IDG

IT WORLD CIO

IDG Tech Report

딥 러닝과 머신러닝의 차이점 이해하기

딥 러닝(Deep Learning)과 머신러닝(Machine Learning)은 모두 데이터에서 패턴을 찾는다는 점에서 같으며, 딥 러닝을 머신러닝의 한 종류라고 해도 무방하다. 하지만 사용하는 방법과 용도에서 큰 차이가 있다. 둘 다 회귀와 분류 문제를 처리할 수 있지만, 딥 러닝은 언어 번역이나 객체 인식과 같은 일부 영역에서 더 나은 결과를 산출한다. 딥 러닝과 머신러닝의 개념과 방법론을 살펴보고, 이 둘의 차이점에 대해 알아보자.



무단 전재
재배포 금지

본 PDF 문서는 IDG Korea의 자산으로, 저작권법의 보호를 받습니다.
IDG Korea의 허락 없이 PDF 문서를 온라인 사이트 등에 무단 게재, 전재하거나 유포할 수 없습니다.

딥 러닝과 머신러닝의 차이점 이해하기

Martin Heller | InfoWorld

머신러닝(Machine Learning)과 딥 러닝(Deep Learning)은 모두 데이터에서 패턴을 찾지만 사용하는 방법에는 큰 차이가 있다. 둘 다 인공지능(Artificial Intelligence)의 한 형태로, 딥 러닝이 머신러닝의 한 종류라고 해도 무방하다. 또한 머신러닝과 딥 러닝 모두 학습 및 테스트 데이터와 모델로 시작해 최적화 프로세스를 거쳐 데이터에 맞는 최적의 모델을 위한 가중치를 찾으며, 숫자(회귀)와 비숫자(분류) 문제를 처리한다. 딥 러닝 모델은 객체 인식, 언어 번역과 같은 일부 영역에서 머신러닝에 비해 대체로 더 나은 결과를 산출한다. 하지만 딥 러닝은 학습과 대규모 배포를 위한 하드웨어 가속기가 필수적이다.

지도 학습과 비지도 학습으로 나뉘는 머신러닝

머신러닝 알고리즘은 많은 경우 지도(Supervised, 학습 데이터에 정답 태그가 지정됨) 학습과 비지도(Unsupervised, 학습 데이터에 어떠한 레이블도 볼 수 없음) 학습으로 나뉜다.

지도 학습은 다시 분류(Classification)와 회귀(Regression)로 나뉜다. 분류는 모기지 이자 연체 가능성과 같은 숫자가 아닌 답을 예측하는 것이고, 회귀는 맨해튼 상점에서 다음 달에 팔릴 제품의 수와 같은 숫자의 답을 예측하는 것이다.

비지도 학습은 클러스터링(Clustering), 연관(Association), 차원 축소(Dimensionality Reduction)로 세분화한다. 클러스터링은 러닝화, 워킹화, 정장 구두와 같은 비슷한 객체의 그룹을 찾는 것이며, 연관은 커피, 크림, 우유와 같이 일반적인 객체의 연계를 찾는 것이다. 차원 축소는 분석 대상의 수를 줄이는 것으로, 투영(Projection), 특성 선택(Feature Selection), 특성 추출(Feature Extraction) 등이 있다.

분류 문제는 2개 이상 클래스에서 선택을 묻는 지도 학습으로, 일반적으로 각 클래스의 확률을 제공한다. 분류의 가장 일반적인 알고리즘은 ▲나이브 베이즈(Naive Bayes) ▲의사결정 트리(Decision Tree) ▲로지스틱 회귀(Logistic Regression) ▲K-최근접 이웃(K-Nearest Neighbors, KNN) ▲서포트 벡터 머신(Support Vector Machine, SVM)이다. 또한 ▲랜덤 포레스트(Random Forest)와 같은 조합 방법 ▲기타 배깅(Bagging) 방법 ▲에이다부스트(AdaBoost) ▲XG부스트(XGBoost)와 같은 부스팅 방법을 사용할 수도 있다.

회귀 문제는 모델에 숫자를 예측하도록 요청하는 지도 학습이다. 가장 간단하고 빠른 알고리

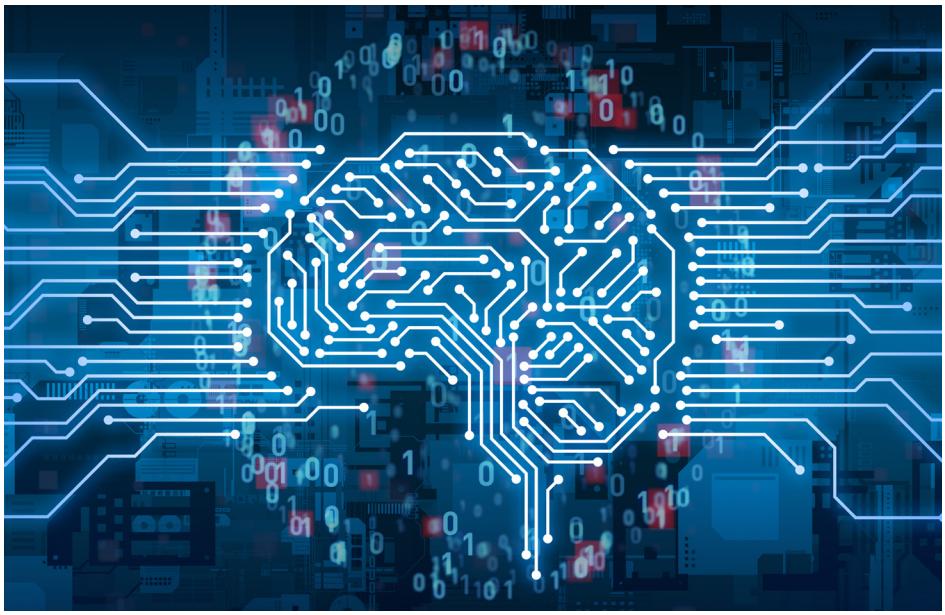
즘은 선형(linear) 회귀, 즉 최소제곱법(Least Squares)이지만 이것으로는 충분하지 않은 경우가 많으므로 다른 방법도 살펴봐야 한다.

다른 일반적인 머신러닝 회귀 알고리즘에는 신경망을 제외하고 ▲나이브 베이즈 ▲의사결정 트리 ▲KNN ▲학습 벡터 양자화(Learning Vector Quantization, LVQ) ▲LARS 라소(LARS Lasso) ▲엘라스틱 넷(Elastic Net) ▲랜덤 포레스트 ▲에이다부스트 ▲XG부스트가 있다. 이와 같이 회귀와 분류 머신러닝 알고리즘 중에는 일부 겹치는 부분이 있다.

클러스터링 문제는 비슷한 유형의 데이터 그룹을 찾도록 요청해, 주로 데이터의 경향성을 파악하는 비지도 학습이다. 가장 많이 사용되는 알고리즘은 ▲K-평균 클러스터링(K-Means Clustering)이며, 그 외에 ▲평균점 이동 클러스터링(Mean-Shift Clustering) ▲DBSCAN(Density-Based Spatial Clustering of Applications with Noise) ▲가우시안 혼합 모델(Gaussian Mixture Models, GMM) ▲계층 병합 클러스터링(Hierarchical Agglomerative Clustering, HAC)이 있다.

차원 축소(Dimensionality reduction)는 결과에 영향이 전혀 혹은 거의 없는 변수를 삭제하거나 병합해 고차원 데이터를 저차원 데이터로 변환하는 비지도 학습이다. 이는 분류 또는 회귀와 같은 지도 학습과 병행해 사용하는 경우가 많다. 차원 축소 알고리즘은 ▲누락된 값이 많은 변수 제거 ▲변동성이 낮은 변수 제거 ▲의사결정 트리 ▲랜덤 포레스트 ▲연관성이 높은 변수 제거 또는 결합 ▲후진 소거법(Backward Feature Elimination) ▲전진 선택법(Forward Feature Selection) ▲요인 분석(Factor Analysis) ▲주성분 분석(Principal Component Analysis, PCA)을 포함한다.

지도 학습 알고리즘은 학습과 평가를 거쳐 매개변수 가중치를 최적화(Optimization)해 데이터의 기반 진실에 가장 일치하는 값을 찾음으로써 모델이 된다. 알고리즘은 자체 옵티마이저(optimizers)를 위해 최속하강(steepest descent) 변형에 의존하는 경우가 많다. 예를 들어 확률적 경사 하강(Stochastic Gradient Descent, SGD)은 본질적으로 무작위로 선택한 시작 포



인트에서 여러 번 최속하강법을 수행하는 것이다.

SGD를 개선하는 일반적인 방법은 모멘텀을 기반으로 경사의 방향을 수정하는 요인 추가 또는 한 패스에서 다음 패스로 이어지는 데이터의 진행(에포크(epoch) 또는 배치(batch)라고 함)을 기반으로 하는 학습률(learning rates) 조절이다.

머신러닝 사용하기 01›› 데이터 정제

원본 데이터에는 클린 데이터(clean data)가 존재하지 않는다. 머신러닝에 사용하려면 데이터를 적극적으로 정제해야 한다. 예를 들어 다음과 같은 사항을 진행해야 한다.

1. 데이터를 관찰해 누락된 데이터가 많은 모든 열(columns)을 배제한다.
2. 데이터를 다시 한번 살펴보면서 예측에 사용할 열을 선택한다(특성 선택). 반복할 때는 매번 다르게 하는 것이 좋다.
3. 남은 열에서 여전히 누락된 데이터가 있는 행(rows)을 제거한다.
4. 오탈자를 수정하고 동일한 답을 병합한다. 예를 들어 U.S., US, USA, America는 하나의 범주로 병합해야 한다.
5. 범위를 벗어난 데이터가 있는 행을 제거한다. 예를 들어 뉴욕 시 내의 택시 운행을 분석 중이라면 뉴욕 시 경계선 외부의 승하차에 대한 위도와 경도는 필터링으로 제거해야 한다.

이 외에도 많은 일을 해야 하지만, 일은 수집하는 데이터에 따라 다르다. 지루한 작업이지만 머신러닝 파이프라인에 데이터 정제 단계를 두면 자유롭게 수정, 반복할 수 있다.

머신러닝 사용하기 02›› 데이터 인코딩과 정규화

머신러닝 분류에서 범주형 데이터(categorical data)를 사용하려면 텍스트 레이블을 다른 양식으로 인코딩(encoding)해야 한다. 일반적인 인코딩은 두 가지다. 하나는 각 텍스트 레이블 값을 숫자로 대체하는 레이블 인코딩(label encoding)이며, 다른 하나는 각 텍스트 레이블 값이 이진 값(1 또는 0)이 있는 열로 변환되는 원 핫 인코딩(one-hot encoding)이다.

대부분의 머신러닝 프레임워크에는 변환을 위한 함수가 있는데, 일반적으로 원 핫 인코딩을 선호한다. 레이블 인코딩의 경우, 종종 머신러닝 알고리즘이 혼란에 빠져 인코딩된 열이 순서가 정렬된 목록이 되어야 한다고 착각하는 경우가 있기 때문이다.

머신러닝 회귀에서 숫자 데이터를 사용하려면 일반적으로 데이터를 정규화(Normalization)해야 한다. 데이터를 일정한 규칙에 따라 변환하는 정규화를 하지 않으면 범위가 큰 숫자가 특성 벡터 간의 유클리드 거리(Euclidian Distance)를 좌우하면서 그 영향을 확대해 반대급부로 다른 필드의 영향을 축소시키고, 이로 인해 최속하강 최적화에서 수렴이 어려워질 수 있다.

머신러닝을 위해 데이터를 정규화 및 표준화하는 방법은 최소 최대 정규화(Min-Max Normalization), 평균 정규화(Mean Normalization), 표준화(Standardization), 단위 길이로 스케일링(scaling to unit length)을 포함해 여러 가지다. 이 프로세스를 보통 특성 스케일링(Feature Scaling)이라고 한다.

머신러닝 사용하기 03›› 특성 엔지니어링

특성은 관찰 대상 현상의 개별적으로 측정 가능한 속성 또는 특성이다. '특성(Feature)'이라는 개념은 선형 회귀(Linear Regression)와 같은 통계 기법에서의 설명 변수(explanatory variable)와 동일하다. 특성 벡터는 단일 행의 모든 특성을 하나의 숫자 벡터로 결합한다.

특성 선택 기술에는 문제를 설명하는 독립 변수(independent variables)의 최소 집합을 선택하는 것이 있다. 두 변수의 연관성이 높은 경우, 두 변수를 하나의 특성으로 결합하거나 하나를 제거해야 한다. 주 성분 분석을 수행해 연관된 변수를 선형적으로 연관되지 않은 변수로 변환하는 경우도 있다.

새로운 특성을 구축하거나 특성 벡터의 차원을 줄이기 위해 사용하는 변환 중에는 간단한 것도 있다. 예를 들어 사망 연도에서 출생 연도를 빼면 사망시 연령을 구할 수 있는데, 이는 수명과 사망 분석을 위한 주요 독립 변수다. 하지만 특성 구축이 이처럼 명확하지 않은 경우도 있다.

머신러닝 사용하기 04›› 데이터 분할(Splitting data)

지도 머신러닝의 일반적인 방법은 데이터 집합을 학습, 검증, 테스트를 위한 하위 집합으로 분할하는 것이다. 80%의 데이터를 학습 데이터 집합에, 그리고 검증과 테스트 데이터 집합에 각각 나머지 10%를 할당하는 방법이 있다(정확한 분할 비중은 선호도의 문제다). 학습의 대부분은 학습 데이터 집합을 사용해 이뤄지며 예측은 매 에포크의 끝에 검증 데이터 집합을 대상으로 수행한다.

검증 데이터 집합의 오류를 사용해 정지 기준을 식별하거나 초매개변수(hyperparameter)를 조정할 수 있다. 가장 중요한 점은 검증 데이터 집합의 오류가 모델이 학습 데이터에 적합한지를 파악하는 데 도움이 된다는 것이다.

테스트 데이터 집합을 대상으로 한 예측(Prediction)은 일반적으로 최종 모델에서 수행한다. 테스트 데이터 집합이 학습에 사용된 적이 없는 경우, 이를 홀드아웃(holdout) 데이터 집합이라고도 한다.

데이터 분할 체계는 그 외에도 여러 가지가 있다. 일반적인 기법 가운데 하나는 교차 검증(cross validation)으로, 전체 데이터 집합을 학습 데이터 집합과 검증 데이터 집합으로 반복적으로 분할하는 방법이다. 각 에포크의 끝에서 데이터를 뒤섞은 다음 다시 분할한다.

파이썬(Python)에서 스파크 MLlib과 사이킷-런(Scikit-learn)은 머신러닝 라이브러리로 탁월한 선택이다. R의 머신러닝 패키지 옵션으로는 CARAT, 랜덤 포레스트, e1071, 컨랩(KernLab)이 있다. 자바에서는 자바-ML, 래피드마이너(RapidMiner), 웨카(Weka)가 좋은 선택이다.

뉴런 간 연결을 강화하는 딥 러닝

딥 러닝은 학습되는 모델에서 입력과 출력 사이에 숨은 계층이 두 개 이상 있는 머신러닝의 한 형태다. 대체로 딥 러닝은 심층신경망(Deep Neural Networks, DNN) 사용을 의미한다. 그러나 신경망 외에 다른 종류의 숨은 계층을 사용해 딥 러닝을 구현하는 알고리즘도 있다.



'인공' 신경망 개념은 1940년대까지 거슬러 올라간다. 기본적인 개념은 상호 연결된 한계 스위치(threshold switch)를 통해 만든 인공신경망(Artificial Neurons Network, ANN)으로 동물의 두뇌와 신경 시스템(망막 포함)과 같은 방식으로 패턴 인식을 학습할 수 있다는 것이다.

딥 러닝의 학습은 기본적으로 학습하는 동안 두 뉴런이 모두 동시에 활성일 때 두 뉴런 사이의 연결을 강화함으로써 이뤄진다. 현대의 신경망 소프트웨어에서 이는 대체로 오차역전파(BackproPagation)라 부르는 규칙을 사용해 뉴런 간의 연결 가중값을 늘리는 것을 의미한다.

그렇다면 뉴런은 어떻게 모델링할까? 각 뉴런에는 연결된 뉴런의 출력을 변환하는 전파 함수(Propagation Function)가 있고, 이 함수에는 가중 합계가 있는 경우가 많다. 전파 함수의 출력은 활성화 함수(Activation Function)로 전달된다. 활성화 함수는 입력이 한계 값을 초과할 때 실행한다.

1940년대와 50년대에 인공 뉴런은 단계 활성화 함수(Step Activation Function)를 사용했으며 퍼셉트론(Perceptron)이라고 불렀다. 현대의 신경망은 퍼셉트론을 사용한다고 할 수도 있지만, 사실 로지스틱(Logistic) 또는 시그모이드 함수(Sigmoid Function), 쌍곡선 탄젠트(Hyperbolic Tangent), 정류된 선형 유닛(Rectified Linear Unit, ReLU)과 같은 평활 활성화 함수(Smooth Activation Functions)를 사용한다. ReLU는 일반적으로 빠른 수렴을 위한 최선의 선택이지만 학습 도중 학습률을 너무 높게 설정하는 경우 뉴런이 '죽는' 문제가 발생한다.

활성화 함수의 출력은 추가 형상화를 위해 출력 함수로 전달할 수 있다. 출력 함수는 항등함수(Identity Function)인 경우가 많다. 즉, 활성화 함수의 출력이 하방 연결된 뉴런으로 전달된다.

머신러닝과 같은 학습 방법을 사용하는 딥 러닝

뉴런을 간략하나마 이해했다면 이제 일반적인 신경망 토폴로지(Neural Network Topolo-

gies)에 대해 알아볼 차례다. 피드포워드 망(feed-forward network)에서 뉴런은 개별적인 계층(입력 계층 1개, 숨은 처리 계층 n개, 출력 계층 1개)으로 구성된다. 각 계층의 출력은 다음 계층으로만 이동한다.

지름길 연결(shortcut connections)이 있는 피드포워드 망에서 일부 연결은 하나 이상의 중간 계층을 뛰어넘을 수 있다. 순환 신경망(Recurrent Neural Networks, RNN)의 뉴런은 직접적으로, 또는 다음 계층을 통해 간접적으로 스스로에게 영향을 미칠 수 있다.

신경망의 지도 학습은 다른 머신러닝과 같은 방법으로 수행된다. 네트워크에 학습 데이터 그룹을 제시하고 네트워크 출력을 원하는 출력과 비교하고 오류 벡터를 생성해 이를 기반으로 네트워크에 수정을 적용한다. 수정을 적용하기 전에 함께 실행하는 학습 데이터 배치를 에포크라고 한다. 세부적으로 설명하면, 역전파는 모델의 가중치 및 편향성과 관련해 오류(또는 비용) 함수의 경사로 오류를 최소화하는 수정 방향을 찾는다. 수정 적용을 제어하는 두 가지 요소는 최적화 알고리즘(optimization algorithm)과 학습률 변수다. 수렴을 보장하고 ReLU 뉴런의 죽음을 방지하기 위해 일반적으로 학습률 변수는 작아야 한다.

신경망의 옵티마이저는 일반적으로 경사 하강 알고리즘을 사용해 역전파를 견인한다. 예를 들어, 임의로 선택한 미니 배치 최적화(확률적 경사 하강)와 국소 최소값에 갇히는 문제를 피하고, 경사에 탄성 수정치를 적용하는 메카니즘을 함께 사용하는 경우도 많다. 아다그라드(AdaGrad), RMSProp(Root Mean Sqaure Propagation), Adam과 같은 일부 최적화 알고리즘은 경사 기록을 확인해 모델 매개변수의 학습률을 조정하기도 한다.

모든 머신러닝이 그렇듯이 신경망의 예측은 별도의 검증 데이터 집합을 사용해 확인해야 한다. 이 과정을 수행하지 않으면 학습을 통해 일반적인 예측 도구가 되지 않고 입력만 기업하는 신경망이 될 위험이 있다.

수많은 계층의 딥 러닝 알고리즘, 연산에 많은 시간 소요

실제로 심층 신경망에는 10개 이상의 숨은 계층이 있을 수 있다. 신경망 토폴로지는 단순할 수도, 상당히 복잡할 수도 있다. 네트워크에 계층이 많을수록 더 많은 특성을 인식할 수 있다. 하지만 계층이 많을수록 연산하는 데 많은 시간이 걸리고 학습도 더 어려워진다.

컨볼루션 신경망(Convolutional neural networks, CNN)은 머신 비전(machine vision)에 자주 사용한다. CNN은 일반적으로 컨볼루션, 풀링(pooling), ReLU, 완전 연결 및 손실 계층을 사용해 시각령(visual cortex)을 시뮬레이션한다. 컨볼루션 계층은 기본적으로 다수의 작은 중첩 영역의 적분(integrals)을 가져온다. 풀링 계층은 비선형(non-linear) 다운샘플링(down-sampling)을 수행한다. ReLU 계층은 비포화(Non-saturating) 활성화 함수 f(x) = max(0,x)를 적용한다. 완전 연결 계층에서 뉴런은 이전 계층의 모든 활성화와 연결된다. 손실 계층의 경우, 분류에는 소프트맥스(Softmax) 또는 교차 엔트로피(Cross Entropy) 손실 함수를 사용하고 회귀에는 유클리드 손실 함수를 사용해, 예측한 레이블과 실제 레이블 사이의 편차를 파악해 조정하는 방법을 계산한다.

RNN은 장단기 메모리(Long Short-Term Memory, LSTM) 신경망, 어텐션 기반 신경망(at-

tention-based neural networks)과 마찬가지로, 자연어 처리(Natural Language Processing, NLP)와 기타 시퀀스 처리에 자주 사용된다. 피드포워드 신경망에서 정보는 입력에서 숨은 계층을 통과해 출력으로 흐른다. 이로 인해 네트워크는 한 번에 하나의 상태만 처리하는 것으로 제한된다. RNN에서는 정보가 루프를 통해 순환하며 신경망이 최근의 이전 출력을 기억할 수 있다. 이를 통해 시퀀스와 시계열의 분석이 가능하다. RNN에는 경사 고정으로 쉽게 수정할 수 있는 경사 폭발(exploding gradients)과 쉽게 수정할 수 없는 경사 소실(vanishing gradients), 두 가지 일반적인 문제가 있다.

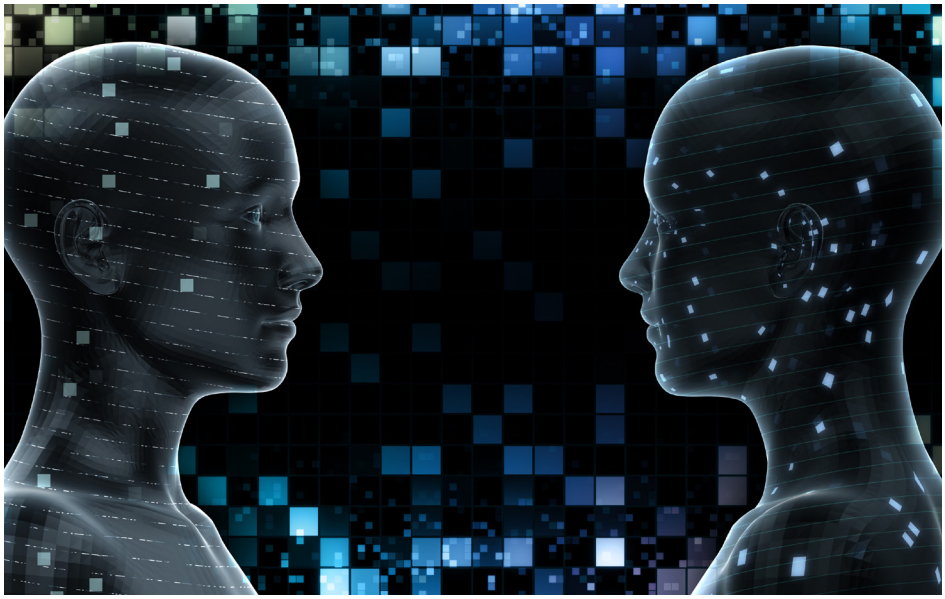
LSTM에서 신경망은 가중치를 변경하는 방법으로 이전 정보를 잊거나(gating) 기억할 수 있다. 이는 사실상 LSTM에 장기 메모리와 단기 메모리를 모두 부여하며 경사 소실 문제를 해결한다. LSTM은 수백 개의 과거 입력 시퀀스를 처리할 수 있다.

어텐션 모듈은 입력 벡터에 가중치를 적용하는 일반화된 게이트다. 계층적 신경 어텐션 인코더는 여러 개의 어텐션 모듈 계층을 사용해 수많은 과거 입력을 처리한다.

랜덤 의사결정 포레스트(Random Decision Forests, RDF)는 신경망이 아닌 또 다른 종류의 딥 러닝 알고리즘으로, 폭넓은 분류와 회귀 문제에 유용하다. RDF는 많은 계층을 구성하지만 뉴런 대신 의사결정 트리에서 RDF를 구축하고 개별 트리 예측의 통계적 평균을 출력한다(분류의 경우 최빈값, 회귀의 경우 평균). RDF의 임의성 측면은 개별 트리에 부트스트랩 총계치(일명, 배깅(baggin))를 사용하고 해당 트리 특성의 하위 집합을 가져온다.

XG부스트(eXtreme Gradient Boosting) 역시 DNN이 아니며, 다수의 머신러닝 과제에 대해 생성된 최신 결과가 있는 확장 가능한 엔드투엔드 트리 부스팅(tree boosting) 시스템이다. 배깅과 부스팅은 함께 언급되는 경우가 많은데, 차이점은 무작위 트리의 조합을 생성하는 대신 경사 트리 부스팅이 하나의 의사결정 또는 회귀 트리로 시작해 이를 최적화한 다음 첫 번째 트리의 잔여물로 다음 트리를 만든다는 점이다.

가장 우수한 파이썬 딥 러닝 프레임워크로는 텐서플로우(TensorFlow), 케라스(Keras), 파이



토치(PyTorch), MX넷(MXNet)이 있다. 딥 러닝4j(Deeplearning4j)는 최고의 자바 딥 러닝 프레임워크 가운데 하나다. ONNX와 텐서RT(TensorRT)는 딥 러닝 모델을 위한 런타임이다.

NLP와 이미지 분류에 능한 딥 러닝, 나날이 용도 확장

대체로 일반 머신러닝 알고리즘은 딥 러닝 알고리즘에 비해 학습 및 예측 속도가 훨씬 더 빠르다. 하나 이상의 CPU로 일반 모델을 충분히 학습시킬 수 있다. 딥 러닝 모델은 학습과 대규모 배포를 위해 GPU, TPU(Tensor Processing Unit) 또는 FPGA(Field Programmable Gate Array)와 같은 하드웨어 가속기가 필요한 경우가 많다. 하드웨어 가속기가 없으면 학습에 수개월이 걸릴 수 있다. 일부 일반 머신러닝 알고리즘은 많은 부문에서 "충분히 좋은" 모델을 생성한다.

일반적으로 딥 러닝이 사용되는 영역 중 하나는 NLP로, 언어 번역, 자동 요약, 대용어 해소(Co-reference Resolution), 담화 분석(Discourse Analysis), 형태학적 분할(morphological segmentation), 명명된 개체 인식, 자연어 생성(natural language generation), 자연어 이해(natural language understanding), 품사 태깅(part-of-speech tagging), 감정 분석(sentiment analysis), 음성 인식(speech recognition)을 포괄한다.

딥 러닝의 또 다른 주된 영역은 이미지 분류(Image Classification)다. 이미지 분류에는 로컬화 이미지 분류(Image classification with localization)를 포함해 객체 탐지, 객체 분할, 이미지 스타일 전이(Image style transfer), 이미지 색상화(Image colorization), 이미지 재구성(Image reconstruction), 이미지 초고해상화(Image super-resolution), 이미지 합성(Image synthesis) 등이 있다.

아울러 딥 러닝은 제약 업체에서 새로운 약물을 설계하기 위한 분자 상호작용을 예측하고, 원자 구성 입자(subatomic particles)를 찾고, 현미경 이미지를 자동으로 분석해 인간 두뇌의 3차원 지도를 생성하는 데 사용하고 있다. ITWORLD