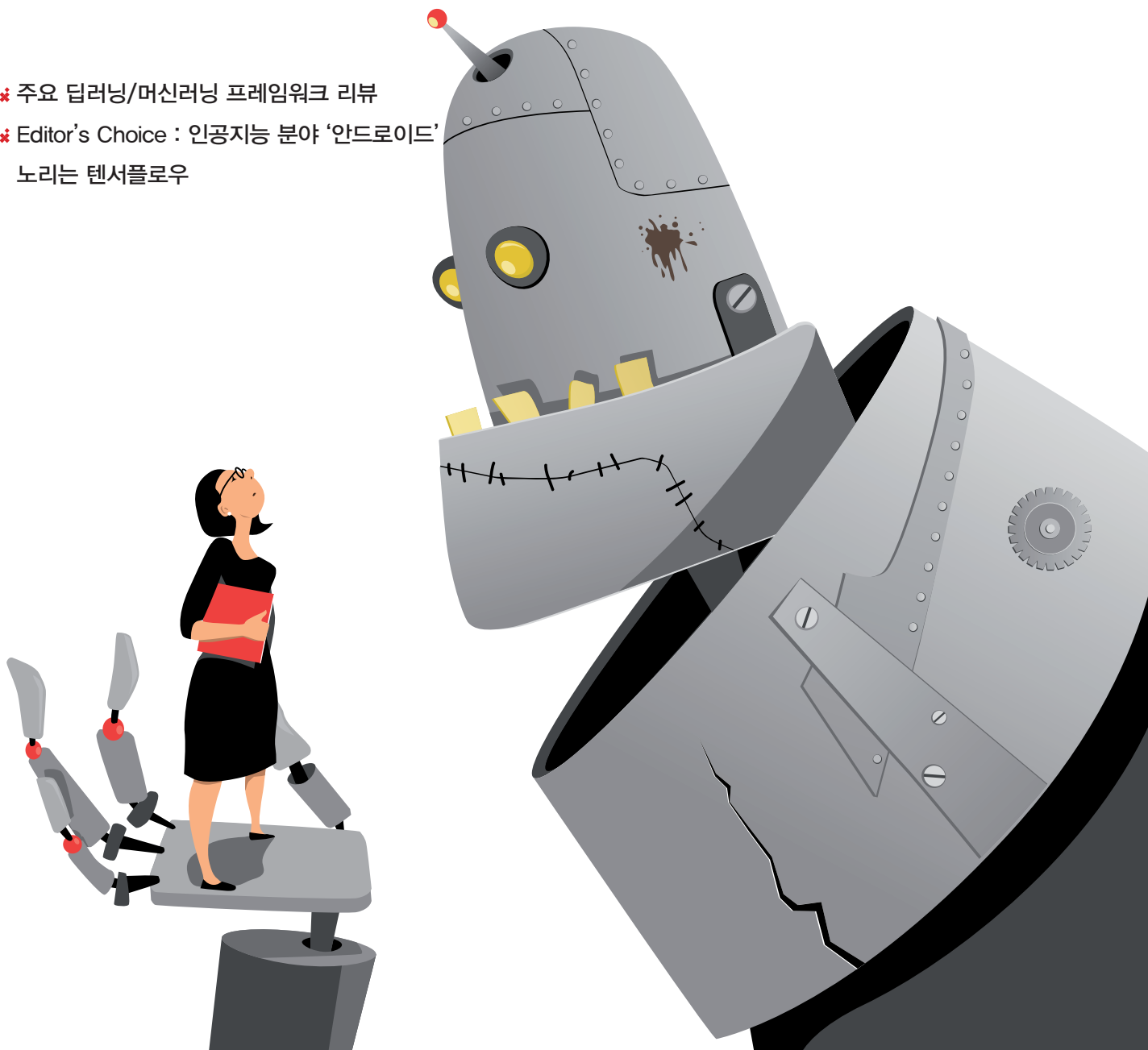


IDG Tech Review

구글 텐서플로우부터 MS CNTK까지 딥러닝/머신러닝 프레임워크 6종 비교 분석

머신러닝과 딥러닝에 대한 열기가 뜨겁게 달아오르고 있다. 그러나 아직 관련 기술이 초기여서 다양한 프레임워크가 난립하고 있다. 이 영역에 처음 발을 내딛는 사람에게는 어떤 프레임워크를 선택할 것인지부터 혼란스럽다. 여기 현재 가장 주목받고 있는 딥러닝/머신러닝 프레임워크 6종을 모아 비교 분석했다. 카페(Caffe)와 마이크로소프트 인지 툴킷(CNTK), MX넷, 텐서플로우(TensorFlow), 사이킷런(Scikit-learn), 스파크 MLlib 등이다. 어느 정도 평가 결과가 예상되기는 하지만 이들 간의 차이를 아는 것도 머신러닝 전문가를 향한 꽤 멋진 첫걸음이 될 것이다.

- ✘ 주요 딥러닝/머신러닝 프레임워크 리뷰
- ✘ Editor's Choice : 인공지능 분야 '안드로이드' 노리는 텐서플로우



무단 전재 재배포 금지

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

구글 텐서플로우부터 MS CNTK까지 딥러닝/머신러닝 프레임워크 6종 비교 분석

Martin Heller | Infoworld

머신러닝(Machine Learning) 프레임워크와 딥러닝(Deep Learning) 프레임워크 간에는 차이가 있다. 머신러닝 프레임워크는 분류, 회귀(Regression), 클러스터링, 비정상행위 탐지(Anomaly Detection), 데이터 준비(Data Preparation)를 위한 다양한 학습 방법을 다루며, 인공 신경망 메소드(Method)를 포함할 수도, 포함하지 않을 수도 있다. 반면 딥러닝 또는 심층 신경망(Deep Neural Network: DNN) 프레임워크는 여러 개의 은닉 계층(Hidden Layer)을 가진 다양한 신경망 토폴로지를 다룬다. 이런 계층은 다단계 프로세스의 패턴 인식으로 이루어져 있다. 망에 계층이 많을수록 클러스터링과 분류를 위해 추출할 수 있는 특징이 더 다양해진다.

대표적인 프레임워크를 꼽아보면, 먼저 카페(Caffe), 마이크로소프트 인지 툴킷(Cognitive Toolkit: CNTK 2)과 딥러닝4j(하둡과 스파크에서 사용하는 자바와 스칼라(Scalar)용 딥러닝 소프트웨어), 케라스(Keras: 테아노와 텐서플로우용 딥러닝 프론트엔드), MX넷, 텐서플로우(TensorFlow) 등은 딥러닝 프레임워크이다. 사이킷런(Scikit-learn)과 스파크(Spark) MLlib는 머신러닝 프레임워크이다. 테아노(Theano: 10년 된 파이썬 딥러닝과 머신러닝 프레임워크)는 두 범주 모두에 걸쳐있다.

일반적으로, 심층 신경망 연산(Computation)은 CPU보다는 엔비디아 CUDA 범용 GPU 등 GPU에서 더 빠른 속도로 구동한다. 한 개 이상의 CPU에서 DNN을 훈련할 수도 있지만, 속도가 떨어지는 경향이 있다. 여기서 속도가 떨어진다는 것은 몇 초 또는 몇 분의 이야기가 아니다. 훈련할 뉴런(Neuron)과 계층 수가 많을수록, 그리고 훈련을 위해 사용하는 데이터가 많을수록, 시간이 급속히 늘어난다. 2016년에 구글 브레인팀이 새 버전의 언어 번역 모델을 훈련할 때 여러 개의 GPU 상에서 한 번에 일주일 동안 훈련 세션을 운영한 바 있다. GPU가 없었다면 각 모델 실험 훈련에만 몇 개월 걸렸을 것이다.

각 패키지는 최소한 하나씩 자신만의 특징을 갖고 있다. 예를 들어 카페는 이미지 인식에 대한 컨볼루션(Convolutional) DNN으로 유명하다. CNTK는 ASP닷넷 웹사이트 상에서 동작하는 예측 모델을 배포하는 별도의 평가 라이브러리가 널리 알려져 있다. MX넷은 다중 GPU와 다중 머신 구성에서의 훈련에서 탁월한 확장성을 갖고 있다. 사이킷런은 다양하고 탄탄한 머신러닝 메소드를 가지고 있으며 배우고 사용하기 쉽다. 스파크 MLlib는 하둡과 통합돼 머신러

닝에서 훌륭한 확장성을 갖는다. 텐서플로우는 네트워크 그래프인 텐서보드를 위한 고유의 진단 기능이 좋은 평가를 받고 있다.

한편, GPU 상에서 모든 딥러닝 패키지의 훈련 속도는 거의 같다. 훈련 내부 루프(Inner Loop)의 대부분이 공통적으로 엔비디아 CuDNN 패키지에서 처리되기 때문이다. 그렇기는 하지만 각 패키지는 신경망을 정의하는 방법에 있어서 조금 다른 접근방식을 취하고 있으며 크게 두 진영이 있다. 그래프 정의 파일(Graph Description File)을 사용하는 진영과 코드 실행을 통해 자신의 정의를 생성하는 진영이다. 이를 염두에 두고 각 프레임워크에 대해 자세히 살펴보자.

카페

카페 딥러닝 프로젝트는 본래 이미지 분류를 위한 강력한 프레임워크를 만들기 위해 시작됐다. 그러나 설립자가 프로젝트를 떠났고 버전 1.0 RC3 상태에서 1년 이상 변화가 없다. 더구나 끊임없이 반복되는 버그까지 고려하면 사실상 교착 상태에 빠진 것으로 보인다. 카페는 간단한 망 정의 포맷뿐 아니라 이미지 인식에 대한 훌륭한 컨볼루션 망이 있고 엔비디아 CUDA GPU에 대한 지원도 여전히 훌륭하다. 그러나 1GB 이상 GPU 메모리가 필요하고 문서화에 허점이 많다. 지원을 받기 힘들고 설치 과정도 매끄럽지 못하다. 특히, 파이썬 노트북 지원에 있어서 그렇다.

카페는 명령어 줄과 파이썬, 매트랩(Matlab) 인터페이스를 가지고 있다. 모델과 솔버(Solver) 정의는 프로토텍스트(ProtoText)에 의존한다. 카페는 자체 모델 스키마(Schema)에서 망을 한 계층씩 정의한다. 망은 입력 데이터부터 손실까지 전체 모델을 철저히 정의한다. 데이터와 도함수(Derivative)가 전방

과 후방 단계로 망을 통과함에 따라 카페는 내부적으로는 C-연속적인(C-contiguous) 방식(어레이의 열이 C 언어처럼 연속적인 메모리 블록에 저장된다)으로 저장된 N-차원 어레이인 블롭(Blob: Binary Large Object)으로 정보를 저장, 통신, 조작한다. 카페에서 블롭은 텐서플로에서 텐서(Tensor)와 같다.

계층은 블롭에 대한 연산을 수행하며 카페 모델을 구성한다. 계층은 필터를 처리하고 풀링(Pooling)을 수행하며 내적(Inner Product)을 계산한다. 또한, 정류선형 변환과 시그모이드(sigmoid) 변환, 다른 엘리먼트 단위(element-wise) 변환 같은 비선형계수를 적용해 정규화하고 데이터를 로드하며, 소프트맥스(softmax)와 힌지(hinge) 같은 손실을 계산한다. 카페는 이미지 분류 작업에서 그 효율성을 입증했지만 이미 전성기를 지났다는 평가이다. 앞으로 사용자의 필요사항에 적합하거나 목적에 맞게 개선되지 않는다면 텐서플로나 MX넷, CNTK를 사용할 것을 권한다.

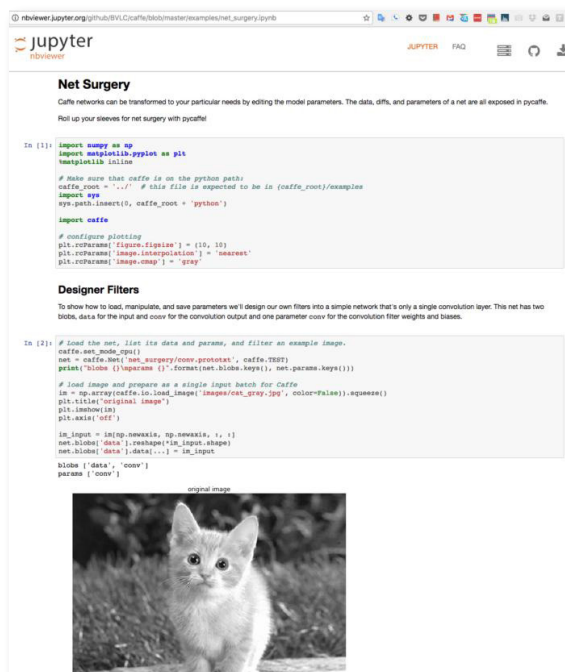


그림 1 | 사전 계산한 결과를 NB 뷰어에 표시한 카페 주피터 노트북(Jupyter Notebook)

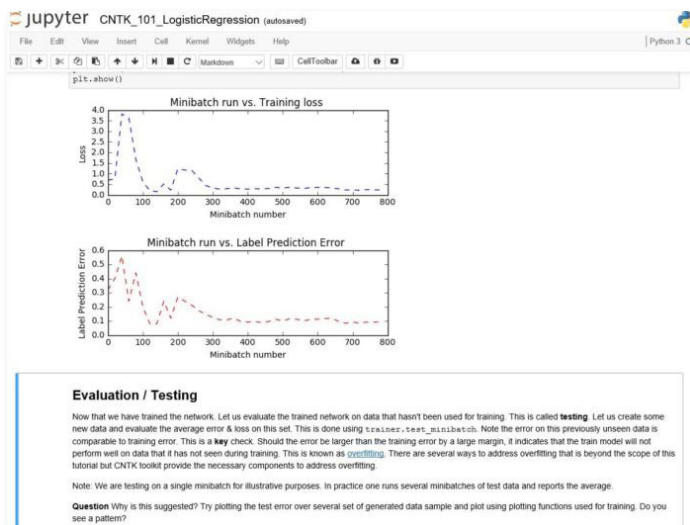


그림 2 | 몇 가지 CNTK 사용 설명서가 주피터 노트북으로 제공된다. 이것은 로지스틱 회귀(Logistic Regression) 학습용으로 그린 시각화 사례이다.

마이크로소프트 CNTK

마이크로소프트 CNTK는 빠르고 사용하기 쉬운 딥러닝 패키지이지만, 텐서플로우와 비교하면 그 범위가 제한적이다. 그러나 모델과 알고리즘이 매우 다양하고 파이썬과 주피터 노트북에 대한 지원이 풍부하며 흥미로운 브레인스크립트(BrainScript) 신경망 구성 언어를 가지고 있다. 윈도우와 우분투 리눅스에 대한 자동 배포도 훌륭하게 지원한다. 단점도 있다. 필자가 베타 1을 리뷰했을 당시에는 문서화가 아직은 CNTK 2로 완벽하게 업데이트되지 않았고 패키지가 맥OS를 전혀 지원하지 않았다. 베타 1 이후 GPU 메모리 사용량을 줄

이기 위한 새로운 메모리 압축 모드를 포함해 여러 가지 개선이 있었다. 그러나 맥OS는 여전히 지원하지 않는다.

베타 1에 추가된 파이썬 API는 CNTK가 파이썬으로 코드를 작성하는 딥러닝 연구자의 '대세'가 되는 데 큰 도움이 됐다. 이 API에는 모델 정의와 컴퓨트, 학습 알고리즘, 데이터 읽기(Data Reading), 분산 훈련에 대한 추상화(Abstraction) 등이 포함돼 있다. CNTK 2에서는 파이썬 API가 더 보완돼 구글의 프로토콜 버퍼 직렬화(Protocol Buffers Serialization)를 지원하고 새로운 파이썬 예제와 사용 설명서가 포함돼 있다. 이 설명서는 주피터 노트북으로 구현됐다.

CNTK 2 구성 요소는 파이썬, C++ 또는 브레인스크립트로부터의 다차원 고밀도(Dense)/저밀도(Sparse) 데이터를 처리할 수 있다. CNTK에는 다양한 신경망 유형이 포함돼 있는데, FFN(FeedForward: 피드포워드 신경망), CNN(Convolutional: 컨볼루션 신경망), RNN/LSTM(Recurrent/Long Short Term Memory: 순환 신경망/LTSM), 배치 정규화(Batch Normalization), 시퀀스-투-시퀀스(Sequence-to-Sequence) 등이 대표적이다. CNTK 2는 강화 학습(Reinforce Learning), GAN(Generative Adversarial Network: 대립쌍 구조를 사용하는 생성모델), 지도(Supervised)와 자율(Unsupervised) 학습, 자동 하이퍼파라미터 튜닝(Automatic Hyperparameter Tuning), 파이썬에서 GPU 상에 새로운 사용자 정의 핵심 구성요소를 추가하는 기능 등을 지원한다. GPU와 머신(Machine)에서 정확하게 병렬 처리를 할 수 있으며, (마이크로소프트의 주장에 따르면) 가장 큰 모델까지도 GPU 메모리에 맞춰 넣을 수 있다.

CNTK 2 API는 망과 리너(Learner), 리더(Reader), 훈련을 정의할 수 있고, 파이썬과 C++, 브레인스크립트, C#로부터의 평가도 지원한다. 파이썬 API에는 넘파이(NumPy)와 호환되면서 재귀(Recurrence)를 포함해 고급 신경망을 간결하게 정의할 수 있는 고급 계층 라이브러리가 들어있다. 이 툴킷은 순환 모델을 표현할 때 재귀 단계의 정적 펼침(static Unrolling) 대신 사이클로서 기호

형태로 표현할 수 있게 지원한다. 사용자는 애저 네트워크와 GPU에서 CNTK 2 모델을 훈련할 수 있다. GPU를 갖춘 N 시리즈 패밀리의 애저 가상머신(Virtual Machine)은 필자가 베타 1을 리뷰했을 당시에는 제한적이었으나, 이제는 누구나 사용할 수 있으며 애저 콘솔에서 완전히 관리할 수 있다.

MX넷

MX넷은 아마존이 선택한 DNN 프레임워크로, 이식성이 좋고 확장 가능한 딥러닝 라이브러리이다. 신경망 형상(Geometry)의 심볼 선언(Symbolic Declaration)을 텐서 연산(Tensor Operation)의 명령형 프로그래밍과 결합한다. MX넷은 85%의 준 선형 확장 효율성으로 여러 대의 호스트에 걸쳐 다중 GPU로 확장할 수 있고, 탁월한 개발 속도와 프로그램 용이성, 이식성을 갖고 있다. 다양한 수준으로 파이썬과 R, 스칼라(Scala), 줄리아(Julia), C++ 등을 지원하며, 사용자가 심볼릭 프로그래밍과 명령형 프로그래밍을 필요에 따라 섞어 사용할 수 있도록 지원한다.

필자가 MX넷을 검토했을 때는 문서화가 완료되지 않았고 파이썬 이외의 언어로 된 몇 가지 예제가 섞여 있었다. 그러나 이후 관련 상황이 모두 개선됐다. MX넷 플랫폼은 사용자가 MX넷에게 어떤 GPU와 CPU 코어를 사용하라고 일일이 지정해야 작동한다. 그러나 심볼릭 연산과 명령형 연산 모두를 자동으로 즉시 병렬처리하는 역동적인 디펜던시 스케줄러(Dynamic Dependency Scheduler)를 기반으로 구축됐다. 또한, 스케줄러 상단의 그래프 최적화 계층은 심볼 실행을 빠르게 해주고 메모리를 효율적으로 만들어 준다.

MX넷은 현재 파이썬과 R, 스칼라, 줄리아, C++를 사용한 모델 구축과 훈련을 지원한다. 훈련된 MX넷 모델은 매트랩과 자바스크립트에서 예측용으로도 사용할 수 있다. 사용자가 모델 구축을 위해 어떤 언어를 선택하든 MX넷은 최

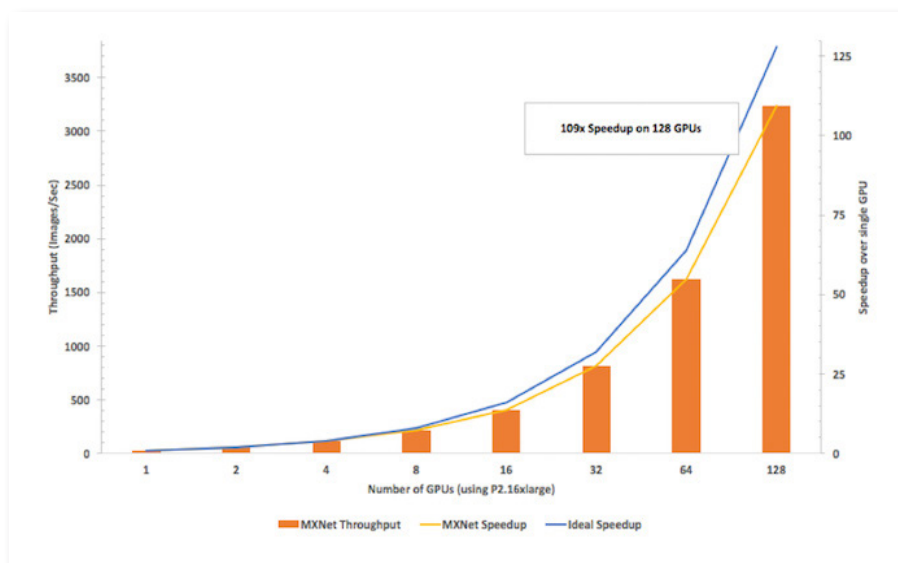


그림 3 | 아마존이 P2 상에서 MX넷으로 구현된 인셉션 V3(Inception V3) 알고리즘을 테스트한 결과이다. 85%의 확장 효율성을 확인할 수 있다.

적화된 C++ 백엔드 엔진을 호출한다. MX넷 개발자들은 이식성과 GPU 클러스터에 대한 지원이 매우 뛰어나다고 주장한다. MX넷 API가 토치(Torch)와 테아노, 체이너(Chainer), 카페에서 제공하는 API의 상위 집합이라는 것이다. 여러 가지 면에서 MX넷은 텐서플로우와 비슷하다. 그러나 명령형 텐서 연산을 내장할 수 있는 것이 특징이다.

MX넷 사용 설명서를 보면 실질적으로 의무적인 MNIST 숫자 분류(Digit Classification) 이외에, CNN(Convolutional Neural Network)을 활용한 이미지 분류와 분할(Segmentation), Faster R-CNN을 활용한 객체 탐지, 뉴럴 아트(Neural Art), 딥 CNN과 이미지넷(ImageNet) 데이터 세트를 활용한 대규모 이미지 분류 방법이 상세히 설명돼 있다. 이밖에 자연어 처리, 음성 인식, 대립 쌍 망(Adversarial Network), 지도 머신러닝과 자율 지도 머신러닝 두 가지 모두에 대한 추가 사용법도 포함돼 있다.

사이킷런

사이킷런 파이썬 프레임워크는 여러 가지 탄탄한 학습 알고리즘을 갖고 있어, 파이썬 팬에게는 머신러닝 라이브러리 중 최고의 선택일 것이다. 매우 다양하게 잘 정의된 알고리즘과 통합 그래픽을 가지고 있으며, 이미 검증된 라이브러리라는 것도 장점이다. 상대적으로 설치, 학습, 사용하기 쉽고 예제와 사용 설명서가 잘 돼 있다.

단점도 있다. 사이킷런은 딥러닝이나 강화 학습을 다루지 않으며, 그래픽 모델과 시퀀스 예측(Sequence Prediction) 기능을 지원하지 않는다. 또한, 파이썬 이외의 언어에서는 사용할 수 없고, 파이썬 JIT(Just-in-Time) 컴파일러인 파이파이(PyPy)나 GPU를 지원하지 않는다. 그러나 신경망에 대한 사소한 불편함 같은 일부 단점을 빼면 매우 만족스러운 속도를 보여준다. 내부 루프처럼 속도가 빨라야 하는 함수는 싸이썬(Cython: 파이썬-to-C 컴파일러)을 사용하면 된다.

사이킷런은 분류와 회귀, 클러스터링, 차원 축소(Dimensionality Reduction), 모델 선택, 전처리에 대해 다양한 알고리즘을 지원한다. 이와 관련된 문서화와 예제도 훌륭하다. 단, 이런 작업을 완료하기 위한 안내 워크플로우가 전혀 없다. 사이킷런의 주요 알고리즘은 설명서에 나온 대로 잘 동작했고, API가 일관성 있고 잘 설계되었으며, 데이터 구조 간에 임피던스 불일치(Impedance Mismatch)가 거의 없었다. 이번 리뷰에서 개발의 용이성 부문에서 최고의 점수를 받은 이유이기도 하다. 잘 정의된 장점과 버그 없는 라이브러리를 이용해 개발한다는 것은 언제나 즐거운 일이다.

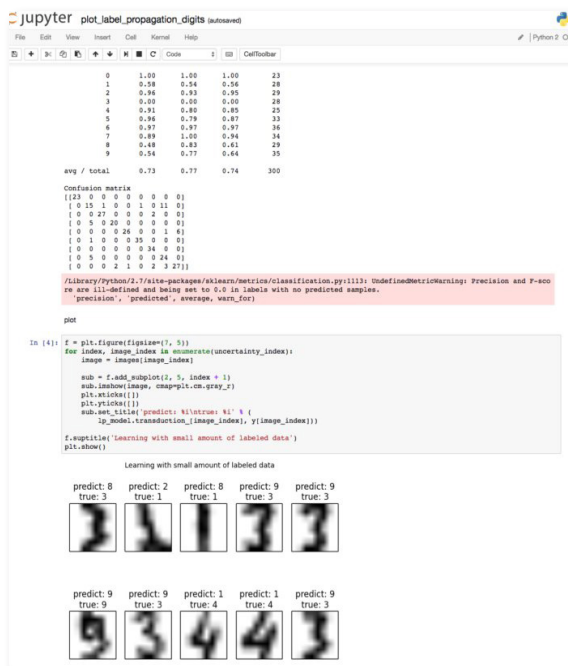


그림 4 | 벨 확산(Label Spreading) 모델을 사용하는 준 지도 학습 설명 예제. 전체 1,797개 샘플 중에서 30개에만 라벨이 붙여졌다.

이 라이브러리는 딥러닝이나 강화 학습을 지원하지 않아 정확한 이미지 분류와 신뢰성 있는 실시간 언어 구문 분석(Language Parsing), 번역 같은 문제를 해결하는 데는 적절치 않다. 특히 딥러닝에 더 관심이 있는 개발자라면 반드시 다른 프레임워크를 함께 살펴봐야 한다. 그러나 여러 가지 다른 관측값(Observation)을 연결하는 예측 함수를 만드는 것부터 관측값을 분류하는 것, 라벨이 붙어있지 않은 데이터 세트의 구조를 학습하는 것까지, 수십 개의 뉴런 계층이 필요 없는 일반적인 머신러닝 용도라면 사이킷런 만큼 훌륭한 프레임워크를 찾기도 힘들다.

스파크 MLlib

스파크용 오픈소스 머신러닝 라이브러리인 스파크 MLlib는 특징 추출(feature extraction), 변환, 차원 축소, 머신러닝 파이프라인 등을 구축, 평가, 조정하는 여러 가지 기능과 도구를 제공한다. 또한, 분류와 회귀, 클러스터링, 협업 필터링(Collaborative Filtering) (하지만 DNN은 아님) 같은 일반적인 머신러닝 알고리즘도 지원한다. 데이터 처리와 선형 대수, 통계 작업 알고리즘을 저장, 로드하는 유틸리티도 포함돼 있다. 스파크 MLlib는 스칼라로 작성됐으며 선형 대수 패키지인 브리즈(Breeze)를 사용한다. 브리즈는 최적화된 수치 처리를 위해 netlib-java를 활용한다

초보라자라면 MLlib를 이용해 분류와 회귀 작업을 하는 것이 다소 어려울 수 있다. 그러나 전문가라면 엄청나게 많은 공용 알고리즘과 모델에 놀랄 것이다. 웬만한 분석 대상 데이터에서는 꽤 적합한 모델을 찾을 수 있을 정도이다. 특히 스파크 2.x에서는 ‘모델 선정’이라고 알려진 하이퍼파라미터 튜닝 기

능이 추가됐다. 분석자가 파라미터 그리드와 에스티메이터(Estimator), 이밸류에이터(Evaluator)를 설정할 수 있으며, (시간이 걸리지만 정확한) 교차 검증 메소드 또는 (빠르지만 덜 정확한) 훈련 검증 스플릿(Split) 메소드를 이용해 해당 데이터에 대한 최상의 모델을 찾을 수 있다.

스파크 MLlib는 스칼라와 자바용 모든 API를, 파이썬용 거의 모든 API를 지원한다. 단, R에 대해서는 개략적이고 부분적인 API만 가지고 있다. 예제 수를 세어보면 지원 범위를 잘 알 수 있는데, 자바는 54개, 스칼라는 60개, 파이썬은 52개이다. 반면 R 머신러닝 예제는 5개에 불과하다. 필자는 주피터 노트북을 사용해서 작업하기가 가장 쉬웠지만 장황한 스파크 상태 메시지에 익숙해지면 당연히 콘솔에서도 사

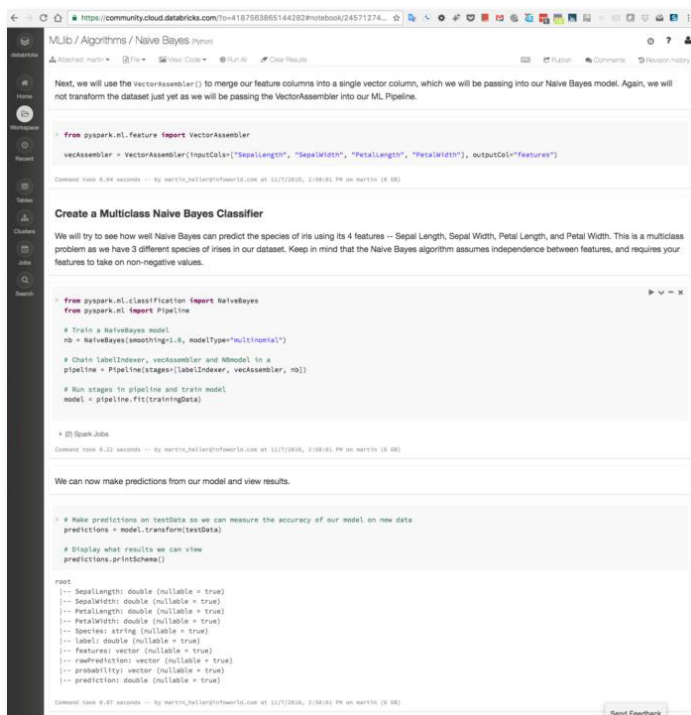


그림 5 | 스파크 MLlib 파이썬 예제

용할 수 있다.

스파크 MLlib는 기본적인 머신러닝 작업과 특징 선택, 파이프라인, 지속성(Persistence) 등 원하는 거의 모든 것을 제공한다. 특히 분류와 회귀, 클러스터링, 필터링 기능이 훌륭하다. 스파크 MLlib는 스파크의 일부이므로 데이터베이스와 스트림, 다른 데이터 소스에 대한 액세스 기능이 뛰어나다. 반면 딥 신경망을 모델링하거나 훈련하는 측면에서는 텐서플로우, MX넷, 카페, 마이크로소프트 CNTK에 미치지 못한다.

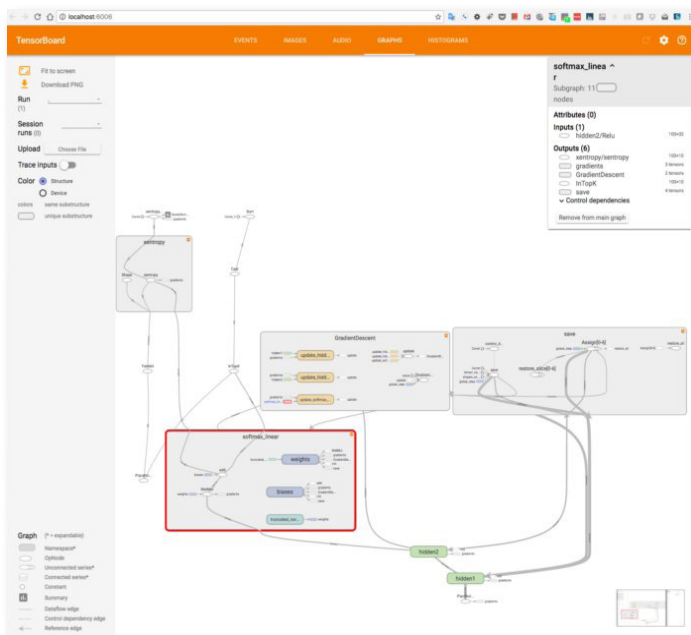


그림 6 | 텐서플로우 계산 결과를 그래프로 보여주는 텐서보드

텐서플로우

텐서플로우는 구글이 내놓은 이식성 좋은 머신러닝과 인공 신경망 라이브러리이다. 배우기가 조금 어렵지만 성능과 확장성이 좋다. 텐서플로우에는 딥러닝에서 많이 사용하는 다양한 모델과 알고리즘이 들어 있으며 GPU(훈련용)나 구글 TPU(현업에 적용할 수 있는 규모로 예측용)를 장착한 하드웨어에서 탁월한 성능을 보인다. 파이썬 지원이 훌륭하며 문서화가 잘돼 있고, 특히 텐서보드라고 하는 소프트웨어가 포함되어 있어 결과를 설명하는 데이터 플로우 그래프(Data Flow Graph: DFG)를 표시하고 이해하기 좋다.

데이터 플로우 그래프에서 노드(Node)는 수학적 연산을 나타내며, 그래프 에지(Edge)는 노드 간을 흐르는 다차원 데이터 어레이(텐서)를 나타낸다. 이런 매우 유연한 아키텍처가 적용돼 있어 사용자가 코드를 재작성하지 않고도 데스크톱과 서버, 모바일 기기에 있는 하나 또는 이 이상의 CPU나 GPU에 배포할 수 있다.

텐서플로우를 사용하기 위한 주요 언어는 파이썬이다. C++에 대한 지원은 일부 제한이 있다. 텐서플로우와 함께 제공되는 사용 설명서를 보면, 수기 숫자 분류와 이미지 인식, 워드 임베딩(Word Embedding: 단어 표현), RNN(Recurrent Neural Network: 순환 신경망), 기계 번역(Machine Translation)을 위한 시퀀스-투-시퀀스(Sequence-to-Sequence) 모델, 자연어 처리, 그리고 PDE(Partial Differential Equation: 편 미분 방정식) 기반의 시뮬레이션에 대한 애플리케이션 등이 포함돼 있다.

텐서플로우를 이용하면 현재 이미지 인식과 언어 처리 분야를 바꿔놓고 있는 딥 CNN과 LSTM 재귀 모델을 포함해 온갖 종류의 신경망을 쉽게 처리할 수 있다. 계층을 정의하는 코드가 다소 복잡하지만 3가지 딥러닝 인터페이스 옵션 중 한 가지로 이 불편함을 해결할 수 있다. 비동기 네트워크 솔버(Asynchronous

Network Solver)를 디버깅하기가 만만치 않지만 텐서보드 소프트웨어는 사용자가 그래프를 시각화할 수 있게 해줘 도움이 된다.

최고의 프레임워크


지금까지 주요 프레임워크를 살펴봤다. 그러나 예측 작업에 어떤 머신러닝 또는 딥러닝 패키지를 선택할 것인지는 필요한 머신러닝의 복잡성과, 훈련을 위해 확보한 데이터의 양과 유형, 컴퓨팅 자원, 선호하는 프로그래밍 언어와 숙련도에 따라 달라진다. 사용할 모델을 정의하기 위해 코드나 구성 파일 중 어느 것을 더 선호하는지도 중요한 변수이다.

만약 스칼라를 선호하고 데이터를 하둡에 저장했다면 스파크 MLlib를 권장한다. 또한, 파이썬을 사용한다면 사이킷런을 권한다. 스칼라와 자바를 선호하고 데이터를 하둡에 저장하고 있다면 딥러닝4j도 좋은 선택이다.

카페와 마이크로소프트 CNTK, MX넷, 텐서플로우 중에서 딥러닝 패키지를 선택하는 것이 더 어려운 결정이다. 일단 카페는 개발이 정체 상태이므로 더는 권장하지 않는다. 다른 3가지는 모두 비슷한 기능을 가지고 있는 훌륭한 플랫폼이어서 이 중 선택하는 것은 다소 까다로울 수 있다.

먼저 마이크로소프트 CNTK는 망 구성 언어인 브레인스크립트뿐 아니라 파이썬과 C++ API도 지원한다. 네트워크 토폴로지를 프로그래밍하는 것보다 구성 파일을 사용하는 것을 선호한다면 CNTK가 좋은 선택이다. 단, 완성도 측면에서 텐서플로우 수준에 이르지 못했고 맥OS에서 쓸 수 없다는 것을 참고해야 한다. MX넷은 파이썬, R, 스칼라, 줄리아, C++를 지원하지만, 가장 잘 지원

하는 API는 파이썬이다. MX넷은 여러 대의 호스트에 있는 여러 GPU에 걸친 훌륭한 확장성(85%의 선형 확장)을 입증했다. 필자가 MX넷을 리뷰했을 때는 문서화와 예제가 너무 적었으나 이후 개선됐다.

3가지 패키지 중에서는 가장 완성도가 높은 것은 텐서플로우이다. 파이썬으로 코드를 작성하기 좋아하고 초기 학습을 기꺼이 감수할 수 있다면 텐서플로우가 훌륭한 선택이다. 또한, 사용할 수 있는 기본 구성 요소가 충실하고 정밀한 제어 기능을 제공한다. 단, 신경망을 정의하기 위해 많은 양의 코드를 작성해야 한다. 이를 더 쉽게 하려면 `tf.contrib.learn`, `TF-Slim`, 케라스 등 텐서플로우용 3가지 API를 사용하는 것도 방법이다. 텐서플로우의 또 다른 장점 중 하나는 텐서보드이다. 데이터 흐름도(Data Flow Diagram: DFD)를 시각화하고 이해하는 데 매우 유용하다. 

Infoworld scorecard	Models and algorithms (25%)	Ease of development (25%)	Documentation (20%)	Performance (20%)	Ease of deployment (10%)	총점 (100%)
카페 (Caffe 1.0 RC3)	8	8	7	9	8	8.0
마이크로소프트 인지툴킷 (Microsoft Cognitive Toolkit v3.0 Beta 1)	8	9	8	10	9	8.8
MX넷 (MXNet v0.7)	8	8	7	10	8	8.2
사이킷런 (Scikit-learn 0.18.1)	9	9	9	8	9	8.8
스파크 MLlib (Spark MLlib 2.01)	9	8	8	9	8	8.5
텐서플로우 (TensorFlow r0.10)	9	8	9	10	8	8.9

Editor's
Choice

텐서플로우, 인공지능 분야 ‘안드로이드’ 노린다

Steven Max Patterson | Network World

머신러닝은 자바와 C#, 파이썬과 같은 언어로 프로그램을 만드는 대신 데이터베이스를 사용해 높은 신뢰성으로 예측을 수행하는 컴퓨터를 프로그래밍하는 방법이다. 이미지 인식과 언어 번역, 코멘트나 추천 등의 순위 매기기와 같은 작업에 효율적이다. 구글을 비롯해 페이스북, IBM, 마이크로소프트는 검색 결과 순위 결정 등에 내부적으로 이미 머신러닝을 사용하고 있다.

구글은 약 1년 전에 독자적인 머신러닝 라이브러리인 ‘디스트빌리프(DistBelief)’를 기반으로 텐서플로우(TensorFlow)를 내놓았다. 자체적으로 만든 모델 설명 파일을 포함해 약 4,000개의 소스 코드 디렉터리가 포함돼 있고, 그 분야도 구글 검색, 지도, 지메일 스팸 필터 등 다양하다.

구글의 선임 연구원이자 인공지능 분야의 슈퍼스타인 제프 딘에 따르면, 텐서플로우는 깃허브에서 가장 인기 있는 머신러닝 리포지토리이다. 실제로 딘이 언급한 몇 가지 수치를 보면, 500명의 독립 프로그래머가 소프트웨어를 개선해 제출했고, 월별 커밋(새 코드 모듈 및 패치 적용) 수는 1,000회이며 5,500개의 독립적인 깃허브 리포지토리가 있다. 토론토, 버클리, 스탠퍼드 대학은 머신러닝 교육 과정의 기본 프레임워크로 텐서플로우를 사용한다.

독창적인 머신러닝 모델을 만들려면 선형 대수와 확률, 머신러닝 분야를 전공한 고급 개발자가 필요하지만, 이러한 조건을 충족하는 개발자는 얼마 되지 않

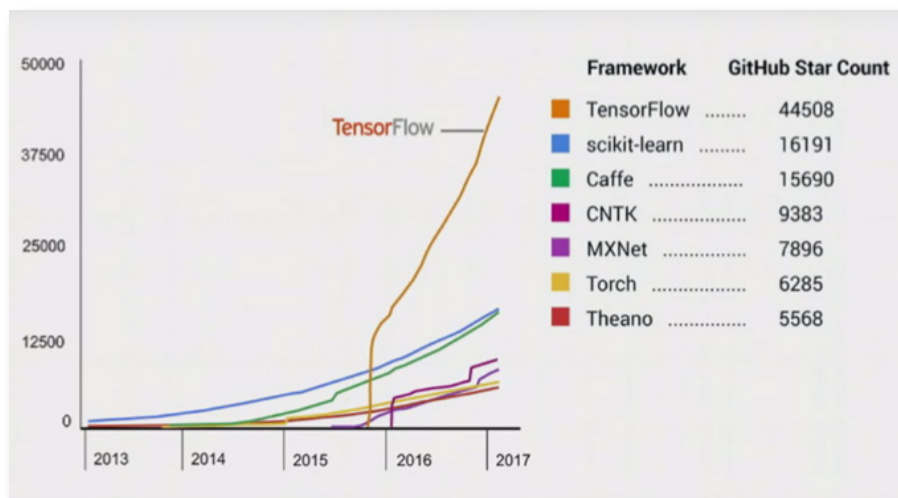


그림 1 | 깃허브 활동으로 본 텐서플로우 성장 추세

는다. 다만 이미 만들어진 머신러닝 모델을 사용하는 것은 다른 개발자들도 가능하다. 이처럼 각기 다른 기술을 보유한 개발자가 수행할 수 있는 머신러닝 관련 작업을 정리하면 다음 3가지이다.

전문가 : 예를 들어 인간보다 아타리 딥 큐(Atari Deep Q)를 더 잘 플레이하도록 학습할 수 있는 모델과 같은 오리지널 머신러닝 모델을 만들려면 선형 대수와 확률, 머신러닝에 대한 전문적인 기술을 보유한 개발자가 필요하다.

분야 전문가 : 두 번째 그룹은 인셉션(Inception) V3 이미지 인식과 같은 기존 모델을 가져와서 분야별 전문 지식을 활용해 자신의 분야에서 사용할 수 있는 개발자이다. 대표적인 사례가 헬스케어이다. 인셉션 V3은 이미지 내의 여러 가지 다양한 사물을 인식하며, 새로운 유형의 이미지를 인식하도록 재교육할 수 있다.

스탠퍼드대학의 박사과정생인 브렛 쿠프렐은 피부과 전문의와 대등한 정확성으로 피부암에서 양성 병변 이미지를 찾아내고, 양성 피부암과 악성 피부암을 구분하는 인셉션 V3 애플리케이션을 발표했다. 이 내용은 네이처(Journal Nature) 지에 게재되기도 했다. 구글의 릴리 핑은 미국 의학 협회 저널(Journal of the American Medical Association)에 게재된 또 다른 인셉션 V3 애플리케이션을 발표했는데, 이를 이용하면 시각 상실의 주원인인 당뇨병 망막증을 인간 안과의사보다 약간 더 높은 정확성으로 진단할 수 있다.

애플리케이션 개발자 : RESTful API 사용에 능숙한 개발자, 즉 거의 모든 웹 및 모바일 개발자는 사전에 교육된 모델을 사용해 기존 애플리케이션에 머신러닝 기능을 추가할 수 있다. 예를 들어 음성에서 텍스트로 변환하기, 언어 번역, 코멘트 순위 매기기 등이 있다.

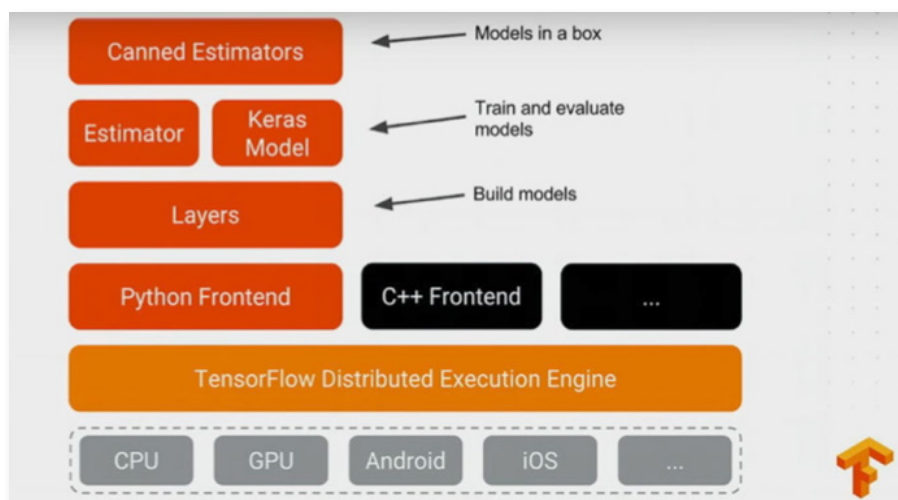


그림 2 | 텐서플로우 아키텍처 구성

추상화를 통해 더 많은 개발자로 확산하는 머신러닝

텐서플로우는 다양한 기량을 갖춘 더 많은 개발자가 애플리케이션별로 이를 사용할 수 있도록 하는 높은 수준의 아키텍처를 제공한다. <그림 2>을 보면 전문가는 계층(Layers) 이하 스택에서 관련 툴을 사용해서 독창적인 모델을 구축할 수 있다. 분야 전문가는 스택의 에스티메이터(Estimator)와 케라스(Keras) 모델 수준에, 애플리케이션 개발자는 맨 위, 즉 ‘박스 내의 모델(Models in a Box)’로 표시된 부분에서 이 기술을 이용할 수 있다.

확장 가능한 하드웨어 플랫폼

텐서플로우는 테스트부터 생산에 이르기까지 다양한 규모로 확장 가능한 소프트웨어 플랫폼이다. 텐서플로우 컴파일러 프레임워크인 XLA는 머신러닝 모델을 LLVM(Low Level Virtual Machine) 오픈소스 프레임워크를 사용하는 다양한 하드웨어 아키텍처로 컴파일한다.

XLA는 하드웨어 스케일과 관련된 중요한 두 가지 사항에 대처하도록 설계됐다. 첫째, 기능 확장을 위해 사물인터넷(IoT)과 스마트폰에서 실행하는 소형 모델을 구축할 수 있다. 실제로 쉘컴은 CPU에서 헥사곤(Hexagon) DSP로 머신러닝 애플리케이션을 옮겨 스냅드래곤 820의 성능을 8배 높이기도 했다. 쉘컴은 XLA 컴파일러를 사용하지 않았다. 그러나 스마트폰의 성능과 전력 프로파일 내에서 최적화된 머신러닝 모델을 실행해 얻을 수 있는 효과를 입증했다. XLA는 현재까지 IBM의 파워AI(PowerAI) 배포판과 모비디우스(Movidius)의 미레이드2(Myraid2) 가속기에 적용됐다.

그러나 현재의 하드웨어 속도로는 대형 머신러닝 모델의 상용 요구 사항을 충족하지 못한다. 연구 용도로는 충분한 속도지만 연구원이 모델을 교육하고 결과를 받기까지 며칠이나 몇 주, 또는 한 달 가까이 걸리기도 한다. 연구원이 자신의 가설을 테스트할 수는 있지만 이를 수천, 수백만 명이 사용하는 데이터센터 서버에 적용하는 것은 전혀 다른 문제인 것이다.

머신러닝의 깊고 넓은 프로그래밍 패러다임은 새로운 것이다. 여기에는 대규모 매트릭스와 벡터를 수용하면서도 높은 추상화를 지원하는 아키텍처가 필요하다. 그러나 현재까지는 대부분의 경우 아직 머신러닝에 최적화되지 않은 하드웨어 아키텍처에서 이 유형의 연산 역량을 강화하기 위한 목적으로 이뤄진다.

프랑수아 콜레가 시연한 케라스(Keras)는 최소한의 코딩으로 빠른 심층 신경망 테스트가 가능하도록 설계된 모듈형 오픈소스 신경망 라이브러리이다. 그는 비디오를 분석해 영상 속 여성이 무엇을 하고 있으며 그가 입고 있는 셔츠의 색이 무엇인지 알려주는 코드 20줄 미만의 케라스 애플리케이션을 선보였다.

첫 번째 질문은 간단치 않다. 여성이 짐을 싸고 있는지, 짐을 풀고 있는지 머신러닝 모델이 판단해야 하기 때문이다. 콜레의 시연은 짧은 코드의 소프트웨어와 높은 수준의 추상화를 통해 이 두 가지 질문에 답하는 방법을 보여줬다. 그러나 프로덕션 환경에서는 주어진 애플리케이션에서 발생하는 지연이 너무 커서 모델에 대한 최적화가 필요할 가능성이 매우 높다.

스택 내에서, 그리고 케라스 프로그램 범위 내에서 나타나는 성능 병목 현상은 문제가 되는 코드를 네이티브 C++ 코드로 다시 작성하는 방법으로 일부 해결할 수 있다. 그러나 구글 검색 랭킹과 같이 초대형 모델은 단일 GPU는 물론 여러 GPU에서도 감당할 수 없고, 고속 내부 시스템 버스 또는 여러 시스템을 연결하는 광섬유 버스로 통신하는 상호 연결된 GPU 뱅크에서 실행해야 한다.

연구 단계에서 입증된 머신러닝 모델이라고 해도 경제성 때문에 프로덕션 환경에 적용하지 않을 수 있다. 특히 구글 검색에서 사용자가 기대하는 것과 같은 수백 ms 수준의 낮은 지연이 필요한 경우 더 그렇다. 하드웨어를 추가할 수 있지만 자칫 비용은 크게 늘면서 문제는 여전히 해결하지 못할 수 있다. 이 때문에 대규모 머신러닝을 추구하는 많은 기업과 연구자가 실리콘 하드웨어 아키텍처에서 필요한 역량이 구현될 때까지 기다리고 있다. 대신 텐서플로우는 전문가가 C++로 네이티브 코드를 작성해서 시스템 버스 및 상호 연결된 시스템을 통해 워크로드를 분산하고 조율할 수 있도록 분산 API를 통해 최적화한다.

왜 텐서플로우인가, 왜 오픈소스인가

구글은 다음과 같은 4가지 이유로 텐서플로우와 이를 중심으로 한 커뮤니티를 구축했다.

1. 텐서플로우의 기능을 확장하기 위해 독립적으로 개발된 코드 기여와 새로운 사용 사례를 확보하는 목적이다. 구글의 직원 혜택이 아무리 좋아도 구글에서 일하기를 원하지 않으면서 텐서플로우에 기여하고자 하는 전문 개발자가 있을 것이다. 또한, 구글의 머신러닝 애플리케이션이 텐서플로우 2.0 이상을 사용할 모든 잠재적 사례를 다 지원하지도 못한다. 결국, 구글은 오픈소스 라이선스를 주고 대신 여러 개발자의 코드와 관점을 취하는 것이다.
2. 구글은 인재를 확보함으로써 번창하는 기업이다. 그리고 최고의 인재를 채용하는 가장 좋은 경로는 오픈소스 커뮤니티와 학생 인턴이다.
3. 학계와 업계의 인공지능 및 머신러닝 커뮤니티는 대단히 개방적이어서 서로의 연구를 공유하고 서로 발전의 발판을 삼기도 한다. 심지어 구글과 경쟁업체인 페이스북의 연구팀이 서로 논문에 기여할 정도이다. 폐쇄적 접근 방식은 혁신의 속도를 느리게 한다.
4. 구글은 커뮤니티의 혁신에 힘입어 제품을 개선하고 구글 클라우드 플랫폼에서 텐서플로우를 제공해 이익을 거둘 수 있다.

텐서플로우는 플랫폼이다. 구글의 궁극적인 목적은 텐서플로우를 리눅스만큼 보편적이고, 안드로이드만큼 큰 플랫폼으로 키우는 것이다. 