Big Data 분석을 위한 Machine Learning

이재구, 이태훈, 윤성로
서울대학교

요 약

본고는 빅데이터 시대에 새로운 가치를 창출할 수 있는 정보 분석을 위한 기계학습을 설명하고자 한다. 기계학습의 일반적 정의와 특성, 그리고 빅데이터 특성에 의한 기계학습의 변화를 확인하고 특별히 다양한 변화 중에서 분산 및 병렬화를 통한 스케일러블 기계학습을 중점으로 주어진 빅데이터를 효율적으로 분석할 수 있는 다양한 플랫폼들과 프레임워크들을 설명한다. 더불어 실제 다양한 응용 활용을 제공하고 있는 Google API 같은 빅데이터 분석 기계학습 프로젝트들을 통해서 기계학습을 통한 빅데이터 분석에 대한 폭넓은 이해를 전달하고자 한다.

Ⅰ. 서 론

최근 온라인 상점의 추천서비스와 심야버스 노선 선정 사례와 같이 다양한 분야에서 활용되고 있는 빅데이터 분석은 연구자들로부터 많은 관심을 받고 있다[1]. 이처럼 빅데이터 분석이 활발해진 이유는 다양한 매체로부터 생성되는 데이터의 폭발적인 증가뿐 아니라 고성능이나 저비용의 하드웨어와 데이터베이스, 데이터 처리 기술 같은 고도화된 소프트웨어 기술이 있었기 때문이다. 빅데이터 분석의 대표 핵심기술인 기계학습 (Machine Learning)은 일반적으로 주어진 Training Dataset으로부터 최적의 학습모델을 생성하고, 해당 모델을 통해 대용량 데이터를 분석, 귀납 추론 등의 결과를 도출하는 방법을 총칭하며 [2] 수리 통계 이론에 기반한 분류, 군집, 추론과 같은 강력한 데이터 분석 기능을 제공함으로써 빅데이터 분석에 폭넓게 활용되고 있다.

본고에서는 우선 일반적인 기계학습 기능과 최근 대두되고 있는 빅데이터의 특성을 알아보고, 이를 바탕으로 최적화된 빅데이터 분석을 위한 스케일러블 기계학습 (Scalable Machine Learning) [3] 과 Transfer Learning [4] 같은 최신의 관련 연구들을 살펴봄으로 빅데이터 분석 측면에서의 기계학습 발전 방향을 살펴보고자 한다.

Ⅱ. 기계학습 (Machine Learning)

기계학습은 다양한 확률, 조합 이론과 수학적 최적화 기법, 통계, 알고리즘, 컴퓨터 구조를 활용해 이상적인 학습모델을 구축하는 기술로 연구자의 경험적 지식 습득과 그 응용 방법까지 포함하는 포괄적인 융합 기술이다. 관련 학문의 경계가 허물어져 합쳐진 기술인 기계학습의 명확한 정의를 내리는 것이 어려워진다는 것은 그만큼 기계학습의 연구 범위와 활용 범위가 더욱 늘어나고 있다는 것이며, 기계학습이 실생활에 미치는 영향도 더 커짐을 의미한다. 본 장에서는 기계학습의 일반적인 정의와 연구 변천사, 그리고 주요 이론들의 분류를 통해서 전반적인 기계학습의 이해를 돕고자 한다.

기계학습은 데이터로부터 학습모델과 예측 알고리즘을 만드는 것이 주요 목적이다. 입력 Domain x (입력데이터)로부터 출력 Domain y (가능한 예측)로 연관하는 함수 (Function) f: x→y을 학습하는 것이며, 함수 f는 선형과 같은 특정 함수 종류로부터 선택되며 x와 y 는 각각 데이터 객체와 예측을 표현한다[5].

일반적인 기계학습은 Learning과 Inference으로 구성되는데 Learning은 예측함수 f를 확인하는 과정을 이야기하고, Inference는 Data Instance x에 따른 f(x)를 계산하는 것을 의미한다[1]. 대부분의 학습 알고리즘은 Inference 를 최적의 f를 찾는 Training Data로부터 임시 후보인 f'을 예측하기 위해 Learning 과정에 일부로 포함한다. Learning 혹은 Inference 과정은 학습 알고리즘이 주어진 Training Dataset에 대한 In-Sample Error (E_{in})와 새로운 입력 Dataset에 대한 Out-of-Sample Error (E_{out})을 최적화 [6]하기 위해 응용 분야에 따라 기계학습을 적용하는데 유연하게 변형되어가면서 사용된다.

1. 기계학습 연구 변천사

기계학습은 1950년대 Arthur Samuel에 의해 "Field of

study that gives computers the ability to learn without being explicitly programmed" [7]라 정의된 이후에 현재까지 수많은 변화 및 발전을 거치면서 현재 다양한 분야에서 활발하게 연구, 적용되고 있다.

초기의 기계학습은 인간의 학습과 같은 지적 활동을 하는 전형적인 형태인 체스나 장기 같은 게임에 적용하면서 시작되었다. Samuel은 논문 "Some Studies in Machine Learning Using the Game of Checkers"에서 어떤 상황의 정량적 평가를 구하는 평가함수와 이에 관련 파라미터 조정에 기초한 최초의 기계학습을 연구하였다[8]. 또한, 최근 활발하게 연구되고 있는 신경망 모델의 전신인 Perceptron이 Rosenblatt에 의해 개발 [9]되었지만, 후에 Minsky와 Papert에 의한 Perceptron 한계 연구 [10]로 연구 침체기를 맞이한다.

1970년대는 기계학습 이론적 연구 시초를 이룬 시기로써 파라미터 조정에 의한 학습으로부터 최적의 파라미터 획득을 목적으로 한 연구와 Winston에 의해 수치로 표현할 수 없는 개념으로부터 지식을 습득하는 학습 방법이 고안되었다[11]. 이외에도 Holland에 의해 고안된 유전 알고리즘처럼 생물 진화를 모방한 연산 방법은 기존의 기계학습 알고리즘이 문제를 수식화하고 미분하여 최적값을 찾은 것과 다르게 미분하기 어려운 문제들에 대해 적합한 값을 찾는 것에 목적으로 연구가 진행되었다[12].

1980년대의 기계학습은 다양한 책과 논문을 통해 이론적인 틀이 잡혔으며 [13], 1990년대는 생물 개체가 환경과의 상호작용에 의해 지식을 획득하는 것을 기초로 한 강화학습이 연구되었다 [14]. 1990년대 말에는 기계학습이 인터넷 발전과 맞물려 Data Mining, Text Mining 등의 구체적 사례 분석에 적용되면서 학술적인 측면뿐만 아니라 산업적인 측면에서 연구되었으며, 최근에는 빅데이터 시대에 맞춰 실질적인 활용 가치 창출을 위해 더욱 다양하고 폭넓은 연구들이 진행 중이다[15].

2. 기계학습의 일반적인 특징과 분류

현재까지 다양한 분야에서 개발된 수많은 기계학습 기법들과 알고리즘들을 명확하게 구분하거나 알고리즘간의 파생 관계 및 유사성을 판단하는 것은 상당히 어렵다. 하지만 기계학습 기법과 알고리즘을 빅데이터 환경에서 적절히 확장 적용하기 위해서는 우선 기계학습의 대표적인 기법들과 알고리즘들의 분류를 통한 전반적인 기계학습의 이해가 필요하다. 이런 분류들은 주어진 데이터와 적용 알고리즘의 특성 이해를 통해 해결하고자 하는 문제에 대한 최적해를 제공하는 기계학습 접근법을 선정하는데 중요한 정보를 제공한다.

기계학습이 적용되는 문제들은 입력데이터와 적용환경, 학습방법, 요구되는 결과로 세분화되어 구분될 수 있으며 일반적인 분류로 주어진 입력데이터를 학습하는 방법에 따른 대표적인 분류와 그에 대응하는 학습방법 혹은 학습모델들을 살펴본다.

교사학습 (Supervised Learning) [5]: Training Data, 즉 학습을 위해 주어진 입력데이터가 메일의 Spam 혹은 Non-Spam의 예와 같이 알려진 라벨 (Label) 혹은 결과를 가지고 있는 경우이다. 일반적으로 학습모델은 학습과정을 통해서 얻어지는데 학습과정에서 잘못된 예측은 정답으로 알려진 라벨에 의한 수정 과정을 통해 점차 주어진 Training Dataset을 정확히 예측하는 학습모델을 얻게 되며, 학습과정은 학습모델이 주어진 입력데이터에 일정 수준의 정확도를 얻을 때까지 진행된다. Classification과 Regression 등이 대표적이며, Classification은 출력 Domain이 유한개의 불연속 집합 (Class, $y=\{c_1, \cdots, c_k\}$)인 반면에 Regression은 출력 Domain이 실수 집합 ($y \in R$)이다. 이외에도 복잡한 출력 Domain을 가지는 Structured Learning [16] 도 포함한다.

Data는 Data Instance인 x와 그에 상응하는 Ground Truth인 y로 구성된 Labeled Example $(x, y) \in X\ Y$ 형태를 가지며, 교사학습 알고리즘은 주어진 Training Data와 Test Data로부터 최적의 예측함수 f를 만든다. 주어진 데이터로부터 정확한 예측을 제공하는 함수 f를 찾는 교사학습은 x의 예측값인 f(x)와 대응되는 Ground Truth인 y간의 정량화된 차이를 최소화하는 것을 목적하는 예측오류 (Prediction Error; Loss) 함수를 이용하며, 최적의 함수f는 알려지지 않은 데이터에 대해서도 보장된 성능 (Generalization)과 주어진 학습 데이터에만 너무 최적화됨 (Over-fitting)을 최소화하는 특성을 가진다[6].

비교사학습 (Unsupervised Learning) [5]: 교사학습과 대조적으로 주어진 입력데이터가 라벨이 없는 경우, 즉 결과를 모르는 데이터로 구성되는 경우이다. 따라서 학습모델은 입력데이터의 특성이나 구조를 연역함으로써 얻어지게 된다. 주요 문제는 Association Rule Learning [17]과 Clustering [1]가 있으며, 적용될 수 있는 알고리즘에는 K-means [18]와 Apriori Algorithm [18]이 있다.

대표적인 비교사학습인 Clustering의 목적은 라벨이 없는 데이터들을 분할 군집하는 함수 f를 만드는 것이며, 같은 군집에 속한 Data Instance들은 다른 군집에 속한 Data instance들보다 상대적으로 유사할 것이다. Data Instance들의 거리 유사성 정의에는 다양한 방법이 있으며, 구체적으로 벡터 데이터인 경우 Euclidean Distance와 Cosine Similarity 등이 있다[5].

반교사학습 (Semi-Supervised Learning) [19]: 반교사학습은 라벨이 없는 대용량 데이터에 적은 수의 라벨이 있는 데이터가 포함된 입력데이터로부터 예측을 요구하는 학습으로 빅데이터 분

석에서 쉽게 확인된다. 예측을 목적으로 한 교사학습 특성과 학습모델이 라벨 없는 데이터간의 관계 구조도 알아야 하는 비교사학습의 특징을 함께 가진 경우이다. 라벨이 없는 데이터와 라벨이 있는 데이터간의 Smoothness, Cluster, Manifold 가정을 통해 문제 해결하며, Classification과 Regression이 대표적이다. Generative Models, Low-density Separation, Graph-based Methods, Heuristic Approach 등 라벨이 없는 데이터들의 모델 추정을 통한 예측 알고리즘의 확장을 통해서 결과를 얻는다.

강화학습 (Reinforcement Learning) [14]: 학습과정에서 입력데이터는 주어진 학습모델에 대해 대응 및 반응하는 환경에 대해 학습모델을 자극하는 수단으로 주어지며, 이는 교사학습의 학습과정과 다르게 환경으로부터 Punishment와 Reward로 구성된 피드백 과정을 통해 이루어진다. 대표적인 예로는 로봇 제어가 있으며, 대표적인 학습으로 Q-Learning [20], Temporal Difference Learning [21], Partially Observable Markov Decision Process (POMDP) [22]이 있다.

많은 경우의 기계학습 문제들이 교사학습과 비교사학습을 이용하여 결과를 얻고 있지만, 최근 Image Classification 같이 일부의 라벨 데이터로부터 대용량의 데이터를 학습하는 반교사학습 연구들이 활발하게 진행되고 있다. 강화학습의 경우도 로봇 제어나 제어 시스템 개발 분야에서 활발하게 연구되고 있다.

3. 기계학습 알고리즘의 분류와 특징

앞선 기계학습 문제의 세분화 외에도 기계학습 알고리즘의 방법에 따른 분류는 기계학습의 대한 폭넓은 이해와 주어진 빅데이터 분석에 최적화된 알고리즘 적용에 도움을 제공한다.

Regression Methods [5]: 알고리즘은 주어진 데이터와 선택된 학습모델에 의해 얻어진 예측값간 오차를 최소화하기 위한 반복적인 과정을 수행하면서 데이터들간의 관계를 모델링 한다. 주요 알고리즘으로 Ordinary Least Squares [18], Logistic Regression [18], Stepwise Regression [5] 이 있다.

Clustering Methods [5]: 주어진 데이터간의 유사성을 최대로 하는 군집 생성을 통해 데이터를 분류하는 방법으로 대표적인 알고리즘들은 데이터 객체간의 거리 근접 특성을 활용하는 Connectivity-based Clustering, K-means [18] Clustering 같이 중점 벡터를 이용하는 Centroid-based Clustering, Gaussian과 같은 통계 분포 모델을 이용하는 Distribution-based Clustering, DBSCAN 예처럼 데이터의 밀도간의 차이를 활용하는 Density-based Clustering으로 분류된다.

Decision Tree Learning Methods [25]: 데이터의 Feature 혹은 속성에 따라 Tree 형태의 의사결정 학습모델을 만들고, 분기 반복을 통해 주어진 문제에 대한 최종 결정을 도출한다. 주요 알고리즘으로 Classification and Regression Tree (CART) [18], Iterative Dichotomiser 3 (ID3) [18] 등이 있다.

Association Rule Leaning Methods: 데이터 변수들에서 관찰되는 주요한 관계를 가장 적합하게 설명할 수 있는 규칙을 찾는 알고리즘으로 높은 다차원이나 복잡한 관계를 가지는 데이터간에 중요 연관성을 찾는데 사용될 수 있다[17]. Apriori Algorithm [18], FP-Growth Algorithm [24] 등이 주요 예이다.

Artificial Neural Networks Methods [23]: 생물의 신경망 구조와 기능을 모방한 알고리즘으로 일반적으로 영상이나 음성 패턴 인식에 사용된다. 입력을 받아들이는 Input Layer와 중간 연결인 Hidden Layer, 결과를 출력하는 Output Layer로 구조를 가지며 각 Layer의 노드들을 상호 연결하는 가중치를 갱신함으로써 결과를 출력한다. Perceptron [9], Restricted Boltzmann Machine (RBM) [27] 등이 대표적이며, 최근 다양한 응용에서 활발히 연구되고 있는 Deep Learning은 Artificial Neural Network을 기초로 한다.

Deep Learning Methods [27]: 최근 컴퓨팅 능력의 향상과 가격 하락으로 인해 Artificial Neural Network의 한계 극복에 대한 활발한 연구가 진행되었으며, 이로 인해 더 복잡하고 더 큰 Network를 가지는 Deep Learning이 파생되었다. Deep Learning은 Manifold Learning과 유사하게 주어진 데이터로부터 Feature를 추론하고, 추론된 Feature로부터 파생된 새로운 Feature를 추론하는 과정을 반복하는 비선형 변환 조합을 통한 고수준의 데이터 추상화하는 학습을 진행하면서 결과적으로 패턴 인식을 한다. 최근 Deep Learning은 컴퓨터 비전, 음석인식, 자연어처리, 신호처리 분야에서 적용되어 뛰어난 성능을 보이며, Deep Belief Network (DBN), Convolutional Network, Stacked Auto-Encoders 등이 대표적이다.

Bayesian Methods [5]: 군집과 예측 문제를 풀기 위해 Bayes 이론을 확장, 적용한 알고리즘들로 Naïve Bayes [18], Bayesian Belief Network (BBN) [23] 등이 있다.

Regularization Methods [25]: Ridge Regression, Least Absolute Shrinkage and Selection Operator (LASSO) 등과 같이 학습에서 Over-fitting 최소화하는 목적으로 사용되는 알고리즘들이 있다.

Kernel Methods [26]: 다른 기본 학습 알고리즘들에 Kernel을 이용하여 입력데이터를 고차원의 벡터 공간으로 변환하여 선택한 학습모델에 의해 좀 더 쉽게 군집 혹은 예측한다. Radial Basis Function (RBF), Linear Discriminant Analysis (LDA) 등에 적용이 가능하다.

Dimensionality Reduction Methods [5]: Clustering Methods와 유사하게 데이터의 고유 구조를 찾지만, Dimensionality Reduction의 목적은 원 정보보다 더 적은 정보를 이용하여 데이터를 요약하거나 표현하기 위해서라는 점이 다르다. 데이터를 단순화하거나 낮은 차원을 가지는 데이터로 시각화하는데 이점이 있다. 예로 Principal Component Analysis (PCA) 등이 있다.

Ensemble Methods [28]: 독립적으로는 약한 성능을 가지는 여러 개의 학습모델을 조합하여 전체적으로 보다 나은 예측 성능을 가질 수 있도록 하는 방법이다. 주로 Weak Learner들의 선택과 최적의 조합 방법을 찾는 것이 알고리즘의 주요한 요소이다. Boosting, Bootstrapped Aggregation (Bagging), AdaBoost, Random Forest들이 대표적이다.

Ⅲ. 빅데이터 (Big Data)

최근 SNS의 성장과 스마트폰 같은 각종 디지털 기기의 다양화로 인해 생산, 전달, 관리, 저장되는 데이터의 양과 종류가 폭발적 증가하였으며, 이와 같이 기존의 일반적인 데이터베이스 기술로 처리, 분석할 수 없는 대용량 데이터를 'Big Data'라고 부른다[7]. 본 장에서는 기계학습과 같은 다양한 데이터 분석들을 활용함으로써 새로운 가치를 창출할 수 있는 빅데이터의 7가지 특징을 살펴보고자 한다.

Volume: 하루 동안 Facebook에서 공유되는 데이터양은 대략 100 Terabytes이며, 전체 데이터의 90%가 최근 2년사이의 생성된 데이터의 양일 정도로 방대한 대용량의 데이터들이 생산되고 있고 앞으로 전세계의 데이터양은 2년마다 2배씩 성장할 것으로 예측된다[15]. 이는 기존 데이터베이스 관리도구의 데이터 수집, 관리, 분석 능력을 넘어선 규모로 앞으로 분야에 따라 몇 십 Terabytes에서 수 Petabytes까지 크기 이상 계속적인 증가가 예상되며, 데이터의 크기가 빅데이터의 가장 중요한 요소이자 특성임을 보인다.

Velocity: 매분 100분정도의 영상들이 YouTube에 공개되고 있다는 사실로부터 빅데이터는 빠른 데이터의 생산뿐 아니라, 데이터의 분석 및 처리, 저장 속도의 향상이 이루어짐을 알 수 있다[15].

Variety: 과거에는 일정 형태를 가지는 정형(Structured) 데이터들이 대부분이었으나, 최근에는 시공간 데이터부터 사진과 영상까지 생성되는 데이터의 90%가 형태와 구조화할 수 없는 비정형(Unstructured) 데이터들이다[15]. 비정형 데이터는 처리 복잡도를 높이는 주요 이유이며, 빅데이터 분석에서는 이 같은 비정형 데이터와 반정형(Semi-Structured)뿐만 아니라 기존의 정형 데이터도 포함한 다양한 형태의 데이터 접근이 요구된다.

Variability: 빅데이터의 가변적이라는 특성은 데이터 의미의 변화를 의미하는데, 동일한 단어라도 문장에 따라 다양한 의미를 가질 수 있음과 동일하다. Tweets 데이터를 가지고 감정분석을 할 때, 정확한 분석을 위해서는 동일한 감정을 표현하는 단어라 할지라도 문맥적인 해석이 필요함은 가변적이라는 빅데이터의 특성의 사례이다[15].

Veracity: 다양한 대용량의 데이터가 빠르게 생성되더라도 데이터 자체가 부정확하다면 유의미한 결과를 도출하기 어려우며, 심지어 부정확한 데이터는 문제도 만들 수 있기 때문에 빅데이터에서 정확은 중요한 요소가 된다[15].

Visualization: 데이터가 방대해질수록 올바른 분석을 위해 데이터 시각화 방법을 이용하며, 패턴 분석과 같은 전체 데이터를 직관적으로 확인할 수 있는 기술들이 요구된다[15].

Value: 약 3천억 달러의 잠재 가치를 창출이 예상된 빅데이터 기술의 미국 의료보호 정책 활용 사례 [15]는 빅데이터가 경제적인 가치뿐만 아니라 복지의 균등한 분배 같은 파생적인 가치 창출을 발생할 수 있음을 보인다.

대용량, 빠른 속도, 가변성과 다양성으로 인한 높은 복잡도 같은 빅데이터 특성들로 인해 기존 데이터 처리도구를 이용한 분석 방법에는 제약이 있다. 구체적으로 일부 상용화 시스템을 제외한 대부분의 DBMS (Database Management System)는 대용량의 데이터 분석과 스트리밍 데이터 수용 능력이 떨어지고, 다양한 고급 통계적 모델링과 기계학습 분석이 어렵다[24]. SAS, R, Matlab과 같은 고급 분석 도구들은 상대적으로 복잡한 분석이 가능하지만, 단일 컴퓨터의 메모리 정도의 데이터 분석으로 제한되어 확장에 어려움이 존재한다[3]. 따라서 빅데이터 분석을 위한 분산 및 병렬 환경의 데이터 분석 기술들과 스케일러블 기계학습 관련 기술들이 요구되며, 실제로 이를 활용한 Text Mining, Opinion Mining, Social Network 분석 및 군집 분석 관련된 빅데이터 분석이 연구 중이다. 특히 Yahoo, Amazon, Google 등의 회사들은 독자 개발한 기술과 오픈 소스화된 분석 인프라 기술을 활용하여 빅데이터 분석을 하고 있다[7].

Ⅳ. 빅데이터 분석을 위한 기계학습

빅데이터 시대의 도래로 인해 기계학습은 학문적 연구로부터 산업적이고 실용적인 측면의 연구로 많은 변화를 겪고 있으며, 빅데이터의 특성을 이용하는 Deep Learning 등의 최신 기계학습 연구들은 실생활 활용을 목적으로 다양한 연구를 시도하고

있다[3].

본 장에서는 다양한 빅데이터 분석을 위한 기계학습 연구들 중에서 스케일러블 기계학습에 의한 빅데이터 분석을 중점으로 서술하려 하며, 이를 스케일러블 기계학습의 빅데이터 분석에 대한 적합성 및 주요 방법들, 구체적인 스케일러블 기계학습을 분산 및 병렬 환경에 적용 사례 순으로 살펴보고자 한다. 더불어 빅데이터 분석 분야에 최적화된 기계학습 사례들을 알아봄으로써 빅데이터 분석을 위한 포괄적인 기계학습의 이해를 높이고자 한다.

1. 빅데이터 시대에 따른 기계학습의 변화

지난 수십 년 동안 대규모 Dataset에 대한 분산 및 병렬 처리는 높은 비용으로 인해 특수한 경우와 분야에서만 활용되었다. 하지만 최근 저렴해진 설치 비용과 다양한 분야에서의 활용 요구 증가로 인해 수많은 병렬 연산 플랫폼이 등장하였으며, 이에 따라 병렬 및 분산 환경에서의 다양한 종류와 크기의 Dataset을 처리하기 위한 기계학습 관련 연구도 활발하게 되었다. 특히 스케일러블 기계학습 연구는 기존의 많은 기계학습 알고리즘을 병렬화할 수 있는 높은 수준의 하드웨어 구조와 프로그래밍 프레임워크의 발전으로부터 비롯하였으며, 실제로 수많은 병렬 및 분산 플랫폼들이 기계학습에서 요구하는 데이터나 데이터의 Feature에 대한 반복적인 연산의 동시 처리 (Concurrent Processing)를 가능하게 하였다[3]. 최근 빅데이터 분석 응용들의 대용량 Dataset을 사용은 스케일러블 기계학습에 대한 관심을 증대시켰으며, 실제 대용량 데이터들은 대체로 분산 저장 플랫폼에 저장 및 처리되어 있기 때문에 기계학습도 적절하게 분산 환경에서 적용될 수 있는 방법의 개발이 필요로 되었다. 더불어서 음성인식과 사물인식과 같은 고차원이며, 복잡한 특징으로 표현되는 정보로부터 실시간 추정을 요구하는 기기들의 확산은 빠른 분석을 위해 기계학습의 병렬화를 필요로 하게 되었다[3].

2. 빅데이터 분석을 위한 기계학습의 특징

대용량 Data Instances: 기계학습이 적용되는 많은 경우에는 잠재적인 Training Example의 수가 방대해서 단일 연산 기기에서는 분석 실행 불가능하다. 웹과 금융과 같은 분야의 매일 10억개 이상 이벤트들이 합쳐진 Dataset은 잠재적인 기계학습 알고리즘의 입력이 될 수 있으며, 최근 IoT (Internet of Things) 시대의 도래로 연속적으로 정보를 수집하는 센서를 가진 기기들로부터 얻어진 데이터도 Training Data로 여겨질 수 있다. 만일 10억개의 Data Instance들이 평균적으로 '0'이 아닌 수천의 Feature들을 가진다면 결과적으로 매일 10^{12}개 이상의 Instance-Feature Dataset이 매일 생성되며, 각 Feature이 1Byte이라 해도 Dataset은 쉽게 수백 Terabytes에 도달한다. 이 같은 대용량 Dataset에 선호되는 효율적인 처리는 분산 저장과 기기들의 클러스터 형태의 처리 대역폭의 결합이며, MapReduce [29]와 DryadLINQ [30] 등과 같은 연산 프레임워크들이 대표적이다. 실제로 높은 수준의 분산 저장 플랫폼에 병렬화할 수 있는 프레임워크가 융합하여 대용량 데이터에 대해 다양한 기계학습 분석 기능을 제공한다.

높은 차원 (Dimensionality)의 입력: 기계학습에서는 데이터의 차원이 증가할수록 데이터 공간이 기하급수적으로 증가하고 데이터 밀도는 희박해지기 때문에 모델추정에 필요한 데이터도 기하급수적으로 증가함에 따른 분석의 어려움을 차원의 저주 (Course of Dimensionality)로 정의한다[6]. 일부 데이터들이 산발적으로 분포된 경우도 있지만 실제 자연어, 사진, 영상 등에 관련된 기계학습 입력의 차원은 일반적으로 분석되는 Data Instance들보다 많은 Feature들을 가지며 이에 따라 데이터 분석에 어려움을 가진다. 데이터들이 많은 Feature를 가지는 응용의 경우, Feature 집합에 따라 연산 Task을 분할하여 병렬화한 기계학습 알고리즘을 적용함으로써 해결이 가능하다[24].

모델과 알고리즘의 복잡도 (Complexity): 기존의 기계학습은 SVM처럼 단순하면서 상대적으로 연산비용이 저렴한 선형 모델들의 변환을 통해서 높은 정확도를 얻었다. 하지만 최근 Decision Tree Ensemble 혹은 Multi-Layer (Deep) Network 등 높은 정확도를 가지는 많은 학습 알고리즘들은 높은 연산 복잡도를 가진 비선형 (Non-Linear) 모델을 이용하거나 높은 연산 처리 비용을 가지는 서브루틴을 사용함으로써 높은 복잡도와 매우 느린 학습 과정 등의 단점을 가진다[3]. 하지만 기계학습의 병렬화는 언급된 단점을 최소화한 구현을 제공하며, 영상 등 일부 응용 데이터들처럼 본질적으로 기본 Feature들에 대해서 비선형 구조를 가지는 경우에도 적용될 수 있다. 실제로 대용량의 Dataset에 대한 높은 복잡도를 가지는 알고리즘이나 모델은 병렬 Multimode나 Multicore 기법 활용한 스케일러블 기계학습에 의해 구현될 수 있으며, GPU와 같은 Coprocessors을 이용하여 복잡한 학습입력 공간에 대해 빠른 변환을 제공함으로 다양한 기계학습 알고리즘 병렬화에 유연성을 제공하였다[31].

추정 시간의 제약: 음성인식이나 로봇이동과 같이 실시간 판단이 필요한 분야들의 추정 속도에 대한 엄격한 제한은 알고리즘의 병렬화 기술을 통해서 충족될 수 있다. 관련된 초기 연구들은 Throughput에 향상을 초점으로 진행되었으나 최근에는 응용 시스템이 예측 결과를 기다리는 상황인 Inference Latency에 최소화에 집중되어 연구되고 있다. 실제로 차량의 센서들에 의해 실시간 판단하여 주행 경로를 만들어주는 자율

주행 자동차 분야에서는 전체 성능이 Latency에 의해 직접적으로 떨어지기 때문에 Latency는 중요한 요소이다[3]. 일반적으로 Throughput 제약과 Latency 제약은 상충적 관계로 인해 완벽하게 양립할 수 없으나, 최근에는 GPU나 FPGA와 같은 높은 병렬화된 하드웨어 구조에 의해 Latency와 Throughput간의 최적의 조합이 제공되는 기계학습이 구현될 수 있다[32].

추정 모델들의 연결: 객체 추적, 음성 인식 등의 실생활 많은 문제들은 독립적인 예측들의 연속 연결인 Prediction Cascades 처리로 결과를 출력한다. 이런 높은 복잡도를 가지는 공동의 출력 공간을 가지는 경우는 병렬화를 통한 기계학습을 통해 향상된 속도의 예측을 제공할 수 있다[24].

모델 선택과 파라미터 조정: 기계학습 알고리즘의 개발과 파라미터 조정, 평가로 구성되는 학습 흐름에서 Dataset에 독립적으로 실행되는 각 Task들은 상호간의 정보교환이 필요 없기 때문에 실제로는 병렬화되어 진행된다. 예로 Parameter Sweep은 같은 Dataset에 대해 선택한 학습 알고리즘을 다양한 설정으로 여러 번 수행하고, 이를 Validation Set에 의해 평가한다[6]. Cross-Validation 혹은Bootstrapping으로 불리는 Statistical Significance Testing은 Training과 Test이 Dataset의 서로 다른 부분 집합 조합으로 반복적으로 수행되며, 통계적인 중요도 측정과 이를 통합하여 결과를 도출한다[6]. 따라서 Parameter Sweeps과 Statistical Significance Testing 같은 학습 알고리즘의 파라미터 조정과 통계적인 중요도 평가는 학습과정에서 독립적으로 병렬화되어 적용될 수 있으며, 기계학습의 병렬화 특성을 잘 보인다.

Ⅴ. 스케일러블 기계학습 (Scalable Machine Learning)

대용량 Dataset 처리는 병렬화 요소를 활용한 높은 성능을 가지는 스케일러블 기계학습 알고리즘을 통해서 분석 가능하며, 스케일러블 기계학습의 구현 요소로 FPGA (Field-Programmable Gate Arrays)와 같은 Customizable Integrated Circuits, GP-GPU (General-Purpose Graphics Processing Unit)과 같은 Custom Process Units, Multiprocessor and Multicore Parallelism과 HPC (High-Performance Computing), 그리고 같은 고속 지역 망에 의해 연결된 클러스터, 상용화 Cloud Computing 제공자에 의한 데이터센터 형태의 Virtual Clusters등 다양한 기본 플랫폼들과 함께 스케일러블 알고리즘을 구현할 수 있는 관련 프로그래밍 프레임워크들을 이용한다[33]. 이러한 다양한 플랫폼들과 프레임워크들은 연구자들에게 폭넓은 연구 기회와 다양한 분석 방법을 제공한다.

1. 스케일러블 기계학습의 병렬화 종류

병렬 및 분산 컴퓨팅을 적용 가능한 하드웨어 자원을 이용하는 기계학습 개발은 기계학습 알고리즘 수정이나 재설계를 요구한다. 즉, 선택된 분석 인프라에 적합하도록 예측 모델이나 학습 알고리즘의 처리 구조 및 데이터흐름, 기본 Task 분해를 결정해야 한다[3]. 앞으로는 다양한 스케일러블 기계학습 알고리즘에 대한 병렬 및 분산 요소들을 살펴보고, 관련 플랫폼과 프레임워크에 대한 구체적인 예들을 알아보고자 한다.

스케일러블 기계학습은 주어진 데이터와 Task들을 동시 실행하기 위한 병렬화 접근과 분산 시스템들을 통해 얻어지며, Data Parallelism과 Task Parallelism, Request Parallelism으로 분류된다. Data Parallelism은 동시에 여러 개의 입력을 처리함으로써, Task Parallelism은 알고리즘 실행을 여러 개의 Task를 부분으로 나누고 독립적으로 동시 실행함으로써, Request Parallelism은 주어진 요청들을 병렬 처리함으로써 얻어진다[33].

Data Parallelism: 다중 입력에 대해 동시성이 보장된 연산을 실행함으로 특정 확률 분포로부터 독립적인 샘플의 묶음을 입력으로 받아들이는 기계학습 응용과 알고리즘에 적합하다. 주어진 Dataset을 Instance-by-Feature 행렬 형태로 표현하면 Data Parallelism을 얻기 위한 두 개의 직교 방향을 확인할 수 있는데 하나는 Logistic Regression의 가중치 갱신하는 연산과 같이 행 분할을 통해 Instance를 부분으로 나누어 독립적으로 연산하는 방법과 Decision Tree 생성에서 Feature를 나누어 구분하는 것과 같이 Feature에 의해 분할 연산할 수 있도록 알고리즘에 맞춰 열로 분할 하는 방법이 있다[24].

Data Parallelism은 행과 열에 의해 독립 분할을 통해 얻어진 데이터의 부분 집합을 독립적으로 실행할 수 있도록 연산을 Concurrent Subtask하여 Master-Slave Model 형태를 가진 병렬화된 알고리즘을 구현한다[3].

만일 Instance나 Feature가 독립이 아닌 경우에는 알고리즘의 병렬화가 쉽지 않지만, Instance와 Feature간의 관계나 구조를 표현하는 그래프로부터 얻어진 결과에 따라 Instance나 Feature를 분할함으로써 Data Parallelism을 구현할 수 있다[33]. 이와 같은 방법은 분할된 데이터 부분집합의 동시 연산 실행에서 서로간의 정보 교환이 필요하기 때문에 알고리즘 패턴에 의존적인 측면이 존재한다.

또한 Instance 혹은 Feature 분할된 부분집합을 Coarse-grained Data와 Fine-grained Data로 구분된다. 보통 Coarse-grained Data 병렬화는 알고리즘 재설계를 통해 얻어지고, Find-grained Data 병렬화는 병렬화된 벡터와 행렬 연산을 가능하게 하는 프로세서 등의 하드웨어로 구현될 수 있다[3].

Task Parallelism: 병렬화를 진행함에 있어 전체 알고리즘을 동시 실행될 수 있는 부분으로 나누는 것이 중요하다. 일반적으로 연산에 대한 Fine-grained Task Parallelism은 Pipelining과 같은 컴퓨터 구조에 의해 얻어질 수 있으며, 높은 효율의 성능을 가진다. Coarse-grained Task Parallelism은 각 Task에 대한 명확한 정의가 요구된다[3].

주어진 알고리즘을 나누는 것은 각 Task를 Node로, Task간의 관계를 Edge로 표현하는 방향을 가진 비순환식 그래프로 표현될 수 있으며, Task간의 데이터 흐름은 간선에 의해 전달된다[33]. 이를 활용한 대표적인 분산 연산 프로그래밍 모델은 MapReduce이 있다[34].

Request Parallelism: 실시간 검색과 같이 대용량의 요구들을 처리하는 응용 분야에서 수많은 Request들을 요청 종류와 목적에 따라 분류하여 독립적으로 병렬 실행함으로써 빠른 처리를 제공하는 기술을 말한다[35].

최근 연구들에서는 Data와 Task, Request에 관련된 다양한 병렬화 요소들을 혼합한 Hybrid Parallelism을 통해 더 효율적인 스케일러블 기계학습을 만들고 있다[3].

2. 스케일러블 기계학습 분석 고려사항

스케일러블 기계학습 구현에 이용될 수 있는 다양한 병렬 및 분산 플랫폼들과 프레임워크들간의 차이를 살펴봄으로써 빅데이터 분석 연구자들이 자신들의 분석에 적당한 기계학습 분석 방법 선택에 대한 이해를 돕고자 한다.

병렬화 정도: GPU, FPGA와 같은 특화된 하드웨어 솔루션을 적용한 경우에는 프로세서들이 Pipeline 적용으로 높은 Throughput을 가짐에 따라 단순 기초 연산 Task와 Data 에 대해 Fine-grained Parallelism 구현에 이점이 있지만 Task 간의 데이터흐름과 병목현상 관점에서 전체 알고리즘의 재정의가 필요하다[33].

CPU와 같은 코어와 프로세서에 대한 병렬화인 경우에는 앞선 Fine-grained 학습 알고리즘의 병렬화보다 완화된 제약사항을 가지며 대상도 단순기초 계산연산에만 국한되지 않는다. 클러스터나 데이터센터 솔루션의 경우에는 상대적으로 통신 비용 증가로 더 높은 Granularity Task 정의가 중요하다[3].

알고리즘의 재정의 정도: 동시성과 병렬화를 위한 알고리즘의 확장은 자동 병렬화 솔루션에 의해 제공되는 간단한 방법부터 완전히 알고리즘을 재정의하거나 직접 하드웨어에 구현하기까지 다양하다. GPU와 같은 하드웨어 특화된 플랫폼에 기계학습 알고리즘을 직접 구현할 때는 분기와 같은 특정 소프트웨어 패턴들을 지양함과 Hardware-aware Task Configuration 등에 특화된 지식이 요구된다. 반면에 고수준의 병렬 및 분산 시스템은 API에 의해 확장된 프로그래밍 언어를 이용하여 손쉽게 병렬화를 제공한다[3].

프로그래밍 패러다임의 혼합 정도: Functional Programming부터 SQL까지의 다양한 기능들을 차용한 Declarative Programming 언어들은 알고리즘을 논리 연산과 데이터흐름의 혼합 형태로 표현함으로써 분산 프로그래밍을 용이하게 하기 때문에 대용량 데이터 조작처리에 점차 많이 사용되고 있으며 [3], MS의 DryadLINQ [30], Google의 Sawzall과 Pregel [36], Apache Pig와 Hive [37] 등의 언어들이 분석 양상에 따라 적절히 사용되고 있다[3].

Dataset 크기 정도: Dataset이 너무 커서 메모리 환경의 분석이 불가능한 응용들은 일반적으로 분산 파일시스템이나 공유 메모리 클러스터에 의존하며, 이런 분산 데이터 저장과 연결에 의한 병렬 연산 프레임워크는 Scheduling동안에 Local Data 흐름을 최대화하는 최적의 Task 할당을 제공한다. 반면에 하드웨어 특화된 병렬의 Scheduling은 대용량 Dataset에 대한 저장 솔루션과 분리되어 있기 때문에 Throughput을 극대화하기 위한 조정이 필요하다[33].

실시간 분석: 일반적으로 클러스터 환경의 MapReduce로 구현된 학습 알고리즘은 분석 예상 시간을 보장할 수 없다. 따라서 정적인 Dataset으로부터 학습을 하는 Offline Learning과 달리 데이터가 실시간으로 입력되는 Online Learning의 경우, 분산 플랫폼을 이용해 스케일러블 기계학습은 하드웨어 특화된 분석보다 지연 발생을 더욱 고려해야 한다[3].

3. 스케일러블 기계학습 성능

기계학습에 대한 성능은 Model Accuracy, Sensitivity, Specificity, F-measure, Positive Predictive, Negative Predictive 등 일반적인 예측 정확도와 신뢰도 [5]를 구하는 지표와 연산 속도 등의 여러 주요 지표들이 다양한 측정 방법을 통해 얻어질 수 있기에 단순 비교는 어렵다. 하지만 대부분의 알고리즘의 서브루틴들이 명확한 연산 복잡도를 가지기 때문에 알고리즘의 복잡도나 성능의 척도로 O(Input Size), O(Instances), O(Parameters) 등의 Big-O 표기법을 이용할 수 있으며 [5], 본 장에서도 대표적인 성능 지표를 이용하여 스

케일러블 기계학습 성능에 대한 고찰을 제시하고자 한다.

기계학습 알고리즘 서브루틴의 연산 복잡도: 일반적으로 스케일러블 기계학습을 적용하는 알고리즘의 복잡도가 O(Input Size)인 경우와 O(Input Size^k, $k \geq 2$)인 경우에는 큰 차이가 있으며, 실용적인 측면에서도 3차원 이상의 복잡도를 가진 알고리즘은 실생활에 적용하기가 어렵다[3].

기계학습 알고리즘 처리에 요구되는 대역폭: 대역폭은 알고리즘의 병렬화로 인한 자원 공유 과정에서 중요한 고려된 성능 요소이다. 공유 메모리나 공유 디스크 자원을 이용하는 스케일러블 기계학습 알고리즘보다 클러스터를 이용한 스케일러블 기계학습 알고리즘에서 더욱 고려되어야 한다.

기계학습 알고리즘의 연산 성능: 스케일러블 기계학습 알고리즘의 성능의 지표로 속도 향상 (Speedup), 효율 (Efficiency), 확장 (Scalability)을 고려할 수 있다[3]. 먼저, 속도 향상 지표는 순차적인 처리를 하는 원 알고리즘과 그에 상응하는 병렬화된 알고리즘간의 분석 시간 비율로 성능을 정량화한다. 효율 지표는 알고리즘을 처리하는 프로세스의 수와 속도 향상의 비율을 측정하며, 확장 지표는 처리하는 프로세스의 수가 증가함에 따라 증가하는 효율성간의 관계를 추적한다.

또한 스케일러블 기계학습의 알고리즘간 성능 평가는 이상적으로 같은 Dataset에 대해서 비교되더라도 응용 분야와 적용 프레임워크 등의 구현적인 측면의 다양성으로 인해 단순 비교가 어렵기 때문에 실행시간과 입력 크기의 비율인 Feature Throughput를 활용하여 대략적인 비교 추정되며, 단일 성능평가 지표보다는 종합적인 성능평가 방법이 적합하다.

4. 스케일러블 기계학습 프레임워크

스케일러블 기계학습을 위한 병렬화 프레임워크의 대표적인 예는 다음과 같다.

MapReduce [29]: Google에 의해 개발된 대표적인 분산 컴퓨팅 프레임워크으로 〈그림 1〉과 같이 Map, Shuffle & Sort, Reduce라는 단계를 거쳐서 대용량 데이터를 나누어 처리하고 결과를 모음으로써 분할 처리한다. MapReduce의 입력과 출력은 〈Key, Value〉쌍 형태의 데이터이며, Map 처리는 입력을 분할하여 각 Map Task에 할당하고, 각 Map Task는 입력을 정의된 연산을 병렬 처리하여 중간단계의 〈Key, Value〉 형태 결과로 출력한다. Map 처리 후 생성된 생성된 데이터는 Shuffle & Sort과정에 의해 같은 Key를 가진 데이터끼리 모이고, Reduce 처리에 의해 같은 Key별 데이터 집합 요소를 통합하는 병렬 연산 처리에 의해 최종 결과를 출력한다.

MapReduce는 분산 파일처리 시스템인 HDFS (Hadoop

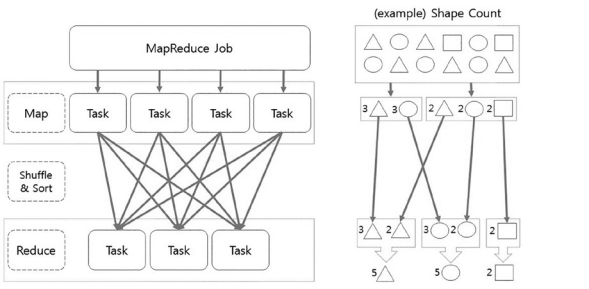

그림 1. MapReduce 기본 개념과 예 [29]

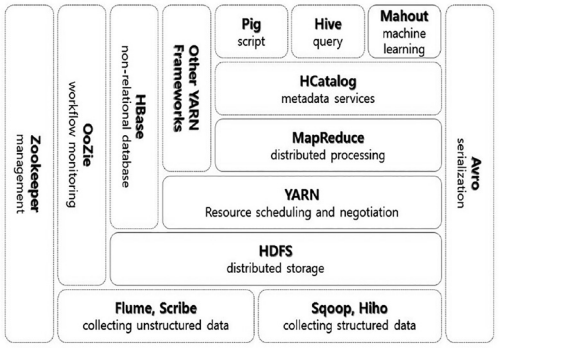

그림 2. Hadoop 2.0 구조 [38]

Distributed File System)과 함께 가장 대표적인 분산 처리소프트웨어인 Hadoop의 기본 요소임을 Hadoop을 설명하는 〈그림 2〉와 〈표 1〉로 확인할 수 있다.

표 1 Hadoop 주요 요소 및 설명 [38]

Hadoop요소	설명
HDFS	높은 Throughput을 유지하면서 동시에 신뢰성을 추구하는 분산 파일 시스템
YARN	Hadoop 2.0부터 자원의 관리하고 응용프로그램 실행을 용이하게 하는 기능 제공
MapReduce	클러스터 환경에서 병렬 분산 처리를 수행하기 위한 프레임워크
Hive	Hadoop조작을 쉽게 하기 위한 SQL 형식의 인터페이스
Pig	Hadoop 데이터 흐름을 기술하기 위한 스크립트 언어 방식 인터페이스
HBase	대용량 테이블을 관리하기 위해서 확장성을 추구한 Key-Value 방식 저장소
ZooKeeper	분산 클러스터 환경에서 동작하는 응용프로그램을 관리

또한, MapReduce는 내고장성 (Fault Tolerance)를 고려하여 저비용의 장비를 분산 컴퓨팅의 노드로 활용할 수 있게 하는 비용효율성 (Cost Efficiency)와 대용량 데이터의 효율적인 분석 처리를 위해 분산 노드 수의 확장성 (Scalability)을 보장한다

Spark [50]: MapReduce와 유사한 분산 컴퓨팅 프레임워크이지만 성능 향상을 위해 In-Memory에 데이터 집합의 추상화된 객체인 RDD (Resilient Distributed Dataset)형태로 데이터를

유지한다. 특히 반복 연산을 수행하는 알고리즘과 데이터 흐름 처리에 유용하며, 이는 Hadoop에서 Iterative Algorithm을 수행할 때, 반복 연산마다 HDFS에 쓰기와 읽기가 반복되면서 디스크 I/O와 네트워크 I/O가 생기는 단점을 극복한다. Spark은 Scala로 개발되었으며 Spark에 특화된 Scala Interpreter를 사용하고, Map, Filter, Join 등 병렬 연산을 위한 고수준의 명령어 집합을 제공하여 개발을 쉽게 한다. Spark는 대화형 질의 분석기인 Shark와 대용량 그래프 처리 및 분석기인 Bagel, 그리고 실시간 분석기인 Spark Streaming, 기계학습 라이브러리인 Mllib 등을 함께 제공하며, 여러 분석 도구를 혼재 사용하지 않고 하나의 프로그래밍 패러다임으로 손쉽게 적용 가능하다.

DryadLINQ [30]: MS의 DryadLINQ는 기계학습 알고리즘을 Declarative Programming 언어 [2]를 이용하여 Dryad Cluster에 의해 실행될 수 있도록 분산 연산 형태로 컴파일 한다. DryadLINQ는 높은 수준의 데이터 추상화 기능뿐만 아니라 다양한 유용한 소프트웨어 도구와 개발 환경 통합, 그리고 표준 라이브러리 호환성 등의 유용함을 제공한다. 더욱이 Dryad 실행 엔진과의 높은 호환성으로 인해 대규모의 클러스터에 대해서도 신뢰성과 확장이 보장되는 실행을 제공한다.

IBM Parallel Machine Learning Toolbox (PML) [40]: MPI 기반 [41]의 기계학습 알고리즘 병렬화 기능을 제공한다. PML은 주어진 알고리즘을 Associativity, Commutability 같은 규칙에 따라 연속된 연산자들로 표현함으로써 병렬화를 용이하게 하고, 특별히 데이터가 동시에 여러 개의 경로에 의해 처리되어야 하는 알고리즘 적용에 이점이 있다.

Compute Unified Device Architecture (CUDA) [31]: NVidia에 의해 고안된 병렬 연산 플랫폼과 프로그래밍 모델로 GPU 능력을 활용하여 컴퓨팅 능력을 향상시킨다. 이같이 GPU를 본래의 용도가 아닌 범용 연산 용도로 활용하는 것을 GPGPU라 한다. 많은 프로세서와 넓은 대역폭을 가지고 빠른 연산처리가 가능한 GPU를 병렬 처리로 연산 횟수를 크게 줄일 수 있는 기계학습이나 실시간 분석을 요구하는 기계학습에 적용한다면 효율적인 결과를 얻을 수 있으며, CUDA는 이런 GPU의 병렬화를 위한 프로그래밍으로 확장된 C언어 형태로 지원한다.

앞에 기술된 플랫폼과 프레임워크 외에도 주어진 분석 응용 대상에 적합한 플랫폼과 프로그래밍 프레임워크를 선택하여 적절한 솔루션으로 최적의 분석을 위한 수많은 관련 연구들이 진행중이다.

5. 스케일러블 기계학습 알고리즘의 예

PSVM (Parallel Support Vector Machines) [42]: 대표적인 기계학습 알고리즘인 SVM을 병렬화하는 방법에는 Incomplete Cholesky Factorization에 의해 Kernel 행렬을 추정하고 분해하여 병렬화한 IPM (Interior Point Method)을 이용한 접근과 하드웨어 가속기를 활용한 SMO (Sequential Minimal Optimization)을 이용한 접근이 있다.

Boosted Decision Tree 병렬화: 우선 LambdaMART [43]과 같이 정보검색 (Ranking)을 위해 MPI [41] 기반에 병렬화된 Boosting 방법과 기존의 Boosting 방법에서 반복 횟수를 현저하게 줄임으로 성능 향상을 제공하는 Transform Regression [44]이 있다.

Graphical Model 병렬화: 추정 예측을 위해 확률 그래프 모델을 기초로 한 Factor Graph을 이용한 병렬화된 Belief Propagation [45]가 있다.

Spectral Clustering 병렬화: MapReduce와 MPI를 이용하여 Spectral Clustering을 병렬화하기 위해 우선 주어진 Affinity 행렬을 희소화 (Sparsification)하고, Subsequent Eigen-Decomposition을 이용하여 분해한다. 얻어진 결과에 Projected Instance를 적용한 K-means를 통해서 최종 군집을 만드는 방법이 있다[46].

Information-theoretic Co-clustering 병렬화: 정보이론에 기초하여 데이터의 Instance에 대한 군집 생성과 Feature에 대한 군집 생성을 동시에 진행하면서 두 군집간의 상호정보량 (Mutual Information)이 최대가 되는 최적의 값을 찾는 학습 방법이 있다[47].

Online Learning 병렬화: Training Instance가 스트리밍 형태로 제공되며, 개별 Example에 대해 즉각적인 학습이 수행되어야 하는 응용에서 적용된다. 이러한 응용은 학습에 대한 갱신 (Update) 지연이 더 많은 오차를 초래하므로, 좋은 결과를 얻기 위해 알고리즘이 갱신 지연을 최소화하는 방법에 의해 병렬화되어야 한다[48].

Transfer Learning 병렬화: 일반적으로 기계학습의 Training Data는 앞으로 수집, 분석될 데이터는 같은 Feature 공간과 같은 확률 분포를 가짐을 가정한다[2]. 하지만, 실제의 분석 환경에서 이 가정은 유지되기 어려우며, 이러한 가정의 문제를 해결하기 위해 Source Domain과 Target Domain간의 지식 전이 관계를 규명하는 Transfer Learning [4]을 정의하였다. 실제 대용량 데이터 분석에서 활용될 수 있는 Transfer Learning의 확장으로는 Cooperative Matrix Factorization (CoMF)를 이용하여 병렬화를 적용한 Distributed Transfer Learning이 있다[49].

Ⅴ. 빅데이터을 위한 기계학습 분석 도구

DBMS는 스트림 데이터 같은 빅데이터 분석을 위해 Approximation, Single-pass/Sub-linear 알고리즘, Massive 데이터를 취합하기 위한 샘플링 기법을 제공하는 별도의 추가 도구 박스를 제공하고 있다. 더불어 DBMS와 같은 대용량 데이터 처리 플랫폼과 SAS, R, Matlab 등의 분석 패키지를 연동하는 Extend Relation Model, Extend MapReduce/Hadoop Model 연구들이 진행 중이며, 이는 DBMS 확장으로 통계적 분석뿐 아니라 대용량 기계학습 알고리즘 제공을 목표로 한다[24].

이외에도 분산 컴퓨팅 환경에서의 빅데이터 분석을 위한 다양한 기계학습 도구 개발 프로젝트들이 진행 중이다. 대표적으로 MapReduce 기반으로 다양한 기계학습 알고리즘을 제공하는 추가 프레임워크을 개발하는 Apache Mahout [50], University of Washington의 HaLoop [51] 등이 대용량 기계학습 알고리즘 개발을 목적으로 활발히 활동 중이다.

Mahout [50]: Apache Software Foundation에서 진행하는 오픈소스 프로젝트로 대용량 데이터 분석을 목적으로 분산 및 병렬 처리가 가능한 스케일러블 기계학습 라이브러리이다. 초기에는 MapReduce 기반의 기계학습 알고리즘 라이브러리를 개발하였으나, 최근에는 포괄적인 데이터 처리 시스템을 지원하는 기계학습 라이브러리 개발하고 있다. Mahout이 이론적으로 기계학습 기술구현에 대해 개방적이라 하지만, 실질적으로는 추천 엔진 (Collaborative Filtering), 군집, 분류 같은 주요 영역에서의 개발이 집중하고 있다.

Mahout과 유사하게 특정 대용량 데이터에 대해 최적화된 기계학습 처리 능력을 가지는 GraphLab [52] 패키지가 있으며, GraphLab는 그래프 기법을 활용하여 반복적인 기계학습을 처리할 수 있는 확장 프레임워크를 제공하지만 데이터 관리 플랫폼이 아니기에 메모리에 적당한 Dataset에만 적용할 수 있다.

간편하게 기계학습을 이용한 빅데이터 분석을 제공하는 상용 서비스의 예로 IBM의 SystemML [53]과 Google Predication API [54] 이 있다.

SystemML [53]: 일반적으로 대규모의 클러스터를 환경에서 포괄적인 분산 프레임워크로 MapReduce가 활발하게 사용되고 있지만 많은 기계학습 알고리즘들을 저수준의 MapReduce 작업으로 구현하는 것은 어려운 일이다. 그래서 IBM에서는 기계학습 알고리즘을 DML (Declarative Machine Learning Language)라는 고수준의 언어로 표현하고 컴파일하면 MapReduce 환경에서 쉽게 실행할 수 있는 SystemML을 개발하고 상용 서비스 하고 있다.

Google Prediction API [54]: Google의 Cloud 환경에서 제공되는 기계학습 도구로 고객 분석, 스팸 감지, 메시지 전달 방식 결정, 상위 제품 판매 기획 분석, 문서 및 이메일 분류 기능들을 손쉽게 다양한 어플리케이션에 연동하여 사용할 수 있도록 API 형태로 제공하는 서비스이다. 다양한 대용량 기계학습 알고리즘을 구현, 제공한다는 점에서는 Mahout과 비슷하지만 분산 환경에서의 기계학습의 라이브러리만 제공하는 Mahout과 달리 Google Prediction API는 사용자가 분석할 데이터만 있다면, Google의 Cloud Storage, BigQuery와 같은 강력한 인프라를 RESTful 인터페이스를 활용하여 간편한 대용량 데이터 분석이 가능하다.

Ⅵ. 결 론

일반적으로 빅데이터의 처리와 분석은 SNS, 웹과 같은 다양한 매체로부터 생성되는 데이터를 수집하는 Collecting 단계로부터 시작하여 수집한 데이터를 분산 저장 장치 플랫폼을 이용하여 저장 관리하는 Store 단계, 저장된 데이터를 분산 병렬 처리 프레임워크를 이용하여 군집과 예측 같은 분석을 하는 Analysis 단계, 그리고 얻어진 결과를 그래프와 같은 시각화 방법을 이용하여 공유하는 Reporting/Searching 단계를 진행한다. 본고에서는 빅데이터 분석을 위한 기계학습의 이해를 위해 다양한 Store단계의 플랫폼과 Analysis 단계의 프레임워크을 활용하는 분산 및 병렬화 관점의 스케일러블 기계학습 방법을 살펴보았다. 스케일러블 기계학습은 아직까지 특수한 경우의 알고리즘과 Dataset 분석에만 제한적으로 적용되지만, 최근에는 다양한 높은 연산 복잡도의 알고리즘이나 높은 차원의 Dataset 적용 같은 범용적인 활용을 위해 병렬 연산, 공유 메모리, 분산 처리와 통신 비용 관점의 최적화 연구들이 진행 중이다. 더불어 제한된 자원을 활용하여 빅데이터로부터 최적의 결과를 만들 수 있는 다양한 기계학습들을 평가하는 연구들이 진행되고 있다. 앞으로 이러한 기계학습의 발전은 M2M과 IoT과 같은 통신 패러다임의 변화와 함께 빅데이터를 활용한 높은 수준의 새로운 가치를 창출할 것으로 예상된다.

Acknowlegement

본 연구는 미래창조과학부 및 정보통신기술연구진흥센터

의 정보통신 • 방송연구개발사업[14-824-09-014, 인간 수준의 평생 기계학습SW 기초 연구(기계학습연구센터)와 2014년도 정부(미래창조과학부)의 재원으로 한국연구재단(No. NRF-2011-0009963, No. NRF-2012-R1A2A4A01008475) 및 2014년 두뇌한국21플러스 사업의 지원을 받아 수행하였음.

참 고 문 헌

[1] Seoul Metropolitan Government, News in Seoul: Utilizing Big Data (2014), Retrieved Oct., 1, 2014 from http://www.seoul.go.kr.

[2] Mitchell T. Machine learning, McGraw Hill, 1997

[3] R. Bekkerman et al., Scaling up machine learning: Parallel and distributed approaches, Cambridge University Press, 2011.

[4] S. Pan and Q. Yang, "A survey on transfer learning,"IEEE Transactions on Knowledge and Data Engineering, vol. 22, pp. 1345-1359, 2010.

[5] E. Alpaydin, Introduction to machine learning, MIT press, 2004.

[6] Y. Abu-Mostafa et al., Learning from data, AMLBook, 2012.

[7] Phil. Simon, Too Big to Ignore: The Business Case for Big Data, John Wiley & Sons, 2013.

[8] Samuel, "Some Studies in Machine Learning Using the Game of Checkers," IBM Journal of Research and Development, vol. 3(3), pp. 210-219, July 1959.

[9] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain," Psychological review, vol. 65, pp. 386, 1958.

[10] M. Minsky and S. Papert, Perceptrons - Expanded Edition: An Introduction to Computational Geometry, MIT press, 1987.

[11] Winston, Patrick, "Learning structural descriptions from examples," Cambridge Project Mac, No. MAC-TR-76, 1970.

[12] J. Holland, "Genetic algorithms and the optimal allocation of trials," SIAM Journal on Computing, vol. 2, pp. 88-105, 1973.

[13] J. Anderson and T. Mitchell et al., Machine learning: An artificial intelligence approach, vol. 2, Morgan Kaufmann, 1986.

[14] A. Barto, Reinforcement learning: An introduction, MIT press, 1998.

[15] Manyika et al. "Big data: The next frontier for innovation, competition, and productivity," Report in McKinsey Global Institute, May, 2011.

[16] A. Kulesza and F. Pereira, "Structured learning with approximate inference," Advances in neural information processing systems, pp. 785-792, 2007.

[17] R. Agrawal et al., "Mining association rules between sets of items in large databases," ACM SIGMOD Record, pp. 207-216, 1993.

[18] X. Wu et al., "Top 10 algorithms in data mining," Knowledge and Information Systems, vol. 14, pp. 1-37, 2008.

[19] O. Chapelle et al., Semi-supervised learning, vol. 2, MIT press Cambridge, 2006.

[20] C. Watkins and P. Dayan, "Q-learning," Machine learning, vol. 8, pp. 279-292, 1992.

[21] G. Tesauro, "Temporal difference learning and TD-Gammon," Communications of the ACM, vol. 38, pp. 58-68, 1995.

[22] Kurniawati et al., "SARSOP: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable Belief Spaces," Robotics: Science and Systems, vol. 2008, 2008.

[23] J. Cheng and R. Greiner, "Learning bayesian belief network classifiers: Algorithms and system," in Advances in Artificial Intelligence, pp. 141-151, 2001.

[24] Rajaraman et al., Mining of massive datasets, Cambridge University Press, 2011.

[25] Bishop, M. Christopher, Pattern recognition and machine learning, springer, 2006.

[26] Scholkopf et al., Learning with kernels: support vector machines, regularization, optimization, and beyond, MIT press, 2001.

[27] J. Dean et al., "Large scale distributed deep networks," Advances in Neural Information Processing Systems, pp. 1223-1231, 2012.

[28] Dietterich, G. Thomas, "Ensemble methods in machine learning," Multiple classifier systems, pp. 1-15, 2000.

[29] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," Communications of the ACM, vol. 51, pp. 107-113, 2008.

[30] Y. Yu et al., "DryadLINQ: A System for General-Purpose Distributed Data-Parallel Computing Using a High-Level Language," OSDI, pp. 1-14, 2008.

[31] Che et al., "A performance study of general-purpose applications on graphics processors using CUDA," Journal of parallel and distributed computing, no. 10, pp. 1370-1380, 2008.

[32] Wu et al., "GPU-accelerated large scale analytics," IACM UCHPC, 2009.

[33] Zaki et al., Large-scale parallel data mining, Springer, 2000.

[34] K.H. Lee et al., "Parallel data processing with MapReduce: a survey," ACM SIGMOD Record, vol. 40, pp. 11-20, 2012.

[35] Kosala et al., "Web mining research: A survey," ACM Sigkdd Explorations Newsletter 2.1, pp. 1-15, 2000.

[36] G. Malewicz et al., "Pregel: a system for large-scale graph processing," Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, pp. 135-146, 2010.

[37] The Apache Foundation, Apache Pig (2014), Retrieved Oct., 1, 2014 from http://pig.apache.org.

[38] The Apache Foundation, Hadoop (2014), Retrieved Oct., 1, 2014 from http://hadoop.apache.org.

[39] The Apache Foundation, Spark (2014), Retrieved Oct., 1, 2014 from https://spark.apache.org.

[40] IBM, Parallel Machine Learning Toolbox (2014), Retrieved Oct., 1, 2014 from https://www.research.ibm.com/haifa/projects/verification/ml_toolbox/index.html.

[41] W. Gropp et al., Using MPI: portable parallel programming with the message-passing interface, vol. 1, MIT press, 1999.

[42] Chang, Edward Y. "PSVM: Parallelizing support vector machines on distributed computers," Foundations of Large-Scale Multimedia Information Management and Retrieval, pp. 213-230, 2011.

[43] Burges, JC. Christopher, "From ranknet to lambdarank to lambdamart: An overview," Learning 11, pp. 23-581, 2010.

[44] Pednault, PD. Edwin, "Transform Regression and the Kolmogorov Superposition Theorem," SDM, pp. 35-46. 2006.

[45] G. Hinton, "Learning multiple layers of representation," Trends in cognitive sciences, vol. 11, pp. 428-434, 2007.

[46] Chen et al., "Parallel spectral clustering in distributed systems," IEEE Transactions on Pattern Analysis and Machine Intelligence, no. 3, pp. 568-586, 2011.

[47] Böhm et al., "Robust information-theoretic clustering," ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 65-75, ACM, 2006.

[48] Dekel et al., "Optimal distributed online prediction using mini-batches," The Journal of Machine Learning Research 13, no. 1, pp. 165-202, 2012.

[49] Koren et al., "Matrix factorization techniques for recommender systems," Computer 42, no. 8, pp. 30-37, 2009.

[50] The Apache Foundation, Mahout (2014), Retrieved Oct., 1, 2014 from http://mahout.apache.org.

[51] Bu et al., "The HaLoop approach to large-scale iterative data analysis," The VLDB Journal—The International Journal on Very Large Data Bases 21, no. 2, pp. 169-190, 2012.

[52] Y. Low et al., "Graphlab: A new framework for parallel machine learning," arXiv preprint arXiv:1006.4990, 2010.

[53] A. Ghoting et al., "SystemML: Declarative machine learning on MapReduce," Data Engineering (ICDE), pp. 231-242, 2011.

[54] Google, Google Prediction API (2014), Retrieved Oct., 1, 2014 from https://developers.google.com/prediction.

약 력

이 재 구

2009년 숭실대학교 공학사
2011년 UC San Diego 공학석사
2011년~2013년 LG전자 CTO부문 주임연구원
2013년~현재 서울대학교 전기정보공학부 박사과정
관심분야: 빅데이터 분석, 기계학습, 시계열 분석, 인과성 분석

이 태 훈

2009년 고려대학교 공학사
2012년 고려대학교 공학석사
2012년~현재 서울대학교 전기정보공학부 박사과정
관심분야: 대용량 바이오 컴퓨팅, 확률 통계 이론

윤 성 로

1996년 서울대학교 공학사
2002년 Stanford University 공학석사
2006년 Stanford University 공학박사
2006년 Stanford University 박사후 연구원
2006년~2007년 미국 인텔 선임연구원
2007년~2012년 고려대학교 전기전자전파공학부 조교수
2012월~현재 서울대학교 전기정보공학부 부교수
관심분야: 기계학습, 생체정보, 빅데이터 분석, 고성능 컴퓨팅 및 스토리지