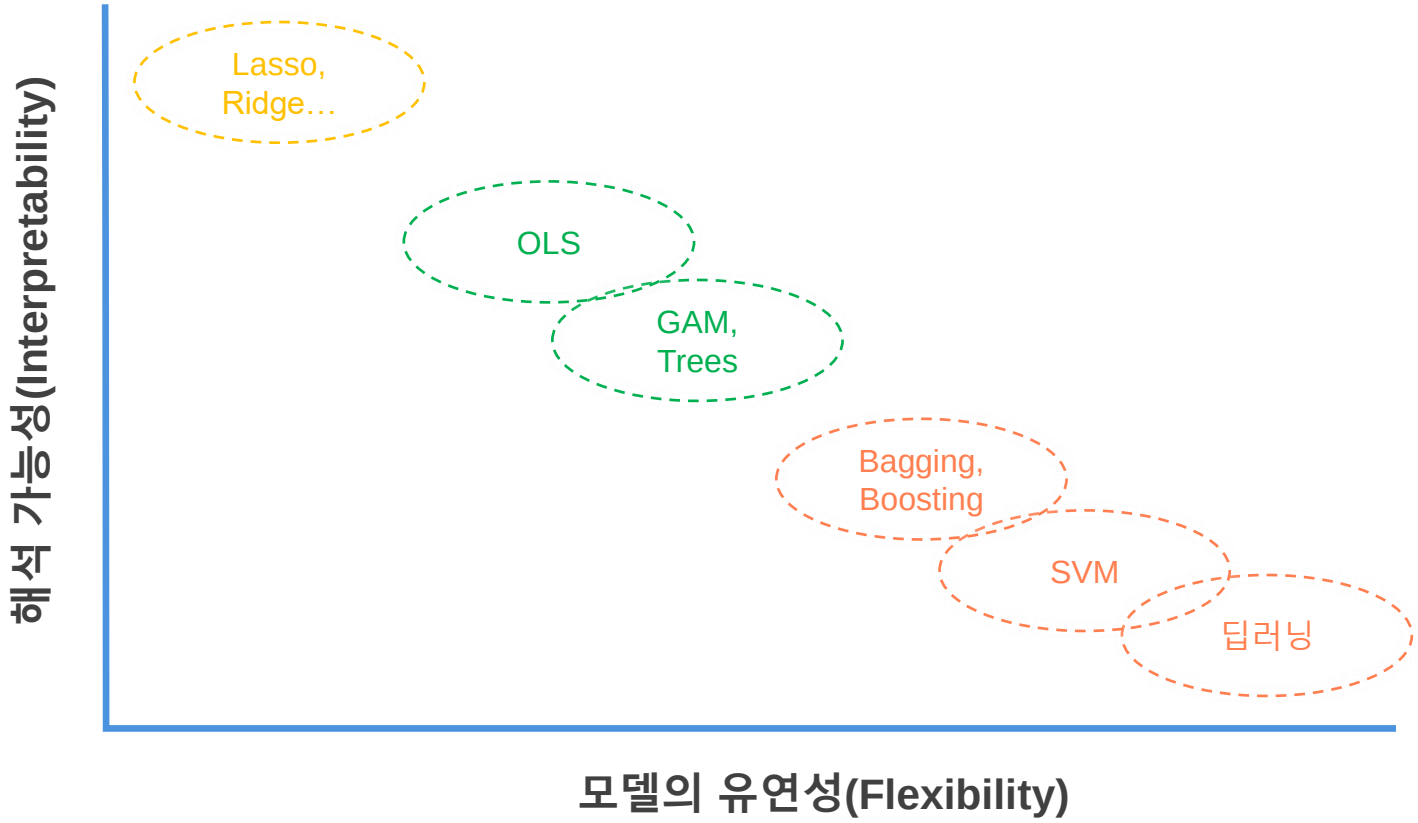
$$Y = \beta_1 X_1 + \beta_2 X_1^2 + \in$$



과대적합(Overfitting)
낮은편향 – 높은 분산

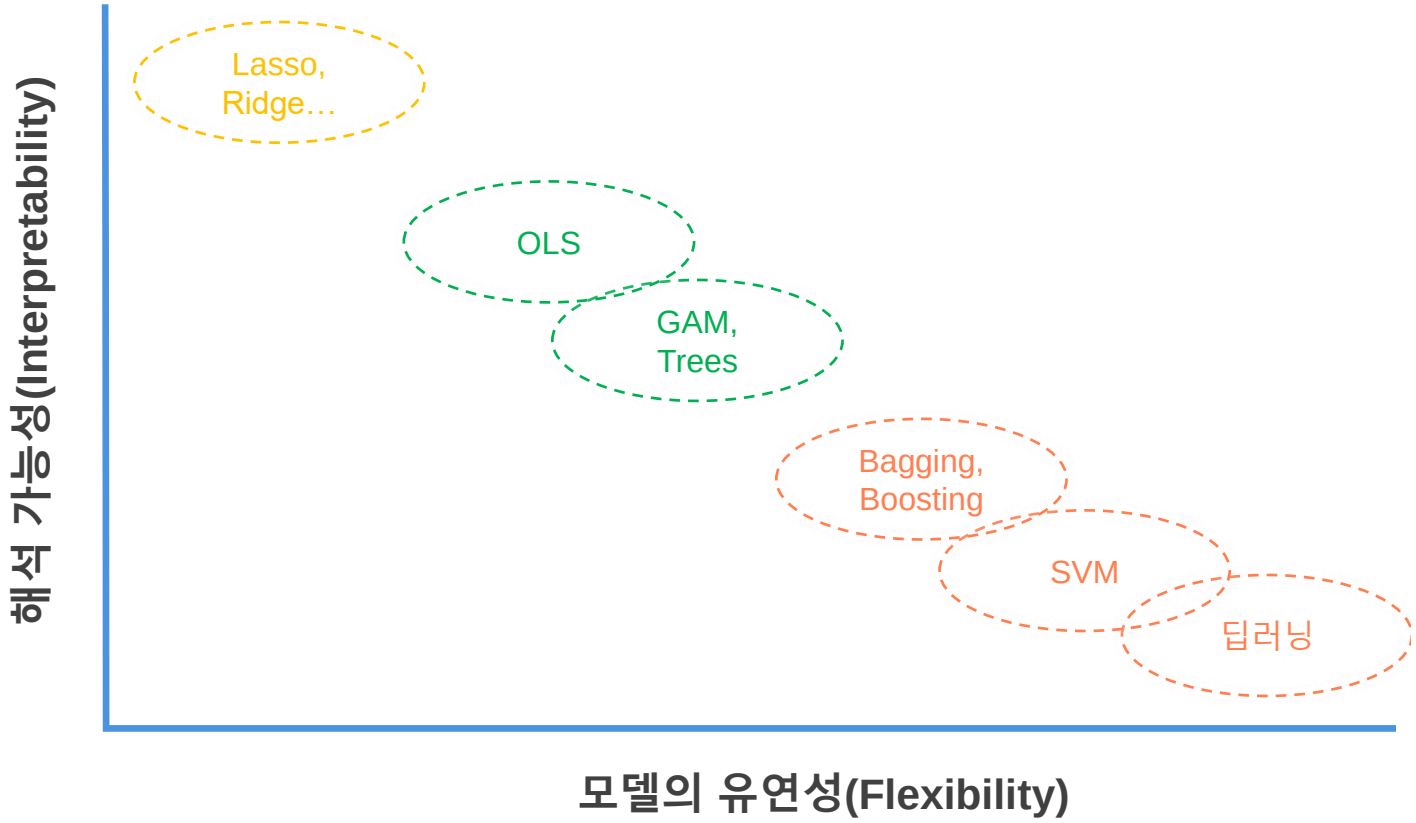
$$Y = \beta_1 X_1 + \beta_2 X_1^2 + + \beta_2 X_1^3 + \cdots + \in$$

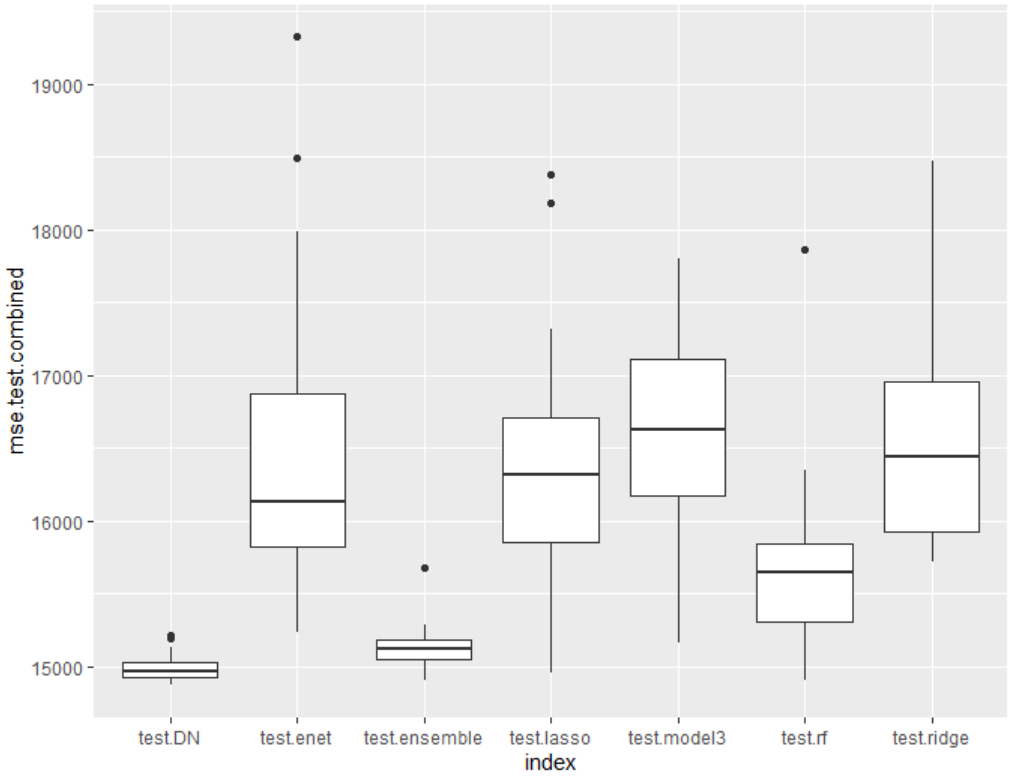
모델의 유연성과 해석가능성 트레이드오프



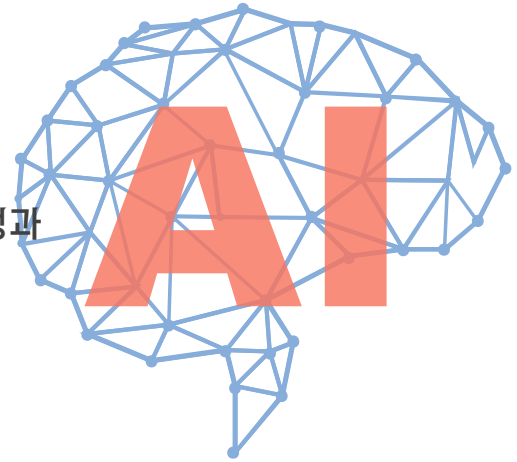
통계학적 학습 방법들 간의 유연성(flexibility)과 해석가능성(interpretability) 간의 상충관계(tradeoff)를 나타내는 것은 매우 중요하다. 일반적으로, 방법의 유연성이 증가할수록 해석가능성은 감소한다. 예를 들어, 선형회귀 모델과 같이 단순한 방법은 비교적 해석이 쉬우나 데이터의 복잡한 패턴을 포착하는 데는 한계가 있다. 반면에 랜덤포리스트나 인공신경망과 같은 복잡한 방법은 데이터의 복잡한 구조를 잘 학습할 수 있지만, 각각의 예측을 설명하는 데는 어려움이 따른다. 따라서 통계모델을 선택할 때는 유연성과 해석 가능성 간의 균형을 고려하여 선택하는 것이 중요하다.

모델의 유연성과 해석가능성 트레이드오프



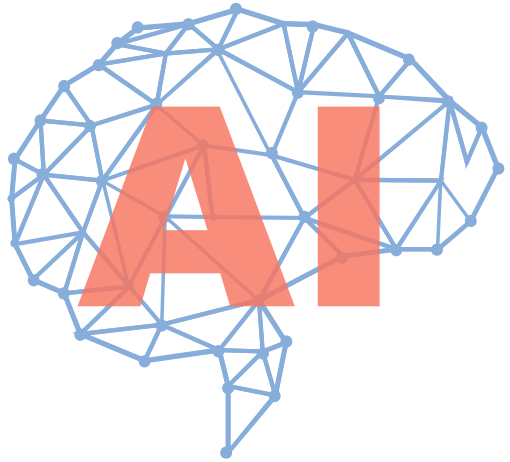


딥러닝 연구강의 요약



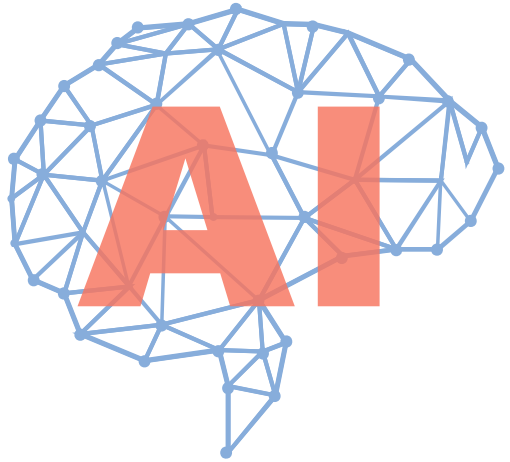
- 1) 딥러닝 AI 모델은 입력층, 여러 개의 은닉층, 출력층으로 구성되며, 데이터가 연속적인 필터를 통과하며 정제되는 증류 과정과 비교할 수 있다.
- 2) 딥러닝에서 학습은 모델의 가중치를 최적화하여 입력 값과 출력 값 사이에 올바른 매핑을 찾아내는 과정을 의미한다.
- 3) 딥러닝은 다른 AI학습 모델들과 비교하여 튜닝해야할 파라미터 값들이 월등하게 많다.
- 4) 딥러닝 학습을 위하여 데이터셋은 훈련셋, 검증셋, 시험셋으로 분할한다. 훈련셋은 실제 학습할 때, 검증셋은 모델의 성능을 모니터링 할 때, 시험셋은 모델의 일반화 능력을 평가할 때 사용한다.
- 5) 딥러닝에서 데이터는 입력층에서 출력층으로 순방향(feed-forward) 이동하며 이 때 모델은 데이터의 패턴을 학습한다. 손실함수(loss function)를 이용하여 모델의 성능을 모니터링하며, 역전파(backpropagation) 알고리즘과 경사하강법(gradient descent)기법을 이용하여 모델의 가중치를 최적화 할 수 있다.
- 딥러닝은 크게 심층 신경망, 합성곱 신경망, 순환 신경망, 트랜스포머로 대표되어지며, 앞으로 진행할 연구사업에서는 이 모든 모델들을 독립적으로 혹은 함께 조합하여 사용하게 될 것으로 판단된다.
- 하이퍼파라미터 튜닝은 모델의 구조를 결정짓는 여러가지 요소들(drop out, regularization, 레이어 수, 뉴런수 등등)을 조정하여 최상의 모델을 찾아내는 과정을 의미하며 많은 시간, 컴퓨팅 파워, 노력을 요하는 과정이다.
- 인공지능 학습에서 모델의 편향과 분산 사이에는 상쇄 관계가 존재하며, 모델의 유연성(복잡성)이 증가할수록 모델의 해석가능성은 감소하는 경향이 있다. 따라서 AI 모델을 훈련모델에 적용하는데 있어서 소요군이 기대하는 설명가능성 수준이 어느정도 일지 미리 고민해봐야 한다고 생각한다.

논의 사항



- 3)과 4)를 바탕으로 생각해 봤을 때, 우리가 진행하고자 하는 연구사업에서 AI 모델의 신뢰성을 높이기 위해 일정 양 이상의 데이터 확보는 필수라고 판단된다.
- 필요한 데이터 식별과 데이터 확보방안 수립을 위해서는 구체적인 목표 수립이 선행돼야 한다고 생각한다. 예를들면:
  - ✓ 어떤 AI 모델을 사용할 것인지?
  - ✓ 학습에 필요한 입력 데이터는 뭘 사용할지?
  - ✓ 언어모델(LLM)을 사용할 계획이라면, 확보 가능한 텍스트 데이터 양은 얼마나 되는지?
  - ✓ 사전학습모델(pre-trained model)을 사용하여 학습을 진행 할 계획이라면 어느 정도의 데이터가 필요할지?
  - ✓ 사전학습 모델과 군도메인의 상관관계는 어느 정도 고려해야 할지?
  - ✓ 모델의 신뢰성 확보를 위해서 각각의 AI모델 마다 최소한으로 필요한 데이터의 양은 얼마나 되는지?
- 훈련모델 학습에 사용할 AI모델에 대한 구체적인 설계 또한 생각해 봐야함(e.g., AI-CGF는 심층신경망, 강화학습, 유전자 알고리즘을 사용함; DN-CGF는 심층신경망, RNN, LSTM, 강화학습을 조합하여 모델을 설계함)

다음 연구강의 예고- AI-CGF & DL-CGF?



- AI-CGF 전반적인 소개
- AI-CGF에 사용된 방법론
  - ✓ 심층신경망(Deep Neural Networks)
  - ✓ 유전자 알고리즘(Genetic algorithm)
  - ✓ 강화학습(Reinforcement learning)
- DL-CGF 전반적인 소개
- DL-CGF 실험설계 및 학습환경 구성
- DL-CGF에 사용된 방법론
  - ✓ 심층신경망(Deep Neural Networks)
  - ✓ 장단기 기억망(Long Short-Term Memory)
  - ✓ 강화학습(Reinforcement learning)

Thank You

딥러닝이란 무엇인가?