1. **플랫폼**
2. 구글 텐서플로우 (Google TensorFlow)

구글의 지주회사인 알파벳의 에릭 슈미트 회장이 지난 11월 10일 구글이 아시아 지역의 기자들을 도쿄에 모아놓고 차세대 기술로 주목 받는 `머신러닝`이 더욱 인간 친화적으로 발전할 것으로 전망했다. 구글 에드센스 (Google Adsense)나 구글 애널리틱스 (Google Analytics)와 같이 이미 2011년부터 구글은 본격적으로 이용자의 정보를 수집해 데이터를 분석하여 이용자 맞춤형 서비스를 시작했다 [그림 1].

광범위한 정보 수집과 분석을 통해 새로운 통찰력을 얻게 되는 것에 세상은 열광하였고, 구글은 IT 서비스 업계의 가장 막대한 영향을 미치는 기업이 되었다. `빅데이터`나 `머신러닝`은 이러한 구글의 중심에 서 있는 기술이다.

|  |  |
| --- | --- |
|  |  |

[그림 1] 구글의 이용자 맞춤형 서비스, 구글 에드센스 및 구글 애널리틱스

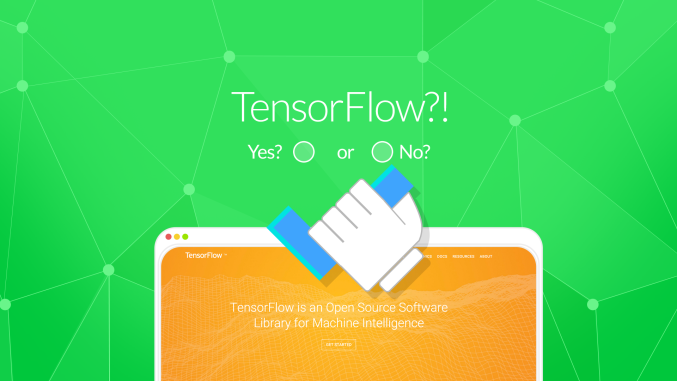
구글은 이미 검색엔진뿐만 아니라, 세상의 많은 요소들에서 활발하게 정보를 수집하고 있다. 데이터 분석이 갖는 의미, 그 자체를 구글은 아주 잘 이용하고 있다. 수집한 데이터는 지금 당장 사용하지 않아도 언젠가는 가치를 갖게 되고, 이미 수집한 데이터를 이용하여 그 답을 찾은 서비스에 대해서는 더 많은 데이터를 모으고자 하고 있다.

2015년 11월, 구글은 자신들의 핵심 기술인 머신러닝 연구를 위해 개발한 오픈소스 라이브러리 텐서플로우 (tensorflow)를 공개하였다 [1]. 텐서플로우는 소스 코드와 API 명세가 공개되어있으며, 아파치 2.0 라이센스로 이용할 수 있다. 아파치 2.0 라이선스를 통해 누구든 텐서플로우 소프트웨어를 이용한 새로운 소프트웨어를 제작할 수 있으며, 저작권을 양도하거나 전송할 수 있다. 또한, 텐서플로우를 활용하여 개발한 소프트웨어를 상업적인 목적으로도 이용할 수 있다.

텐서플로우는 탑제된 엔진 자체도 상당히 유연한 것으로 평가된다. 데이터를 분석할 수 있는 컴퓨팅자원인 CPU뿐만 아니라, GPU를 이용할 수도 있다. 또한, 듀얼코어나 쿼드코어 같은 데크스톱 PC용 멀티 코어 프로세서를 활용할 수 있을 뿐만 아니라, 가상화 기술을 통해 구축한 서버나 수천 대 규모의 데이터센서 환경에서도 이용할 수 있다.

일반적인 구조의 PC뿐만 아니라, 모바일 기기에서도 텐서플로우를 실행할 수 있다. 구글은 텐서플로우가 이렇게 다양한 기기에서 작동하지만 관련 API는 기계에 관계없이 모두 같은 코드를 이용한다고 밝혔다. 텐서플로우는 C++뿐만 아니라, 머신러닝 분야에 많이 사용되고 있는 파이썬으로도 활용할 수 있다.

그러나 텐서플로우에는 몇 가지 제약이 존재한다. 먼저, 클라우드 서비스를 제공하지 않는다는 것이다. 구글은 이미 텐서플로우를 다양한 서비스에 결합하여 머신러닝 기법을 적용하고 있지만, 몇몇 시스템에서는 텐서플로우를 활용할 수 없다. 즉, 텐서플로우가 지원되지 않는 시스템에서 텐서플로우를 이용하기 위해서는 시스템을 직접 구축해야 한다는 것이다.



[그림 2] 구글 텐서플로우 (Google TensorFlow)

또한, 구글은 텐서플로우에 대한 가이드라인이나 텐서플로우를 통해 활용할 수 있는 어떠한 데이터도 제공하지 않고 있다. 구글이 제공하는 것은 데이터를 분석할 수 있는 틀 (frame)뿐이다. 모델을 만들고 이를 적용하는 것은 전적으로 프로그래머 개인이 모두 구현해야 한다.

텐서플로우는 데이터를 플로우 그래프 (data flow graph)를 이용하여 인공신경망을 구현하는 컴퓨팅 라이브러리이다. 데이터 플로우 그래프의 구성요소로는 노드 (node)와 엣지 (edge)가 있으며, 노드는 수학적 계산을 통해 값을 도출하고, 엣지는 노드 사이의 입출력 관계를 표현한다. 노드에서 계산된 값은 엣지를 통해 다차원 배열로 전달되며, 노드는 다차원 배열에서 값을 읽여 병렬적으로 계산한다.

1. IBM 왓슨 (IBM Watson)

IBM에서 개발한 왓슨 (Watson)은 자연어 형식으로 구성된 문장에 대한 답을 도출하는 컴퓨터 시스템이며, IBM의 시험 책임자 데이비드 페루치 (David Ferrucci)가 주도한 IBM의 DeeoQA 프로젝트를 통해 개발되었다. 왓슨은 IBM의 초대 회장인 토머스 J. 왓슨 (Thomas John Watson)에서 이름을 차용했다.



[그림 3] 퀴즈 쇼 제퍼디! (Jeopardy)에 참가하여 우승을 차지한 왓슨 (Watson)

2011년, 왓슨은 미국 ABC의 인기 퀴즈 쇼 `제퍼디! (Jeopardy!)`에 참가하여 인간을 압도적인 차이로 따돌리며 우승을 차지하였다 [그림 3]. 2월 14일부터 16일까지 세 개의 제퍼디 에피소으 반송에서 왓슨은 제퍼디의 금액 기준 사상 최대 우승자 브레드 러터, 가장 긴 챔피언십의 기록 보유자 켄 제닝스와 대결하였다. 첫 상금에서 켄 제닝스와 브레드 러터가 각각 300,000 달러와 200,000 달러를 받는 사이 왓슨이 100만 달러를 거머쥐는 놀라운 성능을 보였다.

왓슨은 4TB의 디스크 공간에 2억개의 페이지로 구조화된 콘텐츠에 접근하여 퀴즈쇼에서 대결하였다. 왓슨은 경기가 치러지는 동안에는 인터넷에 연결되지 않았기 때문에 순수하게 학습된 데이터를 기반으로 퀴즈쇼에서 인간을 압도적으로 따돌리며 우승을 차지한 것이다. 그러나 왓슨은 게임 신호 장치에서 자신과 경쟁하는 사람들을 지속적으로 앞질렀으나, 몇 개의 낱말로 된 단서만 가지고 있는 일부 분류에 대해서는 응답에 오류가 발생하는 등의 문제가 존재했다.

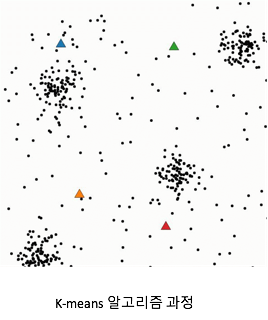
IBM에서 발표한 왓슨의 다음 목표는 인간 수준의 이해력과 분석력을 갖추는 것이다. 2014년을 기준으로, 왓슨의 현재 속도는 2011년에 비해 24배나 빨라졌으며, 크기는 10분의 1로 감소하였다. 이미 미국의 암 연구 센터 등에서는 왓슨을 이용한 논문 분석 등, 다양한 분야에서 이용하고 있다. 또한, 최근에는 보통 과학자가 하루에 5개씩 읽으면 38년이 걸릴 7만개의 논문을 왓슨이 한 달만에 분석하여 항암 유전자에 영향을 미치는 단백질 6개를 찾아내는 놀라운 성능을 보였다. 왓슨은 과학 분야에서뿐만 아니라, 법류 분야에서도 활용 영역을 넓겨하고 있는데, 수 많은 판례를 분석하여 답을 내놓은 방식으로 활용되고 있다. 특정 기업의 인수 합병 금지 조항 같은 것을 물어보면 금방 답을 내놓는 수준으로 발전하였다.



[그림 4] IBM Bluemix 서비스 플랫픔

현재 왓슨은 PaaS (Platform as a Service)의 형태로 제공되고 있는 IBM의 Bluemix를 통해 이용할 수 있다. 서비스를 제공하는 형태는 웹에서 API로 호출하여 이용하는 방법이나 오픈 소스로 GitHub를 통해 다운받아 이용할 수 있다. IBM은 Java나 JavaScript 기반 라이브러리인 Node.js에서도 사용할 수 있는 다양한 플랫폼을 지원하고 있다. 왓슨은 기존의 컴퓨터가 인식하지 못하는 문서, 사진, 음성 녹음 등의 정보를 스스로 이해하고 의미 있는 지식을 만들어낼 수 있다는 점이 큰 장점이다.

1. **알고리즘**
2. K-means Clustering

"K-means”에 대한 개념은 1957년 후고 스테인하우스에 의해 소개되었으나, 용어 자체는 1967년에 James MacQueen에 의해 처음 사용되었다. 현재 사용되고 있는 표준 알고리즘은 1957년에 Stuart Lloyd가 펄스 부호 변조(PCM)를 목적으로 처음으로 고안 하였으나 1982년이 되어서야 컴퓨터 과학 매거진에 처음 공개되었다. 알고리즘이 공개되기 이전인 1965년에 E. W. Forgy 또한 같은 알고리즘을 제안하였다. 차후 1975년과 1979년에 Hartigan과 Wong에 의해 거리 계산이 필요하지 않은 좀 더 효율적인 방법이 소개 되었다.

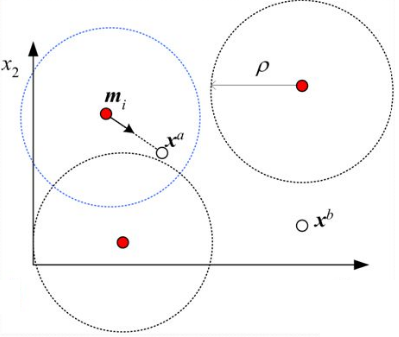
k-means 클러스터링 알고리즘은 클러스터링 방법 중 분할법에 속한다. 분할법은 주어진 데이터를 여러 파티션으로 나누는 방법이다. 예를 들어 n개의 데이터 오브젝트를 입력받았다고 가정하자. 이 때 분할법은 입력 데이터를 n보다 작거나 같은 k개의 그룹으로 나누는데, 이 때 각 군집은 클러스터를 형성하게 된다. 다시 말해, 데이터를 한 개 이상의 데이터 오브젝트로 구성된 k개의 그룹으로 나누는 것이다. 이 때 그룹을 나누는 과정은 거리 기반의 그룹간 비유사도 (dissimilarity) 와 같은 비용 함수 (cost function) 을 최소화하는 방식으로 이루어지며, 이 과정에서 같은 그룹 내 데이터 오브젝트의 유사도는 증가하고, 다른 그룹에 있는 데이터 오브젝트와의 유사도는 감소하게 된다. k-means 알고리즘은 각 그룹의 중심 (centroid)과 그룹 내의 데이터 오브젝트와의 거리의 제곱합을 비용 함수로 정하고, 이 함수 값을 최소화하는 방향으로 각 데이터 오브젝트의 소속 그룹을 업데이트 해 줌으로써 클러스터링을 수행하게 된다.

n개의 d-차원 데이터 오브젝트 , 집합이 주어졌을 때, K-평균 알고리즘은 n개의 데이터 오브젝트들을 각 집합 내 오브젝트 간 응집도를 최대로 하는 k개의 집합 으로 분할한다. 다시 말해, 가 집합 의 중심점이라 할 때

각 중심점 내 오브젝트간 거리의 제곱합을 최소로 하는 집합 S를 찾는 것이 이 알고리즘의 목표다. 이 목적 함수의 전역 최솟값 (global minimum) 을 찾는 것은 NP-난해 문제이므로, 언덕 오르기 (hill climbing) 방식으로 목적 함수의 오차를 줄여가며 지역 최솟값 (local minimum) 을 발견했을 때 알고리즘을 종료함으로써 근사 최적해를 구한다.

1. Adaptive Resonance Theory Ⅱ (ART II)

ART(Adaptive Resonance Theory)가 갖는 가장 큰 특징은 인간의 학습 방법을 모방한다는 것이다. 인간의 두뇌는 새로운 정보가 들어올 때, 기존에 학습했던 정보는 보존하면서 새로운 정보만을 받아들일 수 있다. 예를 들어, 사과라는 것을 학습한 사람이 오렌지를 본다고 해서 사과와 오렌지 모두를 다시 학습하는 것이 아니라, 오렌지만을 학습하면 되는 것과 같다. 방금 전에 100페이지까지 공부한 사람이 101페이지를 공부하려면 다시 1페이지부터 101페이지를 공부해야 하는 것을 비정상적으로 여길 만큼, 인간에게 있어서 기존의 학습 데이터를 보존하면서 새로운 정보를 받아들이는 것은 너무나 당연한 것이다. 그러나 SOM(Self-organizing Map), Perceptron 등 기존의 인공신경망은 학습을 통해 가중치를 설정한 상태에서 새로운 정보가 들어오면, 이전에 학습한 패턴에 대해서도 다시 학습을 수행해야 하는 비효율적인 측면이 존재한다. ART는 이러한 문제점을 해결하기 위해 새로운 정보의 학습이 기존 정보에 영향을 주지 않도록 설계되었다. ART는 1976년 미국 보스턴 대학의 Stephen Grossberg와 Gail A. Carpenter에 의해 개발된 인공신경망 모델로써 이진 값을 처리하는 ART1과 아날로그 입력 데이터도 처리가 가능한 ART2가 있고, 최근에는 화학적 신호 전달 메커니즘과 ART를 결합한 ART3도 연구되었다. 구현한 머신러닝 API에서 제공하는 ART 알고리즘은 아날로그 입력 데이터 처리가 가능한 ART2이다. ART2의 구조는 SOM, LVQ와 매우 유사하다. ART2의 입력 벡터를 , 출력 층의 j번째 노드에 대한 가중치 벡터를 이라 할 때, 출력층의 각 노드들과 유클리드 거리를 계산하고 승자 노드, 즉 입력 벡터와 유클리드 거리가 작은 노드를 선택한다.



**<ART2의 경쟁학습 과정>**

ART2의 경쟁학습은 아래의 (수식 2.5.1, 수식 2.5.2)로 표현된다. O\_j는 유클리드 거리를 계산한 출력 값을 의미하고 O\_(j\*)는 승자 노드의 출력 값을 의미한다.

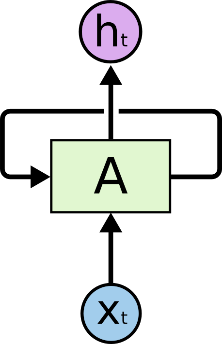
승자 노드의 출력 값과 유사도(ρ) 값의 비교를 통해 유사도 값이 승자 노드의 출력 값보다 클 경우 가중치를 조정한다 (수식 2.5.3). 이와 달리 작을 경우 새로운 클러스터를 생성하고 가중치를 조정한다. n은 요소의 수를 의미한다.

위 과정을 반복하면서 새로운 입력 벡터가 제시될 때마다 군집화한다.

1. Recurrent Neural Network

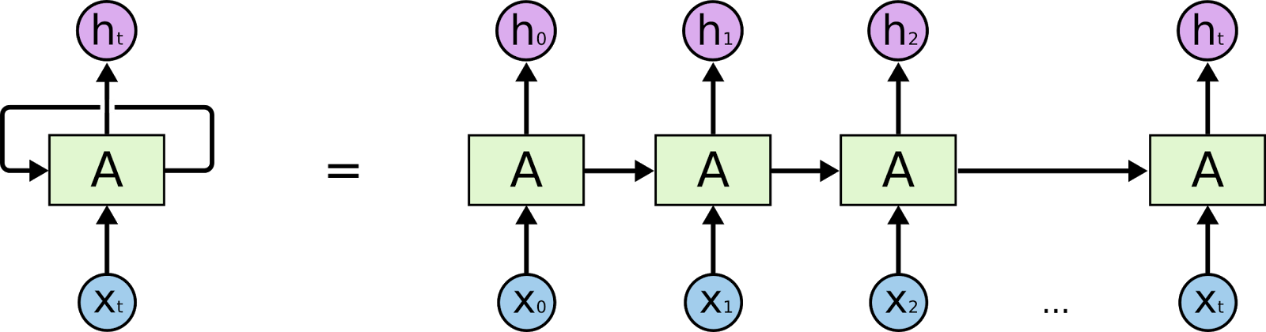
사람은 생각을 매초 처음부터 다시 시작하지 않는다. 우리가 이 글을 읽을 때도, 우리는 각 단어를 이전 단어들에 기반을 두어 이해한다. 우리는 기존의 모든 것을 버리고 처음부터 생각을 다시 시작하지 않는다. 다시 말해 우리의 생각에는 지속성이 있다. 전통적인 신경망의 경우 이러한 속성을 담아내지 않았고, 따라서 Recurrent Neural Network(이하 RNN)은 이러한 단점을 타개하기 위해 고안되었다. 이를테면 영화 속의 매 장면에서 어떤 종류의 사건이 일어나고 있는지 분류하고 싶다고 상상해 보십시오. 전통적 신경망들이 어떻게 이전 장면들을 사용하여 이후 장면들을 추론할지는 불분명합니다.

RNN은 이 주제를 다룬다. 그 구조는 전통적인 신경망에 루프를 추가한 구조이다. 루프를 통해서 전통적인 신경망에 생각의 지속성을 돕고자 했다.



루프를 가진 순환 신경망.

위 다이어그램에서, 신경망 A는 입력 를 보고 값 를 출력합니다. 루프는 정보가 네트워크의 한 단계에서 다음 단계로 전달되도록 한다. 루프를 각각의 네트워크라고 바꿔 그림을 표현한다면 아래와 같은 그림이 된다.



루프를 펼친 순환 신경망

이 사슬 같은 특성은 순환 신경망이 시퀀스 그리고 리스트와 밀접하게 연관되었음을 보여준다. 이 구조는 순차적인 데이터를 처리하기 위한 신경망의 자연스러운 구조입니다. 지난 몇 년 동안, 순환 신경망을 다양한 문제들에 적용해 놀라운 성공이 거두었다. 음성 인식, 언어 모델, 번역, 영상에 주석 달기 등 그 사례는 매우 많다.

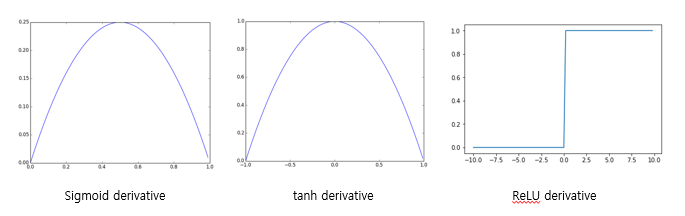
이 작동 과정을 수식을 통해 살펴보면

시간 에서 은닉층의 상태, 즉 은닉층이 갖고 있는 라고 하자. 입력 벡터 와 가중치 , 시간 에서 출력 값 , 관계 행렬 의 함수이다. Feedforward Network Networks와 마찬가지로 출력 값의 오차를 계산하고 이 오차는 다음 은닉층으로 전파되는데, 그 값을 기준으로 를 역전파(Backpropagation)를 이용해 수정해 나간다.

활성함수 는 최근 많이 사용되고 있는 Rectified Linear Units(ReLU)보다 초기에 사용되어 왔던 Sigmoid와 tanh를 여전히 많이 사용하고 있다. 이 함수들의 경우 출력 값의 범위를 제한해주면서 전 구간에서 미분이 가능하기 때문에 역전파가 잘 이루어진다. 루프 구조를 가지고 있는 RNN은 모든 순간마다 피드백이 이루어진다. 즉, 특정 시점 서 이전에 진행되었던 모든 결과 값( ) 에 대한 역전파가 필요하다. 그러나 실제로 값을 무한히 저장할 수는 없으므로 사용가능한 메모리를 적정 수준에 맞추어 저장한다.

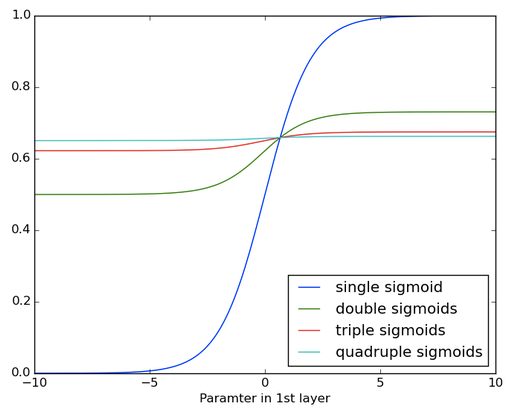
Long Short-Term Memory Units(LSTM)

1990년대 초반 Vanishing Gradient 문제가 신경망의 큰 문제로 나타났다. 역전파를 통해 가중치 수정이 점점 이루어지지 않는다는 것인데 아래 그래프를 통해 설명한다.



왼쪽에서 차례대로 Sigmoid와 tanh 함수를 미분한 그래프이다. 결과 값의 범위는 이다. 이 범위 내의 값이 수 차례 곱하기 연산을 통해 이루어진다면 점점 값은 작아지다 0에 수렴한다. 이러한 값의 특성이 가중치를 수정하는데 있어서 처음 가중치를 수정하는 데에는 문제가 없지만 층이 점점 깊어질수록 변화량이 점점 줄어들 것이다.

또한 아래 그림은 sigmoid 함수를 여러 번 곱한 그래프를 표현한 그래프이다. 파란색 선으로 표시된 하나의 sigmoid 함수에 비해 붉은색 선의 4번 곱해진 sigmoid함수의 경우 거의 모든 구간에서 0에 가까워졌다는 것을 알 수 있고 이를 미분한 값 거의 모든 구간에서 0에 가까울 것이다.



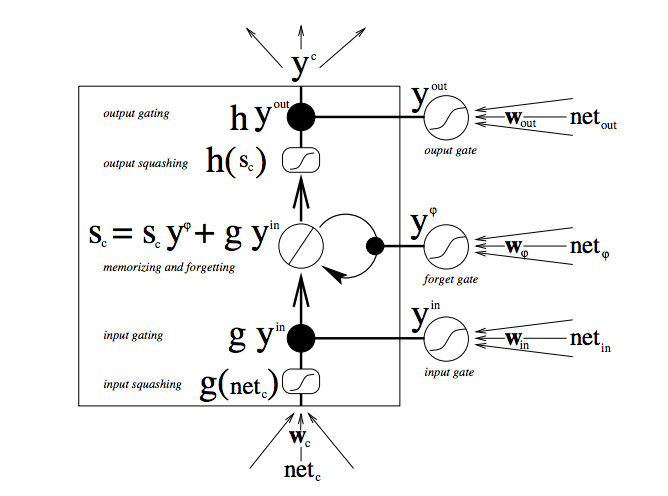
Long Short-Term Memory Units(LSTM)

RNNs의 변경인 LSTM 유닛은 90년대 중반에 처음으로 등장했다. LSTM은 기존이 RNN이 Vanishing Gradient가 발생하는 것을 보완하기 위해 출현했고, 이는 시간을 거슬러서 오차를 통한 수정이 잘 이루어지도록 도와준다. 결과적으로 1000단계가 넘게 거슬러 올라갈 수 있다. 다시 말해, 더 오래 전 일도 잘 기억한다는 의미가 되겠다.

LSTM 유닛은 여러 개의 게이트(gate)가 붙어있는 셀(Cell)로 이루어져 있으며 이 셀의 정보를 새로 셀의 정보를 새로 저장 / 불러오기 / 유지하는 기능을 가지고 있다. 셀은 셀에 연결된 게이트의 값을 보고 무엇을 저장할지, 언제 정보를 내보낼지, 언제 쓰고 언제 지울지를 결정한다. 이 게이트는 바이너리(0,1) 값을 이용해 스위치 되는 것이 아니라 아날로그 값을 이용한다. 즉, 각 게이트는 0에서 1사이의 값을 가지며 게이트의 값에 비례해서 여러 가지 작동을 합니다.

각 게이트가 갖는 값, 즉 게이트의 계수(또는 가중치, weight)는 은닉층의 값과 같은 원리로 학습된다. 즉 게이트는 언제 신호를 불러올지 / 내보낼지 / 유지할지를 학습하며 이 학습과정은 출력의 오차를 이용한 경사하강법(radient descent)을 사용합니다.

아래 그림은 LSTM 유닛과 게이트의 작동 방식을 시각화한 것 입니다.



우선 입력부터 살펴보면, 입력 신호로 3개 벡터가 들어온다. 이 신호들은 현재 입력 신호와 과거의 셀에서 온 피드백을 합친 것인데, 바로 입력되는 것이 아니라 3개의 게이트로 들어가고 각 게이트는 어떻게 다룰지 결정한다. 그림에서 검은색 점은 게이트를 나타낸다. 아래에 있는 게이트는 얼마나 입력 값을 반영할지, 중간 게이트는 현재 갖고 있는 값 중 얼마를 잊을지, 위에 있는 게이트는 얼마나 출력할지를 결정한다.