Numenta

HIERARCHICAL TEMPORAL MEMORY

including

HTM Cortical Learning Algorithms

VERSION 0.2.1, SEPTEMBER 12, 2011

Numenta 번역 라이센스

여기에 포함된 글, 알고리즘, 예시 코드, 가코드 그리고 모든 다른 내용은 Numenta Inc.에서 발행된 hiearchial temporal memory("HTM") 기술에 기반하거나 번역된 것입니다. Numenta 는 원작에 대한 저작권을 가지고 있으며, 여기에 번역된 HTM 이나 알고리즘 관련한 특허에 대한 권리를 가지고 있습니다. Numenta 는 독립된 HTM 시스템의 사용이나 개발이 오직 연구 목적일 경우에 한하여, 특허 권리를 주장하지 않음에 동의하였으며, 상업이나 제품 개발을 위한 목적은 여기서 제외됩니다. HTM 기술을 상업적으로 이용하거나 제품을 사용하는 것이 Numenta 의 특허를 침해할 여지가 있는 경우, Numenta 로부터 상용 라이센스와 관련된 허락을 필요로 합니다.

앞에서 기술한 내용을 바탕으로, Numenta 는 오직 연구 목적으로만 알고리즘과 원작의 사용을 허가하며, 어떠한 상업적, 제품 사용도 이에 해당하지 않습니다. 이 라이센스에서 "상업적, 제품 사용" 이라 함은 추후에 훈련된 HTM 네트워크를 운용할 목적으로 그것을 학습시키거나, 상업적 혹은 제품 생산을 위한 응용프로그램을 개발하거나, HTM 기술의 결과물을 다른 사람들이 상업적 혹은 제품 생산을 위해 이용하도록 하는 것을 말합니다. 이 원본을 배포, 출판 혹은 복사하기 위해서는 반드시 이 번역된 라이센스의 전문을 영어와 해당 언어로 포함해야만 합니다.

어떠한 특허권에 대한 명시적 혹은 묵시적 라이센스도 본 라이센스에 부여되지 않았습니다. Numenta 는 아래의 허가된 번역본에 대하여, 번역의 품질이나 정확도에 대하여 어떠한 책임이나 의무도 가지지 않습니다.

Numenta Translation License

The text, algorithms, sample code, pseudo code and other work included herein are based upon or translated from certain works related to hierarchical temporal memory ("HTM") technology published by Numenta Inc. Numenta holds the copyright in the original works and patent rights related to HTM and the algorithms translated herein. Numenta has agreed not to assert its patent rights against development or use of independent HTM systems, as long as such development or use is for research purposes only, and not for any commercial or production use. Any commercial or production use of HTM technology that infringes on Numenta's patents will require a commercial license from Numenta.

Based on the foregoing, Numenta grants you a license to use these algorithms and works for research purposes only and not for any commercial or production use. For purposes of this license, "commercial or production use" includes training an HTM network with the intent of later deploying the trained network or application for commercial or production purposes, and using or permitting others to use the output from HTM technology for commercial or production purposes. Any distribution, publication, or copying of this work must include the full text of this Translation License in both English and the target language.

NO EXPRESS OR IMPLIED LICENSES TO ANY PATENT RIGHTS ARE GRANTED BY THIS LICENSE. NUMENTA SPECIFICALLY DISCLAIMS ANY LIABILITY OR RESPONSIBILITY FOR THE QUALITY OR ACCURACY OF ANY TRANSLATIONS LICENSED HEREUNDER.

이것 먼저 확인하세요!

이 문서는 아직 미완성 단계입니다. 이 시스템을 이해하기 위해 반드시 알아야 하는 몇가지 사항들이 아직 포함되어 있지 않습니다.

이 문서에 있는 내용은

이 문서는 Numenta 가 개발한, 배우고 예측하는 새로운 형태의 알고리즘에 대한 세부적인 내용을 담고 있습니다. 프로그래머들이 이 알고리즘을 이해하고 직접 실행해볼 수 있을 정도로 자세한 내용이 기술되어 있습니다. 문서는 알고리즘의 배경 내용을 소개하는 장 (chapter)들로 시작합니다. 만약 예전에 Numenta 에서 발표한 논문과 글들을 읽어본 적이 있다면, 익숙한 내용이 있을 수도 있습니다. 그 외의 다른 내용들은 모두 새롭게 작성되었습니다.

이 문서에 없는 내용은

새로운 알고리즘을 실제로 구현하기 위해서 필요한 사항들 중에, 이 초기 문서에 빠져있는 내용은 다음과 같습니다.

- 알고리즘의 대부분을 소프트웨어로 구현하고 테스트하였지만, 실제 그 결과는 이 문서에 포함되어있지 않습니다.

-어떻게 이 알고리즘을 실제적인 문제들에 적용할 수 있을지에 대한 자세한 내용은 담겨있지 않습니다. 센서나 데이터베이스로부터 얻어진 데이터를 어떻게 알고리즘에 맞는 형태로 변환할 것인가에 대한 내용은 빠져있습니다.

-이 알고리즘은 실시간으로도 학습할 수 있습니다. 몇몇 드문 상황에서 이러한 실시간 학습을 제대로 구현하기 위해 필요한 세부사항들이 빠져있습니다.

-앞으로 추가될 내용들은 희소분포표상(sparse distributed representations)에 대한 논의와 실제적인 프로그램과 사례에 대한 기술, 그리고 부록들의 참고문헌 등이 있습니다.

이 알고리즘이 여러 사람들의 흥미를 이끌어낼 것이라 생각했기 때문에 이 문서를 작성하였습니다. 하지만 여기에 빠져있는 내용들이 있다고 해서, 연구자들이 알고리즘을 이해하고 실험해보는 일이 크게
방해받지는 않을 것입니다. 새로운 진행 사항이 생기는대로 정기적으로 이 문서를 개선할 예정입니다.

목 차

서 문

우리는 쉽게 할 수 있지만, 컴퓨터는 할 수 없는 일들이 많이 있습니다. 눈에 보이는 시각 패턴을 인식하는 일이 그렇고, 언어를 이해하는 일, 촉각을 통해 사물을 알아차리고 다루는 일, 복잡한 상황에서 길을 찾는 일들은 우리가 어렵지 않게 할 수 있는 일들입니다. 하지만 수 십년 동안의 연구에도 불구하고, 인간처럼 생각하고 활동할 수 있는 알고리즘을 실제로 구현한 컴퓨터는 아직 없습니다.

인간의 이러한 능력은, 주로 신피질 (neocortex)이라고 불리는 구조에서 비롯됩니다. 여기서 소개하는 Hiearchial Temporal Memory(HTM)은 바로 이 신피질의 기능을 모방한 기술입니다. HTM 은 여러 인지 상황에서 인간 수준의 능력을 가지거나 그것을 뛰어넘는 기계를 만드는 일을 위한 기술입니다.

이 문서는 HTM 기술에 대한 상세한 내용을 다루고 있습니다. 1 장은 HTM 의 전반적인 개요를 보여주며, 계층구조와 공간분포표상 (spatial distribution representation) 그리고 시간에 따라 변화하는 입력을 통해 학습하는 것이 왜 중요한지에 대해 다루고 있습니다. 2 장은 HTM-피질학습 알고리즘을 조금 더 상세히 보여줍니다. 3 장과 4 장은 HTM 학습 알고리즘을 위한 가 (pseudo) 코드를 제공하며, 공간풀러 (Spatial pooler)와 시간풀러 (Temporal Pooler)의 두 부분으로 나뉘어 있습니다. 숙련된 소프트웨어 프로그래머라면, 2 장부터 4 장을 다 읽고 난 후에, 이 알고리즘을 재구성해서 실험을 해볼 수 있습니다. 일부 독자들이 우리의 이러한 성과를 더 발전시키리라 기대합니다.

예상 독자층

이 문서는 전문적인 교육을 받은 독자들을 대상으로 하고 있습니다. 뇌과학의 기본적인 지식이 반드시 필요하지는 않지만, 수학과 컴퓨터 공학의 기본적인 개념들은 이미 알고 있다고 가정하였습니다. 이 문서는 학과 수업에서 읽기 과제로도 사용될 수 있도록 작성되었습니다. 우리가 우선적으로 생각한 독자층은 컴퓨터공학이나 인지과학을 전공하는 학생들이나, 인간과 같은 방식으로 생각하는 인공지능시스템을 만드는데 관심이 있는 소프트웨어 개발자 입니다.

아직 전공지식이 없는 독자들에게는, *1 장: HTM 개요*가 특히 유용하리라 생각합니다.

이전 문서들과의 관계

HTM 이론의 일부분은 2004 년 발간된 책 '온 인텔리전스(On intelligence)'와 Numenta 가 발행한 백서, 그리고 Numenta 의 구성원들이 작성한 출판된 논문들에도 기술되어 있습니다. 이러한 기존 문서들을 다 읽어보지 않았더라도, 대부분의 내용과 개선 사항이 이 문서에 담겨있습니다. 2~4 장에 기술된 HTM 알고리즘은 아직 어느 곳에도 공개된 적이 없습니다. 이 새로운 알고리즘은 제타 1(Zeta 1)으로 불리는 1 세대 알고리즘을 대체합니다. 과거에는 이 새로운 알고리즘을 "고정-밀도 분포 표상 (Fixed-density Distributed Representations" 또는 "FDR"이라고 불렀지만, 지금은 이 용어를 사용하지 않습니다. 이 알고리즘의 새로운 이름은 HTM 피질 학습 알고리즘 또는 HTM 학습 알고리즘입니다.

Numenta 의 공동설립자인 제프 호킨스(Jeff Hawkins)와 산드라 블레이크스리(Sandra Blakeslee)가 쓴 '온 인텔리전스'를 꼭 읽어보기를 권합니다. 이 책에 HTM 이라는 이름이 등장하지는 않지만, HTM 이론과 그 배경이 되는 뇌과학 내용들이 읽기 쉽게 쓰여져 있습니다. 온 인텔리전스가 출판될 당시부터, 이미 HTM 의 기본적인 개념들을 이해하고 있었지만, 어떻게 알고리즘으로 구현해야할 것인지는 명확하지 않았습니다. 온 인텔리전스에서 시작된 일들이 그 이후에 어떻게 진행되었는지가 이 문서에 담겨있습니다.

Numenta 에 대해

Numenta 는 (www.numenta.com) 2005 년에 상업, 학문적 용도로 HTM 기술을 개발하기 위해 설립된 회사입니다. 그 과정과 발견 내용들을 모두 문서화하였으며, 개발된 소프트웨어를 연구와 상업적 개발 모두에 이용할 수 있도록 소프트웨어를 공개하였습니다. 우리는 이 소프트웨어가 독립적인 응용프로그램 개발 모임들을 활성화시킬 수 있도록 구성하였습니다. Numenta 의 소프트웨어와 지적인 발견들을 연구 목적으로 이용하는 것은 모두 무료입니다. 우리는 상업적인 영역의 제품 판매와 소프트웨어 및 지적 발견에 대한 특허를 통해 이윤을 만들어낼 계획을 가지고 있습니다. 우리는 Numenta 뿐만 아니라 우리의 사업 파트너들도 성공할 수 있도록 최선을 다할 것입니다.

Numenta 는 캘리포니아 레드우드 시(Redwood City)에 자리하고 있으며, 사적인 기금을 통해 운용됩니다.

저자들에 대하여

이 문서는 Numenta 의 여러 구성원들의 노력을 통해 만들어 졌습니다. 각 세션의 주요 저자들과 개선 사항들은 다음과 같습니다.

문서 개선 기록

아래 테이블은 여러 버전들의 주요 변화만을 다루고 있습니다. 형식 변경이나 사소한 내용 추가같은 작은 변화들은 나타나있지 않습니다.

버전	날짜	변경사항	주요 저자들
0.1	2010 년 11 월 9 일	1. 서문, 1,2,3,4 장과 부록: 첫번째 발행	Jeff Hawkins, Subutai Ahmad, Donna Dubinsky
0.1.1	2010 년 11 월 23 일	1. 제 1 장: 영역(Regions) 부분에 사용된 용어들을 명확히 함 2. 부록 A: 첫번째 발행	Hawkins & Dubinsky Hawkins
0.2	2010 년 12 월 10 일	1. 제 2 장: 몇 가지 내용 추가 2. 제 4 장: 참고문헌 추가; 코드 37 과 39 행 변경 3. 부록 B: 첫번째 발행	Hawkins Ahmad Hawkins
0.2.1	2011 년 9 월 12 일	1. 이것 먼저 읽으세요: 2010 년에 제거된 사항 참조 2. 서문: 소프트웨어 발행 부분 삭제	

제 1 장: HTM 개요

Hierarchial Temporal Memory (HTM)는 대뇌 신피질(neocortex)의 구조와 알고리즘 특성을 그대로 이용하기 위한 기술입니다.

신피질은 포유류의 뇌에서 지능이 자리하고 있는 곳으로 생각되는 곳입니다. 고차원의 시각, 청각, 촉각, 운동, 언어 그리고 계획과 같은 기능이 모두 신피질에 의해 일어납니다. 다양하고 유연한 인지 기능의 특성을 고려해볼 때, 신피질에도 역시 수많은 특화된 신경 알고리즘들이 있을 것이라 생각할 수도 있습니다. 하지만 실제는 그렇지 않습니다. 신피질은 놀라울 정도로 동일한 패턴과 신경 구조를 가지고 있습니다. 여러 생물학적인 증거들은, 우리가 가진 다양한 지능활동이 신피질의 어떤 공통된 알고리즘으로부터 발현됨을 보여주고 있습니다.

HTM 은 신피질과 그것의 많은 기능들을 이해하는 이론적 토대를 제공합니다. 지금까지 우리는 이 이론적 틀의 아주 일부분만을 구현해냈습니다. 시간이 지나면, 더 많은 이론들이 적용될 수 있을 것입니다. 지금까지 우리가 만든 알고리즘은 신피질 기능의 일부를 성공적으로 구현해냈으며, 상업적이고 과학적인 가치를 가지고 있습니다.

HTM 을 프로그래밍 하는 것은 기존의 컴퓨터 프로그래밍과는 다릅니다. 오늘날의 컴퓨터는, 특정한 문제를 해결하기 위해 특정한 프로그램을 만들어야만 합니다. 반면, HTM 은 일련의 데이터를 직접 받아들이면서 훈련됩니다. HTM 의 실제 능력은 자신이 어떠한 입력을 학습했는지에 따라 결정됩니다.

HTM 은 일종의 신경망으로 생각할 수도 있습니다. 광범위하게 정의하면, 신피질의 구조적인 세부사항들을 모델링하는 시스템은 모두 신경망으로 부를 수 있습니다. 하지만, 실제로는 "신경망"이라는 용어가 적절하지 않은데, 그 용어가 너무 다양한 시스템들에서 사용되었기 때문입니다. HTM 은 뉴런(HTM 에서는 셀(cell)라고 불리는)을 바탕으로 하는데, 실제 신피질처럼 뉴런들이 모여 칼럼구조(column)와 층(layer), 영역(region) 그리고 계층구조를 이루고 있습니다. 이러한 세부적인 면을 고려하면, HTM 은 전혀 새로운 형태의 신경망입니다.

이름에서 알 수 있듯이, HTM 은 기본적으로 기억장치를 바탕으로 합니다. HTM 네트워크는 시간에 따라 변화하는 수많은 데이터를 학습하며, 수많은 패턴들과 연속적인 시퀀스(sequence)를 저장합니다. 데이터를 저장하고 접근하는 방식은 기존의 프로그램들이 사용하는 것과 논리적으로 차이가 있습니다. 기존의 컴퓨터 기억장치는 수평적인 구조를 가지며, 그 안에 시간이라는 개념을 가지고 있지 않습니다. 프로그래머는 어떠한 종류의 데이터 구조도, 수평적인 컴퓨터 메모리의 가장 상위에 위치시킬 수 있었습니다. 하지만 HTM 기억장치는 훨씬 제한적입니다. HTM 의 기억장치는 계층적인 구조를 가지고 있으며,

내재적으로 시간에 기초하고 있습니다. 정보는 언제나 널리 퍼져있는 형태로 저장됩니다. HTM 의 사용자는 계층구조의 크기와 어떠한 정보를 배울 것인지를 정할 수는 있지만, HTM 이 그 정보를 어떻게 다루고 어떻게 저장할지는 지정할 수 없습니다.

비록 HTM 네트워크가 기존 컴퓨터와 차이를 가지고 있지만, 계층구조와 시간 그리고 희소분포표상(sparse distributed representations, 뒷부분에서 자세히 다룸)의 핵심 사항을 이용한다면, 일반적인 컴퓨터를 이용할 수 있습니다. 시간이 지나면, 특정 목적을 가지는 HTM 네트워크에 맞는 특화된 하드웨어가 개발될 수 도 있을 것입니다.

이 문서에서, 우리는 HTM 의 특성과 원리를 인간의 시각과 촉각, 청각, 언어 그리고 행동들을 예로 하여 제시하고 있습니다. 이러한 사례들이 HTM 을 조금 더 직관적이고 쉽게 이해하는데 도움이 될 것입니다. 하지만, HTM 의 능력은 더 일반적인 경우에도 적용될 수 있다는 점이 중요합니다. HTM 은 인간의 감각과는 전혀 다른, 레이더나 적외선 또는 금융시장이나 날씨 데이터, 인터넷 트래픽이나 패턴 혹은 글에도 쉽게 적용될 수 있습니다. HTM 은 여러 종류의 문제에 적용할 수 있는 학습-예측 기계입니다.

HTM 핵심 원리

이제, HTM 의 핵심 원리들을 다룰 예정입니다: 왜 계층구조가 중요한지, HTM 의 영역들이 어떻게 구성되는지, 왜 데이터는 희소분포표상으로 저장되는지, 그리고 왜 시간에 기반한 정보가 중요한지에 대해 설명할 것입니다.

계층 구조

HTM 은 여러 개의 영역들이 하나의 계층구조에 배열된 네트워크입니다. 영역(region)은 HTM 의 기억과 예측에 있어 핵심적인 단위이며, 다음 섹션에서 자세히 다룰 예정입니다. 일반적으로 하나의 HTM 영역은 계층구조에서 하나의 단계를 나타냅니다. 계층구조를 따라 올라가면, 언제나 수렴하는 구조를 볼 수 있으며, 아랫단계의 여러 요소들이 더 높은 단계에서는 하나의 요소로 수렴하게 됩니다. 이와는 반대로, 계층구조를 따라 아래로 내려가면, 되먹임(feedback) 구조를 통해 정보가 발산하는 것을 볼 수 있습니다. (“영역”과 “단계”는 거의 동일한 용어입니다. 앞으로 영역의 내적인 기능을 설명할 때는 “영역”을, 계층구조 안에서 영역의 역할을 언급할 때는 “단계”라는 용어를 사용하겠습니다.)

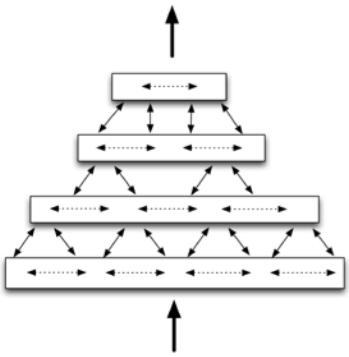


Figure 1.1: 4 개의 HTM 영역을 4 단계의 계층구조에 배열한 경우의 모식도.
정보는 하나의 단계에서 뿐 아니라, 단계와 단계 사이 그리고 계층구조 밖으로도 전달됩니다.

여러 개의 HTM 네트워크를 합치는 것도 가능합니다. 이러한 구조는 여러 종류의 데이터나 센서를 이용하는 경우에 해당합니다. 예를 들어, 하나의 네트워크는 청각정보를 처리하는 반면, 다른 네트워크는 시각정보를 처리할 수 있습니다. 각각의 독립된 네트워크에서 수렴현상이 일어나며, 각각의 갈래들은 오직 윗 단계로 수렴합니다.

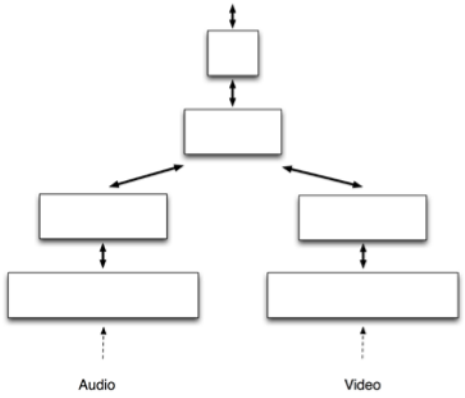


Figure 1.2: 여러 센서들로부터 수렴하는 네트워크

계층구조의 장점은 효율성입니다. 계층구조는 학습 시간과 메모리 사용량을 크게 줄여주는데, 각 단계에서 학습된 패턴들이 더 높은 단계로 가면서 새로운 형태로 조합되고 재사용되기 때문입니다. 설명을 위해 시각을 예로 들어봅시다. 계층구조의 가장 낮은 단계에서, 우리의 뇌는 시야의 아주 작은 영역에 속한 모서리(edge)나 모퉁이(corner)같은 정보를 처리합니다. 이 낮은 단계의 패턴은 중간 단계로 올라가면서 곡선이나 질감같은 조금 더 복잡한 요소로 재조합됩니다. 여기서 만들어진 하나의 호(arc)는 귀의 모서리나, 운전대의 윗부분 또는 커피잔의 가장자리일 수도 있습니다. 이 중간 단계의 패턴들은 머리나 자동차 집 같은 더 높은 단계의 특징들을 나타내는 패턴들로 합쳐집니다. 높은 단계의 사물을 새롭게 배우기 위해, 그 구성요소들을 다시 배울 필요는 없습니다.

또 다른 예로, 우리가 새로운 단어를 배우는 경우를 생각해봅시다. 새로운 단어를 배우기 위해, 글자나 음절, 음소들을 다시 새롭게 배울 필요는 없습니다.

이처럼 계층구조 간에 표상을 공유하는 것은 예측된 행동을 일반화시키기도 합니다. 새로운 동물을 보았을 때, 입이나 이빨을 발견했다면, 우리는 그 동물이 입을 통해 무언가를 먹거나 우리를 공격할 수도 있음을 예상하게 됩니다. 계층구조를 통해, 우리는 이미 알고 있는 구성요소들의 특징을 새롭게 마주한 사물들에 자연스럽게 적용할 수 있습니다.

HTM 계층구조에 속한 단계 하나가 얼마나 많은 것을 배울 수 있을까요? 또는, 계층구조에는 얼마나 많은 단계가 필요할까요? 각각의 단계에 어느 정도의 메모리를 할당할 것인지와 얼마나 많은 단계를 만들 것인지는 서로 상충되는 사항입니다. 다행히, HTM 은 자신이 받아들이는 입력의 특성과 자신이 가진 자원을 통해, 가장 최적화된 조합을 배울 수 있습니다. 우리가 하나의 단계에 더 많은 메모리를 할당한다면, 그 단계는 더 크고 복잡한 표상들을 형성하게 되고, 따라서 더 적은 단계를 필요로 하게 됩니다. 반면, 적은 메모리를 할당한다면 그 단계는 작고 간단한 표상만을 나타낼 수 있으며, 따라서 더 많은 계층구조를 필요로 합니다.

지금까지 우리는 시각 추론("추론"은 패턴 인식과 유사합니다)과 같은 조금 어려운 문제들을 다루었습니다. 하지만 많은 중요한 문제들은 시각보다 훨씬 간단하며, 하나의 HTM 영역만으로도 충분합니다. 예를 들어, 우리는 사람들이 웹사이트를 이용할 때 다음에 어느 곳을 클릭할지를 예측할 수 있는지 HTM 에 적용해보았습니다. 이 문제를 살피기 위해서는, 웹 클릭 데이터를 HTM 에 입력하면 됩니다. 이 문제에는 공간적인 계층구조가 매우 적거나 없기 때문에, 시간적인 통계 특징을 찾는 것만으로도 충분합니다. 예를 들어, 일반적인 사용자의 패턴을 통해, 특정 사용자가 다음에 어디를 클릭할지 예측할 수 있습니다. 시간을 따라 학습하는 HTM 의 알고리즘 특성은 이러한 문제들에 가장 이상적입니다.

요약하면, 계층구조는 학습 시간과 메모리 사용량을 줄이고, 일반화를 시킬 수 있습니다. 하지만, 많은 간단한 예측 문제들이 단 하나의 HTM 영역만으로도 해결될 수 있습니다.

영 역 (Regions)

계층구조에서 자주 사용되는 영역이라는 용어는 생물학에서 비롯되었습니다. 신피질은 두께가 약 2mm 정도 되는 넓고 얇은 신경조직입니다. 생물학자들은 각 영역들이 어떻게 연결되었는지에 기초하여, 신피질을 여러 부분 또는 "영역"으로 나누었습니다. 어떠한 영역들은 감각기관으로부터 직접 입력을 받은 반면, 다른 영역은 다른 수많은 영역들을 거친 후에야 감각 정보를 받아들였습니다. 계층구조를 결정하는 것은 바로 영역과 영역 사이의 연결관계입니다.

자세히 살펴보면, 모든 신피질의 영역들은 비슷한 모습을 하고 있습니다. 그 크기나 계층구조에서의 위치 등은 다르지만, 다르게 보면 그들은 서로 닮아있습니다. 2mm 두께의 신피질 영역들을 단면으로 자르면 6 개의 층을 볼 수 있는데, 5 개의 층은 세포들로 구성되어 있고 1 개는 세포가 없는 층입니다. (몇 가지 예외가 있긴 하지만, 이는 일반적인 규칙입니다.) 신피질 영역의 각각의 층은 서로 연결되어있는 수많은 세포로 이루어져 있는데, 이들은 칼럼 형태로 배열되어 있습니다. HTM 의 영역 역시, 고도로 연결된 셀들이 칼럼구조로 배열되어 있습니다. 신피질의 "3 층"은 뉴런들이 주로 피드포워드 (feed-forward) 입력을 받는 층 중 하나입니다. HTM 영역의 셀은 이 신피질의 3 층에 있는 뉴런과 거의 동등합니다.

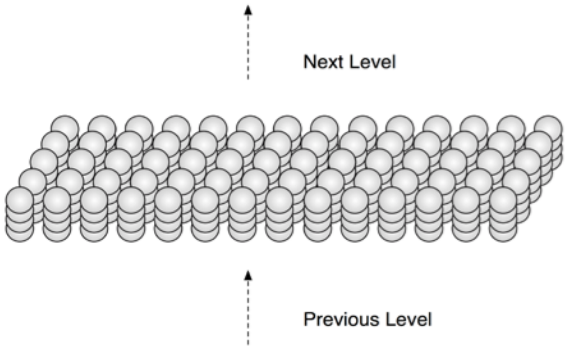


Figure 1.3. HTM 영역의 한 부분. HTM 영역은 수많은 셀들로 구성되어 있으며, 셀들은 2 차원 배열의 칼럼구조를 형성합니다. 이 그림은 각 칼럼구조당 4 개의 셀을 가지는 HTM 영역의 일부분을 보여주고 있습니다. 각각의 칼럼구조는 입력의 일부분과 연결되어 있으며, 각각의 셀은 영역 내의 다른 셀과 연결되어 있습니다 (이 연결구조는 그림에 나타나있지 않음). 칼럼구조를 포함한 이 HTM 영역은, 신피질 영역에서 하나의 층과 동일합니다.

비록 HTM 영역은 신피질 영역의 오직 일부분에 해당하지만, 복잡한 데이터를 바탕으로 추론과 예측을 할 수 있으며, 여러 실제적 문제들에서 유용하게 이용될 수 있습니다.

희소분포표상(Sparse Distributed Representations)

신피질 내의 뉴런들은 서로 복잡하게 연결되어있지만, 억제성 뉴런 (inhibitory neuron)때문에 이들 중 오직 일부만이 한 번에 활성화됩니다. 따라서, 뇌 안의 정보는 항상 수많은 뉴런들 중에 오직 일부의 뉴런만으로 표현될 수 있습니다. 이러한 방식의 인코딩 방법을 '희소분포표상'이라고 부릅니다. "희소"라는 말은 오직 적은 비율의 뉴런이 한 번에 활성화됨을 의미합니다. "분포"는 무언가를 표현하기 위해서는 여러 뉴런들이 필요함을 의미합니다. 단 하나의 활성화된 뉴런도 어떠한 의미를 전달하지만, 전체적인 의미를 이해하기 위해서는 활성화된 뉴런 집단의 맥락을 알아야만 합니다.

HTM 의 영역들 역시 희소분포표상을 이용합니다. 사실, HTM 영역 내에서의 기억 메커니즘은 오직 희소분포표상에 의해 이루어지며, 다른 방식으로는 작동하지 않습니다. HTM 영역으로의 입력은 언제나 분포된 형태로 들어오지만 희소한 형태는 아닐 수도 있습니다. 따라서, HTM 영역이 첫 번째로 하는 것은 자신이 받은 입력을 희소분포표상의 형태로 변환하는 것입니다.

예를 들어, 하나의 영역이 20,000 비트의 입력을 받는다고 가정해 봅시다. 입력 비트 중의 "1"과 "0"의 비율은 시간에 따라 매우 크게 변화할 수 있습니다. 한 순간은 5,000 개의 "1"이 있지만, 다른 때에는 9,000 개의 "1"이 있을 수 있습니다. HTM 영역은 얼마나 많은 입력 비트가 "1"인가에 상관없이, 10,000 비트의 내부 표상 중 오직 2%, 200 개를 활성화시키면서 모든 입력을 표현해낼 수 있습니다. HTM 영역으로 들어오는 입력이 시간에 따라 변화하면, 내부의 표상들도 따라 변화하지만, 오직 10,000 개 중 200 개의 비트만이 활성화된 상태를 유지하게 됩니다.

이러한 과정이 정보의 커다란 손실을 일으키는 것처럼 보일 수도 있는데, 특히 가능한 입력 패턴의 숫자가, 영역 내의 표상만으로 표현할 수 있는 수보다 훨씬 더 클 경우에 특히 그렇습니다. 하지만, 두 가지 숫자 모두 믿을 수 없이 크다는 점이 중요합니다. 하나의 영역이 받는 실제 입력은 모든 가능한 입력들의 극히 일부에 불과합니다. 이후에 우리는 어떻게 하나의 영역이 자신이 받은 입력에서 희소분포표상을 형성하는지 살펴볼 것입니다. 정보의 이론적 손실은 실제적으로는 거의 영향을 끼치지 않습니다.

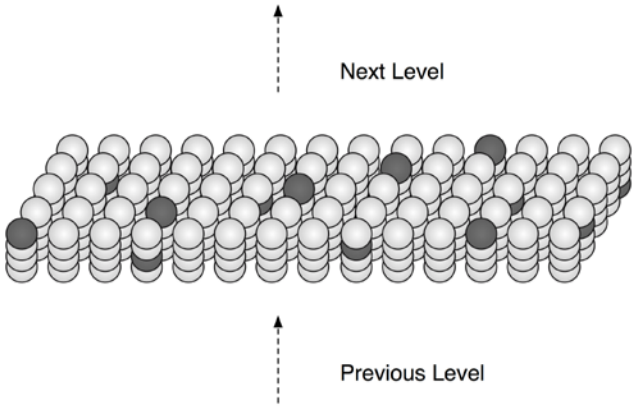


Figure 1.4: 희소하게 분포된 셀 활성화를 보여주는 HTM 영역.

희소분포표상은 여러 유리한 특징을 가지고 있으며, HTM 을 운영하는데 있어 필수적입니다. 자세한 내용은 이후에 다시 다루기로 하겠습니다.

시간의 역할

시간은 학습과 추론 그리고 예측에 있어 정말 중요한 역할을 합니다.

추론부터 시작해보죠. 시간 없이는, 우리는 우리의 촉감과 청각으로부터 아무것도 알아낼 수가 없습니다. 예를 들어, 우리가 눈을 가린 상태에서, 누군가가 손 위에 사과를 올려놓았다고 가정해봅시다. 우리는 사과를 이리저리 만져본 후에, 몇 초 후면 그것이 무엇인지 알 수 있습니다. 사과 위로 손가락을 움직이면, 촉각 정보가 계속 변화하지만, 사과 그 자체와 "사과"라는 고차원적인 인식에는 변화가 없습니다. 하지만, 쭉 편 손바닥 위에 사과를 올려논 상태에서, 손이나 손가락을 움직일 수 없다고 한다면, 우리는 그것이 사과인지 아니면 레몬인지를 구별하는데 커다란 어려움을 겪게 됩니다.

듣는 것도 마찬가지입니다. 멈춰있는 소리는 아무런 의미도 전달하지 못합니다. "사과"와 같은 단어나 사과를 깨물어먹는 소리 등은 오직 수 십, 수 백개의 빠르고 시간에 따라 변화하는 소리를 통해서만 인식할 수 있습니다.

반면 시각은, 조금 섞여 있습니다. 촉각, 청각과는 다르게, 우리는 눈 앞에서 빠르게 나타났다 사라진 물체들도 인식할 수 있습니다. 따라서, 시각적 추론은 항상 시간에 따라 변화하는 입력을 필요로하지는 않습니다. 하지만, 정상적인 시각인식에서 우리는 눈과 머리, 몸을 끊임없이 움직이며, 우리 주위의 물체들도 마찬가지입니다. 아주 짧은 시각 노출만으로도 대상을 추론할 수 있는 것은, 수 년간의 훈련을 통해 습득한 통계적 특성에 기초한 특별한 경우입니다. 일반적인 경우라면 시각, 청각 그리고 촉각 모두 시간에 따라 변화하는 입력을 필요로 합니다.

추론의 일반적인 경우와 시각의 특수한 경우를 모두를 살펴보았으므로, 학습에 대해 살펴보기로 하죠. 배우기 위해서, 모든 HTM 시스템은 훈련 과정동안 시간에 따라 변화하는 입력을 받아야만 합니다. 정적인 추론이 어느정도 가능한 시각조차도, 우리는 그 물체가 어떻게 보이는지 배우기 위해서는 변화하는 이미지를 보아야만 합니다. 예를 들어, 한 마리의 개가 우리를 향해 달려오고 있다고 생각해봅시다. 매 순간마다, 개의 모습은 우리 눈의 망막 위에 변화하는 상을 만들어 냅니다. 우리는 이러한 패턴들을 같은 개의 다양한 각도에서의 모습으로 인식하지만, 수학적으로 그 패턴들은 전혀 닮아있지 않습니다. 우리의 뇌는 이 서로 다른 패턴들이 같은 물체를 의미한다는 것을 시간에 따라 관찰함으로써 배우게 됩니다. 시간은 일종의 "감독관"이며, 우리에게 어떠한 공간적 패턴이 함께 일어나는지를 가르쳐 줍니다.

감각 입력이 단순히 변화하는 것만으로는 충분하지 않다는 점이 중요합니다. 서로 연관이 없는 일련의 감각패턴들은 오히려 혼란을 불러일으킵니다. 시간에 따라 변화하는 입력은 반드시 어떤 공통된 물체로부터 비롯되어야 합니다. 여기에서는 인간의 감각을 예로 들었지만, 인간의 감각이 아닌 일반적인 경우도 마찬가지입니다. 만약 HTM 을 발전소의 온도나 진동 또는 소음 패턴을 인식하는데 이용한다면, HTM 은 이러한 센서로부터 시간에 따라 변화하는 입력을 통해 훈련받아야 합니다.

일반적으로, HTM 네트워크를 훈련시키기 위해서는 수많은 데이터가 필요합니다. 우리가 개를 인식하기 위해서는 많은 종의 개들을 여러 상황에서 봐야만 하며, 단 한 마리의 개의 단 하나의 모습을 통해서는 학습이 이루어질 수 없습니다. HTM 알고리즘의 역할은 시간적으로 변화하는 일련의 입력 데이터로부터 시간적인 시퀀스를 배우는 것입니다. 예를 들면, 하나의 패턴 뒤에 어떠한 패턴이 나타날지에 대한 모델을 만들어내는 것이죠. 언제 시퀀스가 시작되고 끝나는지 알 수 없고, 같은 시간에 반복된 시퀀스가 나타날 수도 있으며, 학습이 지속적으로 이루어져야할 뿐 아니라 노이즈가 있는 상태에서 학습해야할 수도 있기 때문에, 이러한 역할에는 어려움이 따릅니다.

학습과 시퀀스를 인식하는 것은 예측을 만들어내는 기반으로 작용합니다. HTM 이 하나의 패턴 뒤에 어떠한 패턴이 나타나는지를 배우고 나면, 현재의 입력과 바로 직전의 입력을 통해 다음에 어떠한 패턴이 나타날지 예측할 수 있습니다. 예측에 관련된 내용은 뒤에서 더 자세히 다루도록 하겠습니다.

이제 HTM 의 네 가지 기본 기능: 학습, 추론, 예측 그리고 행동에 대해 알아보겠습니다. 모든 HTM 영역은 세 가지 기능: 학습, 추론 그리고 예측을 수행합니다. 하지만 행동은 조금 다릅니다. 생물학을 통해, 대부분의 피질 영역이 행동을 만들어내는 역할을 가지고 있다는 것을 알지만, 여러 흥미로운 응용분야에 필수적인 것은 아닙니다. 따라서 현재의 HTM 구현에는 행동을 포함시키지 않았습니다. 하지만 완전성을 위해 여기서 언급해보기로 하죠.

학 습 (Learning)

하나의 HTM 영역은 입력된 데이터로부터 패턴과 시퀀스를 찾음으로써 세상에 대해 배워갑니다. 하지만 그 영역은 그 입력들이 무엇을 표현하는지는 “깨닫고” 있지 못합니다. 오직 순수한 통계적 방식으로 작동할 뿐이죠. 먼저 HTM 은 공간적 패턴이라 불리는, 입력 비트 중에 함께 일어나는 조합을 찾습니다. 그리고 이렇나 공간적 패턴들이 시간에 따라 어떻게 변화하는지를 살피는데, 이를 시간적 패턴 혹은 시퀀스라고 부릅니다.
만약 어떤 영역의 입력이 빌딩의 환경 센서로부터 들어온다면, 그 영역은 빌딩 북쪽 구역에서 자주 일어나는 온도와 습도 조합을 찾아낼 수 있으며, 남쪽 구역에서는 또다른 조합을 찾아내게 됩니다. 그리고 날이 지남에 따라 이러한 조합이 어떻게 변화하는지를 배울 수 있습니다.

만약 HTM 영역의 입력이 상점에서의 물품판매량을 나타낸다면, HTM 은 주말에 주로 팔리는 물품들을 발견하거나, 날씨가 추운 저녁에는 특정 가격대의 물품들이 선호된다는 것을 발견할

수도 있습니다. 그리고 나면, 서로 다른 사람들이 그들의 구입 패턴에서는 비슷한 패턴을 보인다는 것을 배울 수도 있습니다.

하나의 HTM 영역은 제한적인 학습 능력을 가집니다. 한 영역이 가진 메모리 용량과 입력의 데이터의 복잡도에 따라 무엇을 배울지가 자동적으로 조절됩니다. 할당된 메모리 용량이 줄어들면, 어떤 영역이 배우는 공간적 패턴은 필수적으로 간단해 집니다. 또는 메모리 용량이 늘어나면, 공간 패턴들은 더 복잡해질 수도 있습니다. 만약 학습한 패턴이 간단하다면, 복잡한 이미지를 이해하기 위해서는 더 많은 계층구조가 필요합니다. 인간의 시각 시스템에서도 비슷한 특징을 발견할 수 있는데, 망막으로부터 입력을 직접 받는 신피질 영역은 시야의 아주 작은 부분에서 일어나는 공간 패턴을 배우게 됩니다. 여러 단계의 계층구조를 거친 후에야, 공간 패턴들이 결합되어 시야의 대부분 또는 전부를 표상할 수 있습니다.

생물학적 시스템과 마찬가지로, HTM 알고리즘 역시 "실시간 학습"-매 순간 새로운 입력으로부터 연속적으로 학습하는-을 할 수 있습니다. 추론 단계와 독립된 학습 단계과 필요하지는 않으며, 추론은 학습 후에 더 정교해질 수 있습니다. 입력되는 패턴이 달라지면, HTM 영역 역시 계속해서 변화합니다.

초기 훈련 이후에, HTM 은 계속해서 배울 수 있으며, 또는, 훈련 단계 이후에는 학습을 중지시킬 수도 있습니다. 또다른 방법은 아랫단계에서는 학습기능을 끄고, 더 높은 단계에서는 학습을 시킬 수도 있습니다. HTM 이 자신의 세상의 기본적인 속성에 대해 한번 학습하고 나면, 대부분의 새로운 학습은 계층구조의 상위 단계에서 일어납니다. 만약 HTM 이 이전에 아랫단계에서 한 번도 보지못한 입력을 마주하게 되면, HTM 이 이 새로운 패턴을 배우는데는 더 많은 시간이 필요합니다. 비슷한 특징을 우리에게서도 찾을 수 있습니다. 모국어에서 새로운 단어를 배우는 것은 상대적으로 쉽습니다. 하지만, 익숙하지 않은 발음의 외국어에서 새로운 단어를 배우는 것은 훨씬 더 어렵게 느껴지는데, 이는 그 언어의 아랫단계 정보 (소리)를 알지 못하기 때문입니다.

단지 패턴들을 찾는 것도 매우 중요한 능력입니다. 높은 단계 패턴들, 시장의 변동과 질병, 날씨, 산업 이윤 또는 전력망과 같이 복잡한 시스템의 고장을 이해하는 것은 그 자체로 매우 중요한 일입니다. 그렇다 하더라도, 공간-시간적인 패턴을 배워야만 추론과 예측을 만들어낼 수 있습니다.

추 론 (Inference)

HTM 이 자신의 세상에서 일어나는 패턴들을 학습한 후에는, 새로운 입력에 대한 추론을 만들어낼 수 있습니다. HTM 이 입력을 받으면, 그것을 이전에 학습한 공간, 시간적 패턴과 비교합니다. 새로운 입력을 이전에 저장된 시퀀스들과 짝짓는 것이 추론에서 가장 중요한 부분입니다.

우리가 어떤 멜로디를 듣는 경우를 생각해보죠. 첫 번째 음만 들어서는 그 멜로디에 대해 아무것도 알 수 없습니다. 두 번째 음을 들으면, 어떤 멜로디인지 조금 더 알 수 있지만, 여전히 충분하지 않습니다. 보통 세, 네 개 또는 그 이상의 음을 들어야만 특정 멜로디로 인식할 수 있습니다. HTM 영역에서 일어나는 추론도 비슷합니다. HTM 은 일련의 입력들을 끊임없이 살피고, 이전에 배운 시퀀스들과 계속해서 비교합니다. HTM 영역은 자신이 가진 시퀀스의 시작 부분들과 유사한 패턴을 찾을 수도 있지만, 보통은 더 유연성을 가지고 작동합니다. 이는 우리가 멜로디의 어느 부분에서 시작하더라도 특정 음악을 알아차릴 수 있는 것과 비슷합니다. HTM 영역이 분포된 표상을 이용하기 때문에, 각 영역이 이용하는 시퀀스들은 멜로디보다는 훨씬 더 복잡하지만, HTM 의 동작 원리를 이해하는데는 도움이 될 수 있습니다.

우리가 보고들은 모든 감각 경험들이 완전히 새로운 것이라도, 우리가 거기서 익숙한 패턴을 찾을 수 있다는 점이 쉽게 이해되지 않을 수도 있습니다. 예를 들어, 우리는 “아침”이라는 단어를 말하는 사람의 나이나 성별, 또는 말하는 속도나 억양에 상관없이 거의 대부분 알아들을 수 있습니다. 똑같은 사람이 “아침”이라는 단어를 수 백번 이야기하더라도, 귀를 통해 전달되는 소리가 정확히 동일한 경우는 없습니다.

HTM 영역도 이와 비슷한 상황을 겪습니다. 입력이 똑같이 되풀이되는 경우는 결코 없습니다. 따라서, 우리의 뇌와 마찬가지로, HTM 영역은 추론과 훈련단계에서 새로운 입력을 마주쳐야만 합니다. HTM 영역이 이러한 새로운 입력에 대처하는 방법이 바로 희소분포표상입니다. 희소분포표상의 핵심은 짝이 맞는 패턴을 찾기 위해서 오직 패턴의 일부분만 비교해보면 된다는 점입니다.

예 측 (Prediction)

HTM 의 모든 영역들의 일련의 패턴을 기록합니다. 자신이 가진 시퀀스와 입력을 비교함으로써, HTM 은 바로 다음에 어떠한 입력이 들어올지에 대한 예측을 만들어냅니다. HTM 영역은 희소분포표상들의 전환도 저장할 수 있습니다. 때때로 이러한 전환은 멜로디의 음들에서처럼 선형적일 수도 있지만, 대부분의 경우에 특정시점에서 미래에 들어올 입력은 여러가지가

존재합니다. HTM 영역이 더 과거의 패턴들로부터 맥락을 형성할수록, 서로 다른 예측들이 만들어질 수도 있습니다. HTM 메모리에 저장된 대부분의 정보는 여러 시퀀스거나 공간적 패턴들 사이의 전환과 관련된 것입니다.

HTM 예측의 주요 특징들이 아래에 요약되어 있습니다.

1) 예측은 연속적이다.

우리가 의식하고 있지 못하지만, 우리는 끊임없이 예측을 만들어내고 있습니다. HTM 도 마찬가지입니다. 음악을 들을 때, 우리는 다음 음을 예측합니다. 계단을 내려갈 때, 우리는 발이 언제 다음 계단에 닿을지 예측합니다. 투수가 공을 던지는 것을 볼 때, 우리는 공이 타자 근처로 갈 것인지에 대해 예상합니다. HTM 영역에서의 예측과 추론도 거의 마찬가지입니다. 예측은 하나의 독립적인 과정이 아니라, HTM 이 동작하는 모든 과정과 연관되어 있습니다.

2) 예측은 계층구조의 모든 영역과 모든 단계에서 일어난다.

HTM 영역의 계층구조에서, 예측은 모든 단계에서 일어납니다. 영역들은 그들이 배운 패턴들에 대한 예측을 만들어냅니다. 언어의 예에서, 하위 단계의 영역들은 다음에 나타날 음절을 예측하고, 더 높은 단계의 영역들은 단어나 구문을 예측합니다.

3) 예측은 맥락과 밀접한 연관을 가진다.

예측은 지금 일어나는 일 뿐 아니라, 과거에 어떠한 일이 일어났는지를 바탕으로 합니다. 따라서 과거의 맥락에 따라서 서로 다른 예측들이 만들어질 수 있습니다. HTM 영역은 필요한만큼 과거의 맥락들을 이용하는 방법을 배우며, 가까운 과거부터 먼 과거를 모두 이용하여 맥락을 형성할 수 있습니다. 이러한 능력을 "다차원" 메모리로 부릅니다. 예를 들어, 게티스버그 연설 (Gettysburg Address) 원고를 생각해봅시다. 이 글에서 다음 단어를 예측하기 위해서는, 현재의 단어를 아는 것만으로는 부족합니다; 첫 번째 문장에서, "그리고"라는 단어 뒤에 "일곱"이라는 단어가 나오는 경우도 있지만, "헌정된"이라는 단어가 나오는 경우도 있습니다. 때때로, 아주 사소한 맥락이라도 예측에는 도움이 되는 경우가 있습니다; "80(four scores)" 이라는 구문을 알면 "7(seven)"이라는 단어를 예측하는데 도움이 됩니다. 반면, 같은 구문이 계속 반복되는 경우에, 우리가 연설에서 어느 부분에 있고, 다음에 무엇을 말할지 알기 위해서는 더 긴 시간에 걸쳐 맥락을 만들어내는 것이 필요합니다.

4) 예측은 안정성을 이끌어낸다.

하나의 영역에서의 출력은 자신이 만들어낸 예측입니다. HTM 의 특징 중 하나는 영역들의 출력이 더 안정된다는 것입니다. 따라서 계층구조를 위로 올라갈수록, 출력은 더 천천히 변화하고 더 오랫동안 지속됩니다. 이러한 특징은 영역이 어떠한 예측을 만들어내느냐에 달려있습니다. 하나의 영역은 바로 다음에 일어날 일만을 예측해서는 충분하지 않습니다. 만약

할 수 있다면, 더 멀리 일어날 일까지 예측해야 합니다. 만약 어떤 영역이 5 단계를 앞서 예측할 수 있다고 가정해보죠. 새로운 입력이 들어오면, 새로운 한 단계의 예측은 변화하지만 과거에 만든 4 단계의 예측은 아직 그대로입니다. 그러므로, 새로 들어온 입력이 예측과 완전히 다르더라도, 출력의 일부분만이 변화하며, 따라서 출력은 입력보다 더 안정될 수 있습니다. 이러한 특징은 우리의 일상생활에서의 경험과도 일치합니다. 노래의 제목과 같은 높은 단계의 예측은, 실제 음이나 음악같은 낮은 단계의 입력보다 더 느리게 변화합니다.

5) 예측은 새로 들어온 입력이 예측했던 것인지 아닌지를 알려준다.

각각의 HTM 영역은 새로운 패턴 감지기입니다. 각각의 영역이 다음에 무엇이 일어날지를 예측하기 때문에, 무언가 예측하지 않은 일이 일어나는 것을 "알" 수 있습니다. HTM 은 동시에 한 개가 아닌, 여러 가능한 입력들을 예측할 수 있습니다. 따라서 다음에 무슨 일이 일어날지를 정확히 예측하지 못하더라도, 다음에 들어온 입력이 자신이 예측한 목록에 없다면, HTM 영역은 무언가 비정상적인 일이 일어났음을 알 수 있습니다.

6) 예측은 노이즈로부터 시스템을 더 견고하게 만듭니다.

HTM 이 다음에 무엇이 일어날지를 예측한다면, 그것이 예측한 방향으로 시스템이 추론하도록 할 수 있습니다. 예를 들어, 만약 HTM 이 음성언어를 처리한다면, 어떠한 소리와 단어 그리고 생각들을 다음에 들어올지 예상할 수 있습니다. 이러한 예측은 무언가 손실된 데이터들을 채워넣을 수도 있습니다. 만약 모호한 소리가 들린다면, HTM 은 자신이 예측하고 있는 소리를 바탕으로 추론할 수 있으며, 따라서 노이즈가 있더라도 추론에 도움을 줄 수 있습니다.

HTM 영역에서, 시퀀스 메모리, 추론 그리고 예측은 밀접하게 통합되어 있습니다. 그리고 그것이 한 영역의 핵심 기능입니다.

행 동 (Behavior)

우리의 행동은 우리의 인식에 영향을 미칩니다. 우리가 눈을 움직이면, 망막에는 변화하는 입력들이 들어옵니다. 팔과 손가락을 움직이면, 그에 따라 변화하는 촉각들이 뇌로 전달됩니다. 우리가 하는 거의 모든 행동들이 우리의 감각을 변화시킵니다. 따라서, 감각 입력과 운동은 밀접하게 뒤섞여 있습니다.

수 십년동안, 뇌과학의 지배적인 이론은, 일차운동영역 (primary motor cortex)이라고 불리는 영역이, 신피질에서 운동 명령을 내리는 유일한 곳이라고 생각했습니다. 연구가 진행됨에 따라, 신피질의 대부분 또는 모든 영역, 심지어 하위 단계의 감각 영역들도 운동 출력을 가진다는 점이 발견되었습니다. 모든 피질 영역들이 감각과 운동 기능을 통합하고 있는 것으로 보입니다.

따라서 우리는 현재 HTM 의 이론적인 뼈대에 운동과 관련된 출력을 더할 수 있을 것으로 기대하는데, 운동 명령을 만들어내는 것이 예측을 하는 것과 유사하기 때문입니다. 하지만, 지금 구현되어 있는 HTM 은 운동과 관련된 부분 없이, 순수하게 감각만을 다루고 있습니다.

HTM 을 구현하기 위한 발전

우리는 HTM 의 이론적인 틀들을 실제적인 기술로 발전시킬 수 있었습니다. HTM 피질 학습 알고리즘의 여러 버전을 개발하여 테스트하였고, 기본적인 설계가 잘 작동함을 확인하였습니다. 우리는 새로운 데이터를 이용해 알고리즘을 테스트하면서, 빠진 부분들을 더하고 알고리즘을 개선하였습니다. 그리고 그에 따라 이 문서도 개정하였습니다. 앞으로 다룰 세 개의 장들이 알고리즘의 최신 상황을 기술하고 있습니다.

아직 이론의 많은 부분들이 구현되지 않았습니다. 여기에는 집중 (attention), 여러 영역들 간의 되먹임 (feedback), 정확한 시간적 타이밍 그리고 행동과 운동의 연합 등이 포함됩니다. 이러한 부분들이 기존에 존재하는 틀에 잘 맞아들어갈 수 있으리라 생각합니다.

제 2 장: HTM 피질 학습 알고리즘

이 장은 HTM 영역 안에 구현된 세부적인 알고리즘을 다루고 있습니다. 이번 장은 개념적인 부분을 주로 다루었으며, 3 장과 4 장에서는 가(pseudo) 코드를 이용하여 학습 알고리즘을 구현하였습니다.

용 어 (Terminology)

시작하기 전에, 용어를 정리하는 것이 좋을 것 같습니다. 우리는 HTM 학습 알고리즘에 뇌과학의 용어들을 사용하였습니다. 셀이나 시냅스, 잠재적 시냅스 (potential synapse), 수상돌기 (dendrite segments) 그리고 칼럼구조 (column)와 같은 용어들이 계속해서 사용됩니다. 실제 알고리즘이 대부분 뇌과학의 세부적인 지식들로부터 구현되었기 때문에 이러한 용어를 사용하는 것이 더 적절합니다. 하지만, 알고리즘을 실제로 실행할 때는 성능이 중요하기 때문에, 작동하는 방식을 이해하고 난 다음에는 정보를 더 빠르게 처리할 수 있는 방법을 이용했습니다. 비슷한 결과를 얻을 수 있다면 뇌의 생물학적인 세부사항들에서 조금 비껴난 부분도 있습니다. 뇌과학이 아직 낯설다면 이러한 부분은 큰 문제가 되지 않습니다. 하지만, 뇌과학의 용어에 이미 익숙하다면, 여러분이 생각하는 용어와 조금 차이가 있어 혼란이 생길수도 있습니다. 이를 위해, HTM 학습 알고리즘에 사용된 용어와 생물학에서 쓰는 용어의 비슷한 점과 다른 점을 부록에 따로 상세히 정리해놓았습니다. 여기에는 가장 큰 혼란을 불러일으킬수 있는 몇 가지 용어들에 대해서만 언급하도록 하겠습니다.

셀 상태 (Cell states)

HTM 의 셀은 세 개의 출력 상태를 가지는데, 아랫단계에서의 입력 (feed-forward input)에 의한 활성화, 같은 단계에서의 입력 (lateral input)에 의한 활성화 그리고 비활성화 상태입니다. 첫 번째는 실제 뉴론에서 짧은 시간동안 반복해서 일어나는 활동전위에 해당합니다. 두번째 상태는 조금 더 느리게 지속되는 활동전위와 유사합니다. 우리는 이 두 활성상태 이외에, 각각의 뉴런이 보이는 활동전위나 활성정도를 스칼라 값으로 표현하는 것은 크게 필요하지 않다는 점을 알게 되었습니다. 실제 뉴런의 다양한 활성정도를 이용하지 않고도, 분포표상을 통해 비슷한 결과를 얻을 수 있었습니다.

수상 돌기 (Dendrite segments)

HTM 의 셀들은 실제와 거의 비슷한 (그래서 복잡한) 수상돌기 모델을 가지고 있습니다. 이론적으로 각각의 HTM 셀은 하나의 몸쪽 수상돌기 (proximal dendrite segment)와 수 십개의 먼쪽 수상돌기 (distal dendrite segment)를 가지고 있습니다. 몸쪽 수상돌기는 아랫단계에서 전해진 입력을 받으며, 먼쪽 수상돌기는 근처의 셀들이 보내는 같은 단계에서 들어오는 입력을 받습니다. 억제 기능을 가진 셀들이 같은 칼럼구조에 있는 셀들에 작용하여 비슷한 아랫단계

입력에 대해 함께 반응하도록 합니다. 간단히 말하면, 각각의 셀에 있는 몸쪽 수상돌기를 제거하는 대신, 하나의 칼럼구조에 속한 여러 셀들이 하나의 공통된 수상돌기를 가지도록 하였습니다. 아래에서 자세히 설명할 공간풀러 (spatial pooler) 기능이 이 공통된 수상돌기에 칼럼구조 단위로 작용합니다. 시간풀러 (temporal pooler) 기능은 먼쪽 수상돌기에 적용되며, 칼럼구조 안의 개별 셀들에까지 적용됩니다. 비록 실제 뉴런에는 공통된 수상돌기라는 구조가 없지만, 이러한 간소화를 통해 실제 뉴런들과 비슷한 기능을 할 수 있습니다.

시냅스 (Synapses)

HTM 시냅스는 이진법(0, 1)을 이용합니다. 생물학적인 시냅스는 다양한 값을 가지는 동시에 꽤 확률적이기 때문에, 정확한 하나의 값에 의존하지는 않는 것으로 보입니다. HTM 에서 이용하는 분포표상에 위에서 설명한 수상돌기의 모델을 적용하면, HTM 시냅스들이 별 무리없이 이진수의 값들만 가지도록 할 수 있습니다. 시냅스 연결을 맺고 끊는데에는, 뇌과학에서도 잘 알려진 두 가지 개념이 이용됩니다. 그 중 하나가 "잠재적 시냅스 (potential synapse)"입니다. 수상돌기 근처에 위치하는 모든 축삭돌기(axon)들은 시냅스와 연결을 형성할 수 있습니다. 두 번째 개념은 "지속성 (permanence)"입니다. 지속성은 축삭돌기와 수상돌기가 어느 정도로 결합되어 있는지를 스칼라 값으로 표현합니다. 생물학적으로, 두 연결은 완전히 끊어져있거나, 이제 막 시냅스 형성을 시작했지만 아직 연결이 되지 않았거나, 가장 약하게 연결된 경우에서부터 아주 강하게 연결된 경우까지 가능합니다. 따라서 어떤 시냅스의 지속성 값은 0.0 부터 1.0 까지 다양하게 분포합니다. 학습한다는 것은 바로 이 시냅스의 지속성 값을 증가하거나 감소시키는 것을 의미합니다. 시냅스의 지속성 값이 한계치를 넘으면, 그 시냅스를 연결되었다고 하고 "1"의 값을 할당합니다. 한계치 미만인 경우에는, 연결되지 않았으며, "0"의 값을 가지게 됩니다.

개 론 (Overview)

우리가 HTM 의 한 영역이라고 생각해봅시다. 우리가 받는 입력은 수 천, 수 만개의 비트로 이루어져 있습니다. 이러한 입력들이 감각 데이터일 수도 있고, 계층 구조의 더 아랫단계에서 전달된 입력일 수도 있습니다. 이들이 시간에 따라 켜지고 꺼지면서 복잡하게 변화합니다. 우리가 이 입력을 가지고 무엇을 할 수 있을까요?

우리는 앞에서 이 문제의 답에 대해 간략히 다룬 적이 있습니다. 각각의 HTM 영역은 입력으로부터 어떤 공통된 패턴을 찾고, 이 패턴들로부터 시퀀스를 학습합니다. 이 기억된 시퀀스를 통해, 각 영역은 예측을 만들어냅니다. 이렇게 말로만 설명하는 것은 쉬워보이지만, 실제로 구현하는데는 많은 과정이 필요합니다. 이러한 과정들을 아래와 같이 세 단계로 나누어 보죠.

1) 입력으로부터 희소분포표상을 형성한다.
2) 과거의 입력을 통해 형성된 맥락을 통해 표상을 만든다.
3) 과거의 입력을 통해 형성된 맥락에 현재 입력을 더하여 예측을 만들어낸다.

이제 이 여러 단계들을 더 자세히 살펴봅시다.

1) 입력으로부터 희소분포표상을 형성한다.

하나의 영역에 전달된 입력이, 아주 커다란 비트들이라고 생각해보죠. 실제 뇌에서는 뉴런에서 나오는 축삭돌기에 해당할 것입니다. 어떤 때에 이 입력 비트의 값은 활성 상태 (값 1)일 수도 있고 다른 때에는 비활성화 상태 (값 0)일 수도 있습니다. 입력으로 들어오는 비트들 중에서 활성 상태인 비트의 비율은 다양하게 변할 수 있습니다 (예를 들면, 0~60%). HTM 영역이 첫 번째로 하는 일은 이러한 입력들을 희소 (sparse)하게 분포하는 형태로 변화시키는 것입니다. 예를 들어, 입력 값 중에 40%가 "활성" 상태였다면, 새로운 희소 표상은 단지 2%만 "활성"화된 상태로 바꾸는 것입니다.

하나의 HTM 영역은 여러 세트의 *칼럼*들로 이루어져 있습니다. 각각의 칼럼들은 여러 개의 *셀*들로 구성되어 있습니다. 이론적으로 칼럼들은 2 차원 배열 구조를 가질 수 있지만, 반드시 그런 것은 아닙니다. 영역 안의 각 칼럼들은, 입력 비트 고유의 집합들과 연결되어 있습니다. (하나의 칼럼이 다른 칼럼과 겹치는 입력을 가질 수는 있지만, 결코 완전히 같은 입력을 받지는 않습니다.) 따라서, 서로 다른 입력 패턴은 각 칼럼들에게 서로 다른 활성 상태를 불러 일으킵니다. 가장 강하게 활성화된 칼럼들은 덜 활성화된 칼럼들을 억제하거나 비활성화 시킵니다. (이러한 억제 현상은 한 영역의 아주 작은 부분부터 전체 영역까지 다양한 범위에 걸쳐 영향을 끼칠 수 있습니다.) 입력을 희소하게 분포시키는 것은, 억제 현상을 통해 어떠한 칼럼들은 활성 상태로 남아있는 반면, 다른 칼럼들은 비활성화되기 때문에 가능합니다. 어떻게 얼마나 억제시킬 것인가는 입력으로 들어오는 비트들이 매우 다양하게 변화하더라도, 상대적으로 일정한 비율의 칼럼들이 활성화되도록 하는 것과 관련이 있습니다.

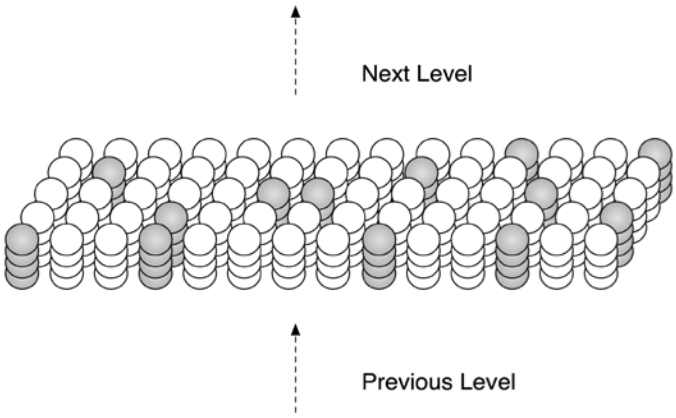


Figure 2.1: 하나의 HTM 영역은 셀로 구성된 칼럼들로 이루어집니다. 그림에는 한 영역의 아주 일부분만이 나타나 있습니다. 각 칼럼의 셀들은 입력의 고유한 집합들로부터 신호를 받습니다. 가장 강하게 활성화된 칼럼들이 더 약하게 활성화된 칼럼들을 억제합니다. 그 결과, 입력을 희소하게 분포시켜 나타낼 수 있습니다. 그림에 밝은 회색으로 나타난 것이 활성화된 칼럼들입니다. (만약 별다른 이전 정보가 없다면, 그림에 나타나 있듯이, 활성화된 칼럼 안의 모든 셀들이 활성 상태에 놓입니다.)

자 이제 입력 패턴이 변화하는 것을 생각해보죠. 만약 아주 일부 비트들만이 변한다면, 어떤 칼럼들은 조금 많거나 적은 수의 "활성" 상태의 입력을 받을 수 있지만, 활성화된 칼럼들의 전체 집합은 크게 변화하지 않을 것입니다. 따라서 비슷한 입력 패턴들은 (대부분의 활성화된 비트들을 공통으로 가지고 있는) 꽤 안정되게 칼럼들을 활성화 시킵니다. 얼마나 안정화되어있는지는 각각의 칼럼들이 어떠한 입력에 연결되어 있는지에 따라 결정됩니다. 이러한 연결관계는 이후에 설명할 방법을 통해 학습됩니다.
이 모든 단계들 (입력의 소집합으로부터 각 칼럼들에 연결관계를 형성하고, 각 칼럼에 어느 정도의 입력을 할당할지, 그리고 활성화된 칼럼들을 억제시켜 희소분포를 만드는 것)을 "공간풀러 (Spatial Pooler)"라고 부릅니다. 이는 "공간적"으로 유사한 (활성화된 비트의 대부분을 공유하는) 패턴들이 서로 "뭉치"는 것 (하나의 공통된 표상으로 모이는 것)을 의미합니다.

2) 이전 입력의 맥락을 바탕으로 입력에 대한 표상을 만든다.

한 영역이 다음으로 수행하는 기능은 기존에 입력된 표상을 새로운 형태의 표상으로 변환하는 것인데, 이는 과거의 맥락이나 칼럼의 상태 등을 바탕으로 이루어집니다. 새로운 표상은 각 칼럼구조에 속한 여러 셀들을 활성화시켜 형성되기도 하지만, 보통은 한 칼럼구조 당 하나의 셀만을 이용합니다.

다음과 같은 두 문장을 듣는 경우를 생각해보죠. "나는 배를 먹는다 (I ate a pear)"와 "나는 8 개의 배를 가지고 있다 (I have eight pears)." 두 문장에 사용된 "ate"와 "eight"는 동음이의어입니다; 똑같은 소리를 가지고 있죠. 우리는 뇌의 어느 단계에서는 "ate"와 "eight"에 동일하게 반응하는 뉴런들이 있으리라고 생각할 수 있습니다. 결국, 같은 소리가 귀를 통해 전달됩니다. 하지만,

뇌의 어느 부분에서는 이 동일한 입력에 대해 다르게 반응하는 뉴런들이 있으리라 예상할 수 있습니다. “ate”이라는 소리에 대한 표상은 우리가 “I ate”과 “I have eight…”을 들을 때 서로 다르게 형성됩니다. 우리가 두 개의 문장 “I ate a pear”와 “I have eight pears”을 암기하고 있다고 생각해봅시다. “I ate…”을 듣는 것과 “I have eight…”을 듣는 것은 서로 다른 예측을 만들어낼 것입니다. “I ate…”을 들은 후와 “I have eight…”을 들은 다음에는 분명 다른 내적인 표상이 만들어 집니다.

이처럼 서로 다른 맥락에 따라 입력을 다르게 코드화하는 원리는 우리가 세상을 지각하고 이해하는데 있어 일반적인 원리이며, HTM 영역에서도 가장 중요한 기능 중 하나입니다. 이러한 능력의 중요성은 아무리 강조해도 지나치지 않습니다.

HTM 영역의 각 칼럼들은 여러 개의 셀로 이루어져 있습니다. 하나의 칼럼 내의 모든 셀들은 같은 피드포워드 입력을 받습니다. 칼럼 안의 각 셀들은 활성 또는 비활성 상태에 있을 수 있습니다. 하나의 칼럼이 활성화되면, 그 안에서 활성화될 셀을 다르게 선택함으로서, 정확히 같은 입력에 대해서 서로 다른 맥락을 만들어낼 수 있습니다. 구체적인 예를 들어보기로 하죠. 모든 칼럼들이 4 개의 셀을 가지고 있고, 모든 입력을 100 개의 활성화된 칼럼으로 표현한다고 가정해봅시다. 만약 하나의 칼럼마다 한 개의 셀만이 활성화된다면, 우리는 똑같은 입력을 4^100 개의 서로 다른 방식으로 표현할 수 있습니다. 동일한 입력은 언제나 똑같은 100 개의 칼럼들을 활성화시키지만, 맥락이 달라지면, 칼럼 안에서 활성화되는 셀도 달라지게 됩니다. 따라서 우리는 똑같은 입력을 엄청나게 다양한 맥락으로 나타낼 수 있습니다. 하지만 각각의 서로 다른 표상들이 얼마나 유일할까요? 4^100 개의 가능한 패턴들 중에서 무작위로 짝을 고른다면, 그들은 대략 25 개의 셀들을 공유하게 될 것입니다. 따라서 특정 입력에서 서로 다른 맥락을 나타내는 표상들은, 약 25 개의 동일한 셀과 75 개의 서로 다른 셀을 가지게 됩니다. 따라서 이들을 손쉽게 구별해낼 수 있습니다.

HTM 영역이 이용하는 일반적인 원리는 다음과 같습니다. 하나의 칼럼이 활성화되면, 그 칼럼은 자신에게 속한 모든 셀들을 살펴봅니다. 만약 하나 또는 그 이상의 셀들이 이미 ‘예측상태 (predictive state)’에 있다면, 바로 그 셀들이 활성화 됩니다. 만약 어떠한 셀들도 예측상태가 아니라면, 모든 셀들이 활성화됩니다. 이러한 원리를 다음과 같이 이해할 수도 있습니다. 만약 입력된 패턴이 예측한 것과 동일하다면, 시스템은 예측상태에 있던 셀만 활성화시킴으로써 예측이 맞아들어갔음을 확인합니다. 만약 입력 패턴이 예측하지못한 것이었다면, 시스템은 칼럼 안의 모든 셀들을 활성화시키고 마치 다음과 같이 이야기합니다. “예상하지 못했던 입력이 들어왔으므로, 모든 가능한 해석이 유효하다.”

만약 이전에 들어온 입력이 하나도 없다면, 당연히 맥락이나 예측도 만들어질 수 없습니다. 이 경우에 하나의 칼럼이 활성화되면, 그 안의 모든 셀들이 활성화됩니다. 이러한 방식은 음악에서

첫 음을 듣는 것과 비슷합니다. 맥락없이 우리는 다음에 어떠한 일이 일어날지 예상할 수가 없습니다. 모든 가능성이 열려있는 것이죠. 하지만 이전에 어떤 입력이 있었고, 그것이 예측한 것과 다르다면, 활성화된 칼럼 안의 모든 셀들이 활성화될 것입니다. 이러한 방식은 칼럼구조 단위로 일어나기 때문에, 예측과 일치하거나 일치하지 않는 것이 "유 아니면 무"로 발생하지는 않습니다.

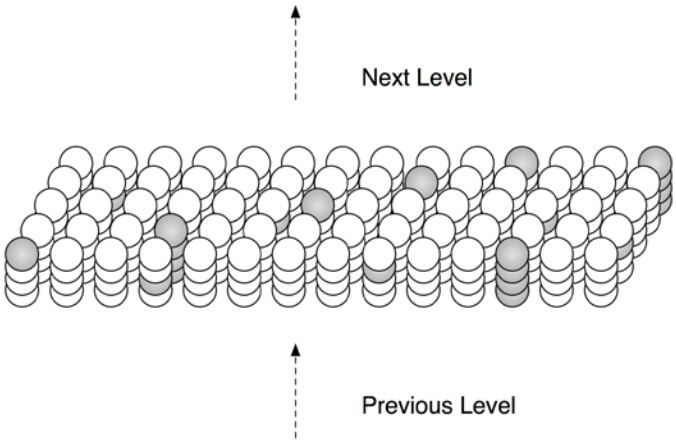


그림 2.2: 각 칼럼 안에 속한 일부 셀들을 활성화시킴으로써, HTM 영역은 같은 입력을 여러 개의 다른 맥락으로 표현할 수 있습니다. 칼럼들은 예측상태에 있었던 셀만 활성화시킵니다. 예측상태에 있는 셀이 없는 칼럼은, 자신이 가진 모든 셀을 활성화시킵니다. 위의 그림에서 어떤 칼럼은 하나의 셀만 활성화되어 있으며, 어떤 칼럼은 모든 셀들이 활성화되어 있습니다.

이전에 용어와 관련된 섹션에서 언급했듯이, HTM 에 속한 셀들은 셋 중 하나의 상태를 가집니다. 만약 어떤 셀이 피드포워드 입력에 의해서 활성화되면, 우리는 그 셀이 "활성화" 되었다고 말합니다. 만약 셀이 근처의 다른 셀들과의 연결을 통해 활성화되면, 그 셀이 "예측상태"에 있다고 말합니다. (그림 2.3.)

3) 현재 들어온 입력과 과거의 맥락을 바탕으로 예측을 만들어낸다.

마지막 단계는 다음에 무슨 일이 일어날지 예측을 만들어내는 것입니다. 예측은 단계 2)에서 형성된 표상을 바탕으로 일어나는데, 이 표상에는 모든 이전 입력들로부터 형성된 맥락이 포함되어 있습니다.

영역이 예측을 만들어낼 때에는, 미래에 들어온 입력에 의해 활성화될 것 같은 모든 셀들을 예측상태로 활성화시킵니다. 하나의 영역 안에서 표상은 희소하게 형성되기 때문에, 동시에 여러 개의 예측이 만들어질 수 있습니다. 예를 들어, 어떤 입력에 의해 2%의 칼럼들이 활성화되었고, 10 개의 다른 예측이 만들어진다면, 전체 칼럼들 중 20%가 예측상태에있는 셀을 가지게 됩니다. 또는 20 개의 예측이 만들어지면, 40%의 칼럼이 예측상태에 있는 셀을 가집니다.

만약 각각의 칼럼이 4 개의 셀을 가지고 있고, 한 번에 한 개만 활성화 된다면, 총 10%의 셀들이 예측상태에 놓이게 됩니다.

희소분포표상과 관련된 앞으로의 내용들은, 서로 다른 예측들이 한데 뭉치더라도, 특정 입력이 예측한 것인지 아닌지를 높은 정확도로 알 수 있음을 보여줍니다.

영역은 어떻게 예측을 만들어내는 것일까요? 입력이 시간에 따라 변하면, 서로 다른 조합의 칼럼과 셀들이 활성화됩니다. 하나의 셀이 활성화되면, 그것은 근처에 있는 셀들 중에서 바로 전에 활성화되었던 것들과 연결을 형성합니다. 이러한 연결은 특정상황에서 필요로하는 학습 속도에 따라 빠르게 혹은 느리게 형성될 수 있습니다. 이제 하나의 셀이 할 일은 자신이 가진 연결이 우연히 활성화되는지를 살피는 것입니다. 만약 특정 연결이 활성화되면, 그 셀은 곧 자신이 활성화될 것이라고 기대하고 예측상태로 들어갑니다. 따라서 피드포워드 입력에 의해 활성화된 셀들은 연이어 활성화될 셀들을 예측상태로 만듭니다. 이러한 방식은 우리가 음악을 처음 들었을 때, 다음 음을 예상하는 것과 유사합니다.

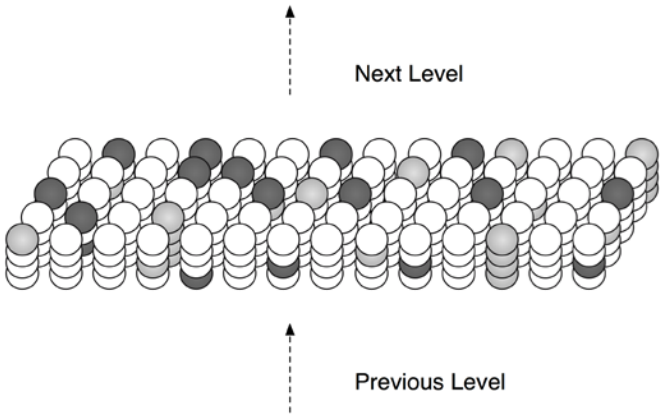


그림 2.3. 어느 한 순간에, HTM 영역의 어떤 셀들은 피드포워드 입력에 의해 활성화 됩니다 (밝은 회색으로 표시됨). 활성화된 셀로부터 같은 단계의 입력(lateral input)을 받은 셀들은 예측상태로 돌입합니다 (짙은 회색으로 표시됨).

요약하면, 새로운 입력은 칼럼들이 드문드문 분포하며 활성화되도록 만듭니다. 각 칼럼들마다 하나 혹은 그 이상의 셀들이 활성화되면, 연이어 학습된 연결에 의해 같은 영역 내의 다른 셀들이 예측상태로 들어가게 됩니다. 같은 단계의 연결을 통해 활성화된 셀들은 다음에 어떠한 일이 일어날지에 대한 예측을 의미합니다. 다음에 들어오는 새로운 피드포워드 입력은 다른 형태의 활성화된 칼럼집합을 형성합니다. 만약 새롭게 활성화된 칼럼이 예측한 것이 아니었다면, 즉 그 칼럼의 어느 셀도 자신이 활성화 되리라 예상하지 않았다면, 그 칼럼의 모든 셀들이 활성화됩니다. 만약 새롭게 활성화된 칼럼이 하나 혹은 그 이상의 예측상태의 셀을 가지고 있으면, 오직 그 셀들만이 활성화됩니다. 따라서 한 영역의 출력은 그 영역 안의 모든

셀들의 활성화 상태이며, 이는 피드포워드 입력에 의해 활성화된 셀과 *그리고* 예측상태에 있는 셀을 포함합니다.

이전에 언급했듯이, 예측이 *바로 다음* 시간단계에서만 일어나는 것은 아닙니다. HTM 영역에서의 예측은 미래의 여러 시간단계에 걸쳐 일어날 수 있습니다. 멜로디를 예로 들어보면, HTM 은 단지 바로 다음 음 뿐 아니라, 그 다음 4 개의 음들까지도 예측할 수 있습니다. 이렇게 멀리까지 예측하는 것은 중요한 속성입니다. 한 영역의 출력 (활성화 그리고 예측상태에 있는 모든 셀들의 집합)은 입력보다 더 느리게 변화합니다. 어떤 영역이 멜로디에서 4 단계 앞의 음까지 예상한다고 가정해보죠. 멜로디의 순서를 알파벳을 이용하여 A,B,C,D,E,F 로 표시해 봅시다. 처음 두 음을 듣고 나면, HTM 영역은 시퀀스를 인식하고 예측을 하기 시작합니다. 그는 C, D, E, F 를 예측했습니다. "B"와 관련된 셀은 이미 활성화되어 있기 때문에, B, C, D, E, F 와 관련된 셀들은 모두 활성화되거나 예측상태에 있습니다. 이제 HTM 영역이 다음 음인 "C"를 들었다고 해봅시다. 이제 활성화되거나 예측상태에 있는 셀들은 "C, D, E, F, G" 입니다. 입력이 "B"에서 C"로 완전히 바뀌었음에도 불구하고, 오직 20%의 셀들만 변화했음에 주목합시다.

HTM 영역의 출력이 영역 내의 모든 셀들의 활성화 상태를 벡터로 표현하기 때문에, 위의 예에서 출력은 입력에 비해 5 배나 더 안정화되어 있습니다. 영역들의 계층구조를 따라 위로 올라가면, 시간적인 안정성이 증가하는 것을 볼 수 있습니다.

우리는 기존의 표상에 맥락을 더하는 과정과 예측하는 과정을 설명하기 위해 "시간풀러 (temporal pooler)"라는 용어를 사용할 것입니다. 일련의 패턴들로부터 느리게 변화하는 출력을 만들어냄으로써, 우리는 연이어 나타나는 서로 다른 패턴들을 본질적으로 "뭉칠(pooling)" 수 있습니다.

이제 조금 더 자세한 과정으로 넘어가봅시다. 공간풀러와 시간풀러에서 공유하는 개념들부터 다루도록 하겠습니다. 그 이후에, 공간풀러에만 특화된 개념이나 세부사항을 다루고, 시간풀러의 세부적인 사항을 이야기하도록 합시다.

공통된 개념들 (Shared concepts)

공간풀러와 시간풀러에서 학습하는 과정은 서로 비슷합니다. 두 경우 모두에서 학습은, 셀들 간의 연결 혹은 시냅스를 형성합니다. 시간풀러는 같은 영역에 속한 셀들 간의 연결을 학습합니다. 공간풀러는 입력 비트와 칼럼들 간의 피드포워드 연결을 학습합니다.

이진수 가중치

HTM 의 시냅스는 오직 0 과 1 의 값만 가집니다; 따라서 그들의 "가중치"는 이진수이며, 다른 여러 신경망들과는 차이가 있습니다. 대부분의 신경망들은 0 과 1 사이의 다양한 값을 이용합니다.

영속성 (Permance)

학습이 이루어지는 동안, 시냅스가 형성되기도 하고 사라지기도 합니다. 이전에 살펴보았듯이, 하나의 연결이 얼마나 지속될것인지를 나타내기 위해 각 시냅스들은 스칼라 값 (0.0 부터 1.0 까지)을 이용합니다. 연결이 강화될수록, 영속성 값이 증가합니다. 어떤 조건에서는, 영속성 값이 감소할 수 있습니다. 만약 영속성 값이 임계치 (예를 들어, 0.2) 이상이 되면, 시냅스가 형성된 것으로 간주합니다. 영속성 값이 임계치 미만인 경우에는, 시냅스는 아무런 기능을 하지 않습니다.

축삭돌기 (Dendrite segments)

시냅스들은 축삭돌기에 연결됩니다. 축삭돌기에는 몸쪽(proximal) 수상돌기와 먼쪽(distal) 수상돌기의 두 가지 타입이 있습니다.

-몸쪽 수상돌기는 피드포워드 입력을 통해 연결을 형성합니다. 이 수상돌기와 연결된 시냅스들의 활성정도를 더하여, 하나의 칼럼을 활성화시킬 것인지 아닌지를 결정합니다.

-먼쪽 수상돌기는 한 영역 내의 셀들과 연결을 형성합니다. 모든 셀들은 몇 개의 먼쪽 수상돌기를 가지고 있습니다. 이 수상돌기들에 연결된 시냅스의 활성정도가 임계치를 넘으면, 연관된 셀이 활성화되어 예측상태로 돌입합니다. 하나의 셀은 여러 개의 먼쪽 수상돌기를 가지기 때문에, 어떤 셀의 예측상태는 여러 개의 임계치 값에 논리함수 'OR' 연산자를 적용하여 결정합니다.

학습 (Learning)

수상돌기에 붙어있는 잠재적인 시냅스들의 영속성 값을 증가시키거나 감소시키는 것이 바로 학습입니다. 시냅스를 더 혹은 덜 영속적으로 만드는 것은 "헤비안(Hebbian)" 학습 규칙과 비슷합니다. 예를 들어, 어떤 시냅스에 연결되어 있는 셀이, 임계치 이상의 입력을 받은 수상돌기를 통해 활성화되었다면, 그 수상돌기에 속한 시냅스들의 영속성 값이 조정됩니다. 어떤 시냅스가 활성화되어 자신이 연결된 셀을 활성화시켰다면, 그 시냅스의 영속성 값은 증가합니다. 비활성화되어 있어서 아무런 역할도 하지 못했다면, 영속성 값이 감소합니다. 시냅스의 영속성 값이 구체적으로 어떻게 변화하는지는 공간풀러와 시간풀러에서 조금 다릅니다. 자세한 내용은 다음에 다루도록 하겠습니다.

이제 공간과 시간풀러의 개념적인 부분에 대해 이야기해봅시다.

공간풀러 개념 (Spatial pooler concepts)

공간풀러의 가장 근본적인 기능은 어떤 영역의 입력을 희소한 패턴으로 변환시키는 것입니다. 이러한 기능은 시퀀스를 학습하고 예측을 만들어내는대에 아주 중요한데, 이 모든 것이 희소하게 분포된 패턴으로부터 비롯하기 때문입니다.

공간풀러가 작동하고 학습하는데에는 몇 가지 서로 중복되는 사항이 있습니다.

1) 모든 칼럼들을 이용하라

HTM 영역은 고정된 수의 칼럼을 가지고 있는데, 이 칼럼들이 입력에서 공통적인 패턴들을 학습합니다. 한가지 중요한 목표는, 몇 개의 칼럼을 이용하는가에 관계없이, 모든 칼럼들이 무언가 유용한 정보를 표현하도록 하는 것입니다. 한 번도 활성화되지 않는 칼럼은 필요하지 않습니다. 이러한 현상을 방지하기 위해서, 하나의 칼럼이 자신의 주변 칼럼들에 비해 얼마나 자주 활성화되는지를 살필 필요가 있습니다. 어떤 칼럼의 상대적인 활성도가 너무 낮다면, 입력 신호에 가중치를 주어 활성화되는 칼럼 그룹에 속하도록 할 수 있습니다. 핵심 사항은, 모든 칼럼들이 자신의 이웃들과 경쟁하면서 입력 패턴을 표현하는데 참여하도록 하는 것입니다. 만약 어떤 칼럼이 자주 활성화되지 않는다면, 그것은 좀 더 적극적으로 변할 필요가 있습니다. 그렇게 되면, 다른 칼럼들은 자신들의 입력을 변화시킬 수 밖에 없으며, 조금씩 다른 입력 패턴을 나타내기 시작합니다.

2) 적절한 밀도를 유지하라

하나의 영역은 입력에 대한 희소분포 표상을 형성할 필요가 있습니다. 가장 많은 입력을 가지는 칼럼이 자신의 이웃을 억제합니다. 칼럼이 입력을 받는 크기에 비례하는 억제반경(inhibition radius)을 설정하여, 이 반지름 이내에서는 가장 많은 입력을 받은 오직 정해진 비율의 칼럼들만이 “승자”가 되도록 합니다. 그 이외의 나머지 칼럼들은 비활성화될 것입니다. (“반지름”이라는 개념이 2 차원의 배열을 암시하지만, 다른 위상의 배열에서도 마찬가지 개념이 적용됩니다.)

3) 사소한 패턴들을 피하라

모든 칼럼들이 입력에서 사소하지 않은 정보를 나타내는 것이 중요합니다. 이를 위해, 어떤 칼럼이 활성화되기 위한 입력의 최소 임계치를 정할 수 있습니다. 예를 들어, 임계치를 50 이라면, 어떤 칼럼의 수상돌기가 활성화되기 위해서는, 그 수상돌기에 연결된 시냅스 중에 적어도 50 개 이상이 활성화되어야함을 의미합니다. 이를 통해 칼럼들이 나타내는 패턴의 복잡성을 어느 정도로 유지할 수 있습니다.

4) 부수적인 연결을 피하라

주의를 기울이지 않으면, 어떤 칼럼은 매우 많은 유효한 시냅스를 가질 수 있습니다. 그렇게 되면, 그 칼럼은 서로 관련이 없는 많은 입력들에 모두 반응하게 됩니다. 자신이 가진 시냅스들의 여러 다른 소집합들이 각자 다른 패턴들에 반응할 것입니다. 이러한 문제를 피하기 위해서는, 어떠한 칼럼이 승자(winner)가 되는데 기여하지 않은 시냅스들은 영속성 값을 감소시킬 필요가 있습니다. 아무런 기여도 하지 않은 시냅스들에 패털티를 줌으로써, 우리는 하나의 칼럼이 제한된 수의 입력패턴 (때때로 오직 하나의) 들을 나타내도록 할 수 있습니다.

5) 스스로 적응하는 수용영역 (receptive field)

실제 뇌는 매우 "유연" 합니다. 대뇌피질의 여러 영역들은 다양한 변화에 대해서 전혀 다른 것들을 배우고 학습할 수 있습니다. 대뇌피질의 일부분이 손상되더라도, 다른 부분들이 손상된 영역이 표현하던 것들을 대신해서 나타낼 수 있습니다. 만약 감각 기관이 손상되거나 변화하면, 대뇌피질의 연관된 부분들이 적응하여 표상을 나타냅니다. 뇌는 스스로 적응하는 기관입니다.

우리는 HTM 영역들도 비슷한 유연성을 가지기를 원합니다. 만약 하나의 영역에 10,000 개의 칼럼이 있다면, 그 영역은 어떻게 입력을 10,000 개의 칼럼으로 가장 잘 표현할지를 배워야만 합니다. 20,000 개의 칼럼을 할당하면, 이 숫자의 칼럼들을 어떻게 이용할지 배워야만 합니다. 입력의 통계적인 특징이 변화하면, 칼럼들도 이 새로운 세상을 어떻게 가장 잘 나타낼지 변화해야 합니다. 즉, HTM 의 설계자들은 하나의 영역에 어떠한 자원을 할당하더라도, 그 사용가능한 칼럼과 입력의 통계적 특성을 이용하여, 그 영역이 어떻게 입력을 가장 잘 표현해내도록 해야 합니다. 일반적으로, 하나의 영역에 더 많은 칼럼이 있을수록, 각각의 칼럼은 더 크고 상세한 패턴을 나타낼 수 있습니다. 이 경우 칼럼들은 더 드물게 활성화되기 때문에, 적절한 희소값을 유지하도록 해야합니다.

이처럼 높은 수준의 목표를 달성하는데 무언가 새로운 규칙이 필요하지는 않습니다. 잘 활성화되지 않는 칼럼들에 가중치를 부여하고, 일정한 희소성을 유지하기 위해 이웃의 칼럼들을 억제하며, 입력의 최소 임계치를 설정하고, 넓은 범위의 잠재적 시냅스를 유지하며, 기여 정도에 따라 시냅스 연결을 더하거나 빼면, 칼럼들이 다양하게 동적으로 변화하며 우리가 목표한 효과를 나타내게 됩니다.

공간풀러 세부사항 (Spatial pooler details)

이제, 공간풀러 함수가 어떻게 작동하는지 그 모든 것을 살펴볼 차례입니다.

1) 정해진 숫자의 비트들로 이루어진 입력을 먼저 떠올려봅시다. 이 입력들은 감각기관으로부터 직접 들어온 데이터일 수도 있고, 계층구조의 낮은 단계로부터 전달된 것일 수도 있습니다.

2) 이 입력을 받는 영역에 정해진 숫자의 칼럼을 할당합니다. 각 칼럼은 관련된 수상돌기를 가집니다. 각 수상돌기에는 잠재적 시냅스들이 연결되어 있는데, 이 시냅스들이 입력 비트의 일부 집합을 표현합니다. 각각의 잠재적 시냅스는 영속성 값을 가집니다. 자신이 가진 영속성 값에 따라서 일부 잠재적 시냅스들은 유효화됩니다.

3) 새로운 입력이 들어올 때마다, 각 칼럼에 있는 유효한 시냅스들 중에 몇 개가 실제 활성화된 입력 비트를 받고 있는지를 확인합니다.

4) 활성화된 시냅스에 "가중치"를 곱합니다. 가중치는 어떤 칼럼이 이웃의 칼럼들과 비교해서 얼마나 자주 활성화하는지에따라 달라집니다.

5) 가중치를 부여한 후에, 가장 높은 활성화값을 가지는 칼럼들이 억제반경 내에 있는 거의 모든 (미리 정해진 비율의) 칼럼들을 불활성화시킵니다. 억제반경은 입력 비트의 팬 아웃 (fan out)에 따라서 계속 변화합니다. 이제 활성된 칼럼들에 대한 희소한 표상이 만들어졌습니다.

6) 매 활성화 칼럼마다, 잠재적 시냅스의 영속성 값을 조정합니다. 활성화된 입력 비트와 연결된 시냅스들의 영속성 값은 증가합니다. 비활성화된 입력들과 연결된 시냅스의 영속성 값은 감소합니다. 이렇게 영속성 값이 변화하면, 어떠한 시냅스들을 유효한 상태에서 유효하지 않은 상태로, 또는 그 반대로 바뀌게 됩니다.

시간풀러 개념 (Temporal pooler concepts)

시간풀러가 시퀀스를 배우고 예측을 만들어낸다는 점을 다시 생각해봅시다. 기본적인 개념은, 하나의 셀이 활성화되면, 바로 이전에 활성상태에 있던 다른 셀들과 연결을 형성하는 것입니다. 이렇게 하면, 셀들은 자신의 주변 연결을 둘러봄으로써 자신이 언제 활성화될 것인지를 예측할 수 있습니다. 모든 셀들이 이와 같이 행동하면, 그들은 집단적으로 시퀀스를 저장하고 불러내며, 다음에 무슨 일이 일어날지를 예측할 수 있습니다. 패턴들의 시퀀스를 따로 저장하는 중심 저장소같은 것은 없습니다. 정보는 널리 퍼져 저장되기 때문에, 시스템은 노이즈와 오류에 더 견고하게 대응할 수 있습니다. 각각의 셀들이 잘못되더라도, 눈에 띄는 오류는 거의 없습니다.

시간풀러가 활용하는 희소분포표상에 대한 몇 가지 중요한 점들을 짚고 넘어갈 필요가 있습니다.

총 10,000 개의 셀들 중에 200 개의 활성화된 셀만을 이용해서 (어느 때에도 2%의 셀들만 활성화 상태에 있는) 표상을 만들어내는 가상의 영역을 생각해봅시다. 이 200 개의 활성화된 셀을 가지고 어떻게 특정 패턴을 인식하고 기억할 수 있을까요? 가장 간단한 방법은 우리가 관심있는 이 200 개의 활성화된 셀들에 대한 리스트를 만든느 것입니다. 만약 똑같은 200 개의 셀이 활성화되면, 우리는 그 패턴을 알아차릴 수 있습니다. 하지만 우리가 200 개 중에 오직 20 개만 이용해서 리스트를 만들고, 나머지 180 개는 무시해버린다면 어떻게 될까요? 어떠한 일이 일어날까요? 단지 20 개의 셀만 기억하는 것은 많은 오류를 만들어낸다고 생각할 수도 있습니다. 왜냐하면 이 20 개의 셀들은 200 개로 표현되는 다양한 서로 다른 패턴들에 의해 활성화될 수도 있기 때문입니다. 하지만 실제로는 그렇지 않습니다. 패턴들이 매우 크고 희소하기 때문에 (이 예에서는, 10,000 개 중에 200 개의 셀), 20 개의 활성화된 셀을 기억하는 것만으로도 200 개를 모두 기억하는 것과 비슷한 효과를 얻을 수 있습니다. 실제 시스템에서 오류가 일어날 확률은 매우 적으며, 이 방법을 통해 우리는 메모리 요구량을 크게 줄일 수 있습니다.

HTM 영역의 셀들은 이러한 특성의 이점을 이용합니다. 각 셀들의 수상돌기는 영역 내의 다른 셀들과 연결을 가지고 있습니다. 수상돌기는 특정 시점에서 네트워크의 상태를 인식함으로써 이러한 연결관계를 인식합니다. 근처에 수천, 수만개의 활성화된 셀이 있을 수 있지만, 수상돌기는 그중 오직 15 혹은 20 개의 셀들과만 연결됩니다. 만약 수상돌기가 15 개 이상의 셀들이 활성화되는 것을 본다면, 무언가 더 큰 패턴들이 일어나고 있다고 생각할 수 있습니다. “부차추출(sub-sampling)”이라고 불리는 이 기술이 HTM 알고리즘 전반에 걸쳐 사용됩니다.

모든 셀들은 수많은 분포 패턴과 수많은 시퀀스에 참여합니다. 특정 셀은 수 십, 수 백개의 시간적 변화의 일부분일 수 있습니다. 따라서 모든 셀들은 한 개보다 많은 여러 개의 수상 돌기를 가집니다. 이상적으로, 하나의 셀은 자신이 인식하고자하는 각 패턴들마다 하나의 수상돌기를 가질 수 있습니다. 하지만 실제로는, 전혀 다른 몇 개의 패턴들에 대한 연결을 배우고 제대로 인식할 수도 있습니다. 예를 들어, 하나의 수상돌기가 4 개의 서로 다른 패턴들에 대해 20 개의 연결관계를 학습하여, 총 80 개의 연결을 가지고 있다고 가정해 봅시다. 그리고 이 연결들 중 15 개 이상이 활성화되면, 그 수상돌기가 활성화 되도록 임계값을 설정해보죠. 이러한 방식은 오류의 가능성을 가지고 있습니다. 우연히, 서로 다른 패턴들의 일부분이 섞여서, 15 개의 활성화된 연결이 생길 수도 있습니다. 하지만, 이러한 종류의 오류는 매우 드문데, 그것은 패턴들의 표상이 희소성을 가지고 있기 때문입니다.

자 이제 10~20 개의 수상돌기와 수 천개의 시냅스를 가진 하나의 셀이, 어떻게 수 백개의 독립된 셀 활성 상태를 인식할 수 있는지 알아보기로 하죠.

시간풀러 세부사항 (Temporal pooler details)

시간풀러가 어떻게 작동하는지에 대한 여러 단계가 아래에 기술되어 있습니다. 공간풀러를 통해 피드포워드 입력이 몇 개의 활성화된 칼럼들로 표현된 그 이후부터 시작해봅시다.

1) 매 활성화된 칼럼들마다 예측상태에 있는 셀이 있는지를 확인하고, 만약 있다면, 그것을 활성화 시킵니다. 어느 셀도 활성화 상태에 있지 않다면, 그 칼럼 안의 모든 셀을 활성화 합니다. 여기서 활성화된 셀의 집합이, 바로 이전에 들어온 입력을 고려하여 표현된 입력표상입니다.

2) 한 영역 안의 모든 셀의 모든 수상돌기마다, 얼마나 많은 유효한 시냅스들이 실제 활성화된 셀과 연결되어 있는지를 확인합니다. 그 값이 임계치를 넘으면, 그 수상돌기는 활성화된 것으로 표시합니다. 활성화된 수상돌기를 가진 셀은, 이미 피드포워드 입력에 의해 활성화되지 않은 이상, 예측상태로 돌입하게 됩니다. 활성화된 수상돌기가 없거나 피드포워드 입력에 의해 활성화되지 않은 셀들은 비활성화된 상태로 남아있습니다. 이 예측상태에 있는 셀들의 집합이 바로 그 영역이 만들어낸 예측을 의미합니다.

3) 수상돌기가 활성화되면, 그 수상돌기와 관련되어있는 모든 시냅스들의 영속성 값을 조정합니다. 모든 활성화된 수상돌기의 잠재적 시냅스마다, 활성화된 셀과 연결된 시냅스들의 영속성 값을 증가시키고, 비활성화된 셀과 연결된 시냅스들의 영속성 값은 감소시킵니다. 이러한 시냅스 영속성 값의 변화를 우선 임시적인 것으로 표기하고 저장합니다.

이러한 변화는 여러 시냅스들 중에서, 이미 자신이 속한 수상돌기를 활성화시킬 수 있을만큼 충분히 훈련된 시냅스를 변화시켜 예측을 만들어 냅니다. 하지만, 가능하다면 더 먼 미래에 대한 예측까지 하는 것이 더 좋습니다. 따라서, 같은 셀에 속한 두 번째 수상돌기를 골라 훈련시킵니다. 이 두 번째 수상돌기는 이전 시간 단계의 시스템 상태와 가장 잘 맞는 것으로 선택합니다. 이전 시간 단계의 시스템 상태를 이용해서, 이 수상돌기에 속한 시냅스들 중에 실제 활성화된 셀과 연결된 것들은 영속성 값을 증가시키고, 비활성화된 셀과 연결된 시냅스의 영속성 값은 감소시킵니다. 이러한 변화 역시, 우선 임시적인 것으로 표기합니다.

4) 어떤 셀이 피드포워드 입력에 의해 비활성화 상태에서 활성화 상태로 전환되면, 이 셀과 연결된 각각의 잠재적인 시냅스들을 모두 추적해서 임시 표기를 제거합니다. 그리고 이 셀들이 피드포워드 입력에 의해 셀이 활성화될 것이라는 사실을 올바르게 예측한 경우에만 시냅스들의 영속성 값을 업데이트 합니다.

5) 만약 어떤 셀이 활성화 상태에서 비활성화 상태로 변화하면, 임시로 표기한 모든 영속성 값 변화를 취소합니다. 피드포워드 입력에 의해 셀이 활성화될 것이라고 잘못 예측한 시냅스들의 영속성 값을 강화시킬 필요는 없습니다.

오직 피드포워드 입력에 의해서 활성화된 셀들만 자신의 활성상태를 영역 *내에서* 퍼뜨릴 수 있다는 점이 중요합니다. 그렇지 않으면, 예측이 또 다른 예측을 만들어내게 됩니다. 하지만 모든 활성화된 셀들 (피드포워드 입력과 예측에 의해)은 어떤 영역의 출력을 만들어내고, 계층구조를 따라 다음 영역으로 퍼져 나갑니다.

일차원 vs. 다차원 시퀀스와 예측

우리가 공간과 시간풀러에 대한 토의를 마치기 전에 토의해야할 중요한 사항이 하나 더 있습니다. 모든 독자들이 관심을 가지는 내용이 아닐 수도 있으며, 3, 4 장을 이해하는데에 반드시 필요한 것은 아닙니다.

하나의 칼럼이 더 많거나 적은 셀을 가지게 된다면 어떻게 될까요? 구체적으로, 만약 하나의 칼럼마다 한 개의 셀만 있다면 무슨 일이 일어날까요?

이전에 예로 들었듯이, 하나의 칼럼마다 4 개의 셀을 가진 100 개의 활성화된 칼럼들을 이용하여 입력을 나타내면, 4^100 개의 서로 다른 패턴을 표현할 수 있습니다. 그러므로, 똑같은 입력이 다양한 맥락에서 나타나더라도 큰 혼란이 생기지 않습니다. 예를 들어, 어떤 영역의 입력 패턴이 단어인 경우에, 같은 단어를 반복해서 사용하는 많은 문장들을 큰 혼동없이 기억할 수 있습니다. “개(dog)”와 같은 단어는 서로 다른 맥락에서도 유일한 표상을 유지할 것입니다. 이러한 기능을 통해 HTM 영역은 소위 “다차원(variable order)”에 걸친 예측을 만들어낼 수 있습니다. 다양한 차원에서 일어나는 예측은 현재 일어나는 일 뿐 아니라, 과거의 맥락의 다양한 부분을 고려하여 예측을 만들어 냅니다. 하나의 HTM 영역은 다차원에 걸친 기억장치인 셈입니다.

하나의 칼럼에 속한 셀의 숫자를 5 개로 늘리면, 어떤 특정 입력에 대해 표현할 수 있는 수는 5^100 개가 되는데, 이는 이전의 4^100 에 비해 훨씬 더 큰 숫자입니다. 하지만 이 두 숫자 모두 엄청나게 크기 때문에 많은 실제적 문제에서 용량을 늘리는 것은 크게 유용하지는 않습니다.

하지만, 셀의 숫자를 훨씬 더 적게 하는 것은 커다란 차이를 만들어 냅니다.

만약 하나의 칼럼마다 한 개의 셀만 있다면, 표상들에 어떤 맥락을 집어넣는 기능이 사라집니다. 영역으로 들어오는 입력들은 과거의 활동과는 관계없이, 언제나 같은 예측을 만들어낼

것입니다. 칼럼 당 하나의 셀을 가지는 HTM 영역은 "일차원(first order)" 메모리에 해당합니다; 예측은 오직 현재의 입력을 통해서만 만들어집니다.

이러한 1 차원의 예측이 실제 뇌에서도 이상적으로 이용되는 경우가 있습니다. 바로 정적공간추론(static spatial inference)입니다. 이전에 언급했듯이, 인간은 눈을 움직일 수 없을 정도로 아주 짧은 순간동안만 시각 자극을 보더라도 그것이 무엇인지 알아차릴 수 있습니다. 하지만 청각을 통해 지금 듣고 있는 것이 무엇인지 인식하기 위해서는, 반드시 일련의 시퀀스와 패턴들을 들어야만 합니다. 우리가 일상적으로 보는 것은 대부분 이미지들의 흐름이기 때문에, 시각도 마찬가지입니다. 하지만 어떤 특정 상황에서, 시각은 단 한번의 짧은 노출만으로도 무엇을 보았는지 별 어려움없이 인식해낼 수 있습니다.

시간과 공간 인식은 아마도 서로 다른 추론 메커니즘을 이용할 것입니다. 시간 인식은 패턴들의 시퀀스를 인식하고 예측을 만들어내기 위해, 다양한 길이의 맥락을 이용합니다. 반면 공간 인식은 시간적 맥락없이 정적인 공간 패턴만을 이용해서 인식할 수 있습니다. 하나의 칼럼에 여러 개의 셀을 가지는 HTM 영역은 이상적으로 시간에 기초한 시퀀스에 적합하지만, 하나의 칼럼에 한 개의 셀을 가지는 HTM 영역은 공간 패턴을 인식하는데 이상적입니다. Numenta 에서, 우리는 칼럼 당 한 개의 셀만 있는 구조를 시각에 응용하는 여러 실험들을 진행하였습니다. 이 실험들의 자세한 내용은 이 장의 목적을 벗어나지만, 중요한 개념들은 다룰 필요가 있습니다.

우리가 HTM 영역에 어떤 이미지를 주면, 그 영역 안의 칼럼들은 픽셀들의 공통된 공간적 배치를 나타내는 법을 배웁니다. 이러한 패턴은, 서로 다른 기울기의 선과 모서리를 이용하는 대뇌피질의 V1 영역 (생물학에서 가장 많이 연구된 대뇌피질 영역)과 유사합니다. 움직이는 이미지를 학습함으로써, HTM 영역은 이러한 기본적인 모양들이 어떻게 변화하는지를 배웁니다. 예를 들어, 어떤 위치에서는 하나의 수직선에 이어 왼쪽 또는 오른쪽으로 움직이는 수직선이 자주 나타날 수 있습니다. 패턴들 사이에서 자주 나타나는 이행과정이 HTM 영역에 저장됩니다.

만약 어떤 영역에 오른쪽으로 움직이는 수직선 영상을 보여주면 어떻게 될까요? 한 개의 칼럼에 한 개의 셀만 있는 경우, 그 영역은 다음에 나타날 수직선이 왼쪽 또는 오른쪽으로 움직이리라 예상할 것입니다. 그 영역은 이 수직선이 과거에 어디에 있었는지를 알 수 없으며, 따라서 그것이 왼쪽 혹은 오른쪽으로 움직일지 알 수 없습니다. 이러한 칼럼 당 한개의 셀 구조가 대뇌피질의 "복잡세포(complex cells)"와 비슷하다는 점이 떠오를 것입니다. 이러한 세포의 예측 결과는 다른 위치에 있는 선들에서도, 그것이 왼쪽 혹은 오른쪽으로 움직이는 것과는 관계없이 동일하게 나타납니다. 우리는 이 영역이 어떠한 변형이나 스케일 변화가 있더라도, 서로다른 이미지에 대한 분별력은 유지함을 확인하였습니다. 이러한 특징이 바로 공간불변량(spatial invariance)-이미지의 다른 위치에서 똑같은 패턴을 인식하는 것-에 반드시 필요한 것입니다.

하나의 칼럼에 여러 개의 셀을 가지는 HTM 영역으로 같은 실험을 해보면, 대뇌피질에서 “방향-조율된 복잡세포(directionally-tuned complex cells)”에 해당하는 것을 발견할 수 있습니다. 이 경우에 셀은 왼쪽으로 움직이는 선이나 오른쪽으로 움직이는 선에 예측 출력을 만들어내지만, 둘 모두에 반응하지는 않습니다.

이러한 점을 고려하면, 다음과 같은 가설을 세울 수 있습니다. 대뇌피질은 일차와 다차원에 걸친 예측과 추론을 만들어내야 합니다. 대뇌피질의 각 영역에는 4 개~5 개의 층이 있습니다. 각 층들은 조금씩 다르지만, 모두 공통된 칼럼구조와 각 층 내의 커다란 수평적 연결관계를 가지고 있습니다. 우리는 대뇌피질의 각 층들이 이 장에서 언급한 HTM 의 추론과 학습 규칙과 유사한 형태로 작동한다고 추측합니다. 서로 다른 층의 세포들은 서로 다른 역할을 합니다. 해부학적인 관찰에 의하면, 6 층은 계층구조에서 되먹임을 만들어내며, 5 층은 운동 기능과 관계가 있습니다. 피드포워드 기능을 담당하는 두 개의 주요한 층은 3 층과 4 층입니다. 우리는 이 3 층과 4 층의 차이 중 하나가 바로 4 층 세포들의 독립적인 활동이라고 생각합니다. 예를 들면, 4 층에는 하나의 칼럼마다 하나의 셀이, 3 층에는 하나의 칼럼마다 여러 개의 셀이 있는 것과 유사합니다. 다라서 감각 입력을 받는 곳 근처의 대뇌피질 영역은 일차원과 다차원의 메모리를 모두 가집니다. 일차원 시퀀스 메모리 (대략 4 층 뉴런들에 해당하는)는 공간적 변화에 대해 불변하는 표상을 만들어내는데 유용합니다. 다차원 시퀴스 메모리 (대략 3 층 뉴런들에 해당하는)은 움직이는 이미지에 대해서 추론과 예측을 만들어내는데 적합합니다.

요약하면, 우리는 이 장에서 기술된 알고리즘과 유사하게 대뇌피질의 여러 층에 속한 뉴런들이 기능할 것이라고 가정하였습니다. 실제 대뇌피질의 층들은 여러 중요한 면에서 차이가 있는데, 피드포워드 vs. 되먹임(feedback), 집중 그리고 운동능력들이 그것입니다. 감각 입력과 가까운 영역들은, 일차원 메모리처럼 공간 불변량을 만들어내는 뉴런들의 층이 더 적합합니다.

Numenta 에서, 우리는 일차원(1 개의 셀/ 칼럼) HTM 을 이용하여 이미지 인식 실험을 진행하였습니다. 또한 다차원(여러 개의 셀/칼럼) HTM 은 다양한 계층의 시퀀스를 인식하고 예측하는데에 이용하였습니다. 이후에, 이 두 가지를 하나의 영역으로 융합하여 다른 응용분야에 확장 적용할 생각입니다. 하지만, 많은 흥미로운 문제들을 일차원, 다차원 HTM 을 따로 이용하거나, 계층구조로 연결하여 해결할 수 있습니다.

제 3 장: 공간풀러 구현 및 가코드(pseudocode)

이 장은 공간풀러 함수를 실제로 구현하는데 필요한 상세한 코드들을 포함하고 있습니다. 이 코드에 들어오는 입력은 센서나 계층구조의 이전 단계에서 전달된 이진(binary)데이터입니다. 이 코드는 activeColumns(t)를 계산하는데, 이 함수는 시간 t 에 들어온 입력을 통해 활성화되고 최종 승리하는 칼럼들의 목록을 담고 있습니다. 이렇게 얻어진 목록이 시간풀러 (다음 장에서 자세히 다룸)의 입력으로 사용됩니다. 즉, activeColumns(t)는 공간풀러 루틴의 출력에 해당합니다.

가코드는 연이어 일어나는 세 개의 특징적인 단계로 구성되어 있습니다.

단계 1: 매 칼럼마다 현재의 입력과 일치하는 정도를 계산
단계 2: 억제 과정 후에 승리한 칼럼들을 계산
단계 3: 시냅스 영속성 정보와 내부 변수들을 업데이트

공간풀러의 학습은 본질적으로는 실시간으로 일어나지만, 단계 3 을 끄는것만으로 간단히 학습을 하지않도록 할 수 있습니다.

이 장의 나머지 부분들은 위의 각 3 단계에 대한 가코드를 다루고 있습니다. 코드에 사용된 다양한 데이턱 구조와 보조 함수들은 마지막에 따로 정의하였습니다.

초기화(Initialization)

처음 입력을 받기 전에, 그 영역에 속한 모든 칼럼의 잠재적 시냅스 목록을 계산하여 초기화시킵니다. 이 과정은 전체 입력 중에서 무작위로 집합을 선택하여 진행합니다. 각 입력은 시냅스로 표현되며, 무작위의 영속성 값을 할당합니다. 이 무작위 영속성 값은 다음과 같은 두 가지 기준에 의해 정해집니다. 첫번째로, connectedPerm (어떤 시냅스를 "연결" 된 것으로 간주하는 최소 영속성 값) 과 거의 비슷한 값을 선택합니다. 이를 통해, 잠재적인 시냅스들은 몇 번의 훈련만으로도 실제 연결되거나 연결이 끊어질 수 있습니다. 두번째로, 각 칼럼들은 입력 영역 위에 어떤 중심점을 가지며, 영속성 값은 이 중심점을 향해 편향됩니다 (중심에 가까울수록 더 큰 값을 가집니다).

단계 1: 겹치게 하기(Overlap)

주어진 입력 벡터에 대해 첫번째로 할 일은, 각 칼럼들이 그 벡터와 얼마나 겹치는지를 계산하는 것입니다. 각 칼럼들의 겹치는 정도는, 활성화된 입력과 연결된 시냅스들의 수에 가중치를 곱한 값입니다. 이 값이 minOverlap 보다 낮으면, 겹치는 정도를 0 으로 설정합니다.

```
1.  for c in columns
2.
3.              overlap(c) = 0
4.              for s in connectedSynapses(c)
5.                      overlap(c) = overlap(c) + input(t, s.sourceInput)
6.
7.              if overlap(c) < minOverlap then
8.                      overlap(c) = 0
9.              else
10.             overlap(c) = overlap(c) * boost(c)
```

단계 2: 억제(Inhibition)

두 번째 단계는 억제 과정 이후에 어떻나 칼럼들이 승리할지를 계산하는 것입니다. desiredLocalActivity 라는 변수를 통해 최종 승리할 칼럼들의 갯수를 조절합니다. 예를 들어, 만약 desiredLocalActivity 가 10 이라면, 억제반경 안에서 10 번째로 높은 값을 가지는 칼럼보다 더 큰 값을 가지는 칼럼만이 승리합니다.

```
11. for c in columns
12.
13.     minLocalActivity = kthScore(neighbors(c), desiredLocalActivity)
14.
15.     if overlap(c) > 0 and overlap(c) ≥ minLocalActivity then
16.             activeColumns(t).append(c)
17.
```

단계 3: 학습(Learning)

세 번째 단계는 학습을 수행합니다: 필요하다면, 모든 시냅스들의 영속성 값을 업데이트하고, 가중치와 억제반경을 조절합니다.

핵심 학습 규칙은 20-26 행에 구현되어 있습니다. 만약 승리한 칼럼들의 시냅스가 활성화되었다면 영속성 값을 증가시키며, 반대의 경우에는 감소시킵니다. 만약 어떤 칼럼이

자주 승리하지 못한다면 (activeDutyCycle 로 측정된) , 그것의 전반적인 가중치값을 증가시킵니다 (30-32 행). 그 대신에, 만약 어떤 칼럼의 연결된 시냅스가 입력과 잘 겹치지 않는다면 (overlapDutyCycle 로 측정된), 영속성 값을 가중시킵니다 (34-36 행). 참고: 학습 기능을 끄는 경우에, boost(c)함수는 정지됩니다.

단계 3 의 마지막에는, 억제반경을 재계산 합니다 (38 행).

```
18. for c in activeColumns(t)
19.
20.       for s in potentialSynapses(c)
21.             if active(s) then
22.                   s.permanence += permanenceInc
23.                   s.permanence = min(1.0, s.permanence)
24.             else
25.                   s.permanence -= permanenceDec
26.                   s.permanence = max(0.0, s.permanence)
27.
28. for c in columns:
29.
30.       minDutyCycle(c) = 0.01 * maxDutyCycle(neighbors(c))
31.       activeDutyCycle(c) = updateActiveDutyCycle(c)
32.       boost(c) = boostFunction(activeDutyCycle(c), minDutyCycle(c))
33.
34.       overlapDutyCycle(c) = updateOverlapDutyCycle(c)
35.       if overlapDutyCycle(c) < minDutyCycle(c) then
36.             increasePermanences(c, 0.1*connectedPerm)
37.
38. inhibitionRadius = averageReceptiveFieldSize()
39.
```

지원 데이터 구조와 루틴(Supporting data structures and routines)

아래에 제시된 변수와 데이터 구조들이 가코드에서 사용되었습니다.

columns	모든 칼럼들의 목록
input(t,j)	시간 t 에서 이 단계에 들어온 입력. j 번째 입력이 1 인 경우에 input(t,j)는 1
overlap(c)	특정 입력 패턴과 칼럼 c 공간풀러의 겹침 정도
activeColumns(t)	아랫단계로부터의 입력으로 인해 승리한 칼럼들의 목록
desiredLocalActivity	억제 단계 이후에 승리할 칼럼들의 숫자를 조절하는 변수
inhibitionRadius	칼럼들이 입력 영역과 연결된 평균 크기
neighbors(c)	칼럼 c 의 inhibitionRadius 이내에 있는 모든 칼럼들의 목록
minOverlap	억제 단계동안에 어떤 칼럼이 활성화된 것으로 간주하기 위해 필요한 최소 입력의 수
boost(c)	학습 과정동안 계산된 칼럼 c 의 가중치 값. 비활성화된 칼럼들의 overlap 값을 증가시킬 때 이용됨
synapse	시냅스의 데이터 구조를 나타내는 변수 -영속성 값과 입력 소스의 색인을 포함.
connectedPerm	만약 어떤 시냅스의 영속성 값이 이보다 크면, 연결된 것으로 간주.
potentialSynapses(c)	잠재적 시냅스들과 그 영속성 값의 목록
connectedSynapses(c)	potentialSynapses(c)의 소집합 중에서 영속성 값이 connectedPerm 보다 집합. 이것이 칼럼 c 에 현재 연결된 아랫단계의 입력임
pemanenceInc	학습 과정동안 증가될 시냅스의 영속성 값
pemanenceDec	학습 과정동안 감소될 시냅스의 영속성 값
activeDutyCycle(c)	억제 과정 이후에 얼마나 자주 칼럼 c 가 활성화되었는지에 대한 변화평균값 (예, 최근 1,000 번 반복동안의 평균)
OverlapDutyCycle(c)	얼마나 자주 칼럼 c 가 입력과 의미있는 overlap 이 있었는지 (minOverlap 보다 큰) 에 대한 변화평균값 (예, 최근 1000 번 반복동안의 평균)
minDutyCycle(c)	어떤 셀에 필요한 최소 발화율(firing rate)을 나타내는 변수. 만약 셀의 발화율이 이 값보다 낮다면, 가중치를 부여함. 이 값은 자신 주위의 셀이 가지는 최대 발화율의 1%로 설정

아래 지원 루틴들이 위의 코드에서 사용되었음.

KthScore(cols,k)

주어진 칼럼 목록에 대해서, k 번째로 높은 겹침 값을 반환함

updateActiveDutyCycle(c)

억제 이후에 얼마나 자주 칼럼 c 가 활성화되었는지에 대한 변화평균값을 계산

updateOvelapDutyCycle(c)

얼마나 자주 칼럼 c 가 minOverlap 보다 큰 값을 가졌는지에 대한 변화평균값을 계산

averageReceptiveFieldSize()

모든 칼럼들의 연결된 수용영역 크기에 대한 평균 반지름. 연결된 수용영역 반지름은 오직 연결된 시냅스들만을 포함 (영속성 값 >=conntectePerm 인 경우). 이 값은 칼럼들 사이의 같은 단계에서의 억제 정도를 결정하는데 사용됨.

maxDutyCycle(cols)

주어진 칼럼 리스트에서 칼럼의 최대 activeDutyCycle 을 반환함

increasePermanences(c,s)

칼럼 c 에 속한 모든 시냅스들의 영속성 값을 s 단계만큼 증가시킴

boostFunction(c)

어떤 칼럼의 가중치 값을 반환. 가중치 값은 스칼라>=1 임. 만약 activeDutyCycle(c)의 값이 minDutyCycle(c)보다 크다면, 가중치 값은 1. 이 가중치 값은 activeDutyCycle 이 minDutyCycle 값 아래로 떨어지는 순간 선형적으로 증가함

제 4 장 : 시간풀러 구현과 가코드(pseudocode)

이 장에서는 시간풀러함수 구현을 위한 자세한 가코드를 다룰 예정입니다. 이 코드의 입력은 공간풀러에 의해 계산된 activeColumns(t) 입니다. 이 코드는 현재 시간단계 t 의 모든 셀들의 활성 정도와 예측상태를 계산합니다. 각 셀들의 활성 정도와 예측상태에 논리함수 OR 를 적용하면 시간풀러의 출력이 만들어지고, 다음 단계로 전달됩니다.

가코드는 연이어 일어나는 세 개의 특징적인 단계로 이루어져 있습니다.

단계 1: 각 셀마다 활성상태인 activeState(t)를 계산합니다.
단계 2: 각 셀마다 예측상태인 predictveState(t)를 계산합니다.
단계 3: 시냅스들을 업데이트 합니다.

단계 3 은 학습 과정에서만 사용됩니다. 하지만 공간풀러와 다르게, 단계 1 과 단계 2 에도 학습에 특화된 기능들이 있습니다. 시간풀러는 공간풀러에 비해 훨씬 더 복잡하기 때문에, 먼저 시간풀러에서 추론 기능만을 따로 다룬 후에, 추론과 학습이 합쳐진 버전을 다루도록 하겠습니다. 실제 구현을 위한 세부사항과 용어, 그리고 지원 루틴들은 이 장의 마지막에 정리해놓았습니다.

시간풀러 가코드 : 추론만 구현

단계 1

첫 번째 단계는 모든 셀들의 활성 상태를 계산하는 것입니다. 공간풀러를 통해 승리한 모든 칼럼마다 어떠한 셀이 활성화되어야할지를 정해야합니다. 만약 어떤 셀이 아랫단계로부터의 입력을 예상했다면 (즉, 이전 시간 단계에서 중요 수상돌기에 의해 predictiveState 가 1 이 된 경우), 그 세포가 활성화 됩니다 (4-9 행). 만약 아랫단계로부터의 입력이 예측하지 못한 것이라면 (어떠한 셀들도 predictiveState 가 1 이 아니었다면), 그 칼럼에 속한 모든 세포가 활성화 됩니다 (11-13 행).

```
1.  for c in activeColumns(t)
2.
3.      buPredicted = false
4.      for i = 0 to cellsPerColumn - 1
5.          if predictiveState(c, i, t-1) == true then
6.              s = getActiveSegment(c, i, t-1, activeState)
7.              if s.sequenceSegment == true then
8.                  buPredicted = true
9.                  activeState(c, i, t) = 1
10.
11.     if buPredicted == false then
12.         for i = 0 to cellsPerColumn - 1
13.             activeState(c, i, t) = 1
```

단계 2

두 번째 단계는 각 셀들의 예측상태를 계산하는 것입니다. 만약 자신이 가진 수상 돌기중 하나라도 활성화되어 있다면, 그 셀은 predictiveState 로 들어갑니다. 이는 그 셀과 같은 단계에 속한 많은 연결들이, 현재 아랫단계로부터 입력을 받고 있음을 의미합니다.

```
14. for c, i in cells
15.     for s in segments(c, i)
16.         if segmentActive(c, i, s, t) then
17.             predictiveState(c, i, t) = 1
```

시간풀러 가코드: 추론과 학습을 함께 구현

단계 1

첫 번째 단계는 승리한 칼럼에 속한 모든 셀들의 activeState 를 계산하는 것입니다. 그 칼럼들 중에서, 시간풀러 코드는 하나의 칼럼마다 하나의 셀을 선택하여 학습 셀(learnState)로 지정합니다. 방법은 다음과 같습니다: 만약 아랫단계로부터의 입력을 예측한 세포가 있다면 (중요 수상돌기에 의해 predictiveState 가 1 이라면), 그 세포들이 활성화 됩니다 (23-27 행). 만약 그 수상돌기의 활성이 learnState 가 1 인 세포에서 비롯된 것이라면, 이 세포가 학습 세포로 선택됩니다 (28-30 행). 만약 아랫단계에서의 입력을 예측하지 못했다면, 그 칼럼에 속한 모든 셀들이 활성화 됩니다(32-34 행). 게다가, 가장 짝이 잘 맞는 세포가 학습세포로 선택되고 (36-41 행) 세로운 수상돌기가 그 세포에 더해집니다.

```
18. for c in activeColumns(t)
19.
20.     buPredicted = false
21.     lcChosen = false
22.     for i = 0 to cellsPerColumn - 1
23.         if predictiveState(c, i, t-1) == true then
24.             s = getActiveSegment(c, i, t-1, activeState)
25.             if s.sequenceSegment == true then
26.                 buPredicted = true
27.                 activeState(c, i, t) = 1
28.                 if segmentActive(s, t-1, learnState) then
29.                     lcChosen = true
30.                     learnState(c, i, t) = 1
31.
32.     if buPredicted == false then
33.         for i = 0 to cellsPerColumn - 1
34.             activeState(c, i, t) = 1
35.
36.     if lcChosen == false then
37.         l,s = getBestMatchingCell(c, t-1)
38.         learnState(c, i, t) = 1
39.         sUpdate = getSegmentActiveSynapses (c, i, s, t-1, true)
40.         sUpdate.sequenceSegment = true
41.         segmentUpdateList.add(sUpdate)
```

단계 2

두 번째 단계는 모든 셀들의 예측상태를 계산하는 것입니다. 하나의 셀은 자신이 가진 수상돌기 중 하나라도 활성화되면 예측상태로 들어갑니다. 이는 그 셀과 같은 단계에 속한 많은 연결들이, 현재 아랫단계로부터 입력을 받고 있음을 의미합니다. 이 경우에, 셀은 다음과 같은 변화를 겪습니다: a) 현재 활성화된 수상돌기와의 연결을 강화하고 (47-48 행), b) 이러한 활성화를 예측해야만 했던 수상돌기와의 연결을 강화합니다 (즉, 이전 시간 단계에서 활성화되기에는 조금 약한 연결을 가지고 있었던 수상돌기).

```
42. for c, i in cells
43.     for s in segments(c, i)
44.         if segmentActive(s, t, activeState) then
45.             predictiveState(c, i, t) = 1
46.
47.             activeUpdate = getSegmentActiveSynapses (c, i, s, t, false)
48.             segmentUpdateList.add(activeUpdate)
49.
50.             predSegment = getBestMatchingSegment(c, i, t-1)
51.             predUpdate = getSegmentActiveSynapses(
52.                                 c, i, predSegment, t-1, true)
53.             segmentUpdateList.add(predUpdate)
```

단계 3

세 번째이자 마지막 단계는 실제 학습을 수행하는 것입니다. 이 단계에서 수상돌기는 대기하고 있던 업데이트를 실제로 실행하게 되는데, 이는 아랫단계에서 입력이 들어오고, 그 셀이 학습 세포로 선정된 경우에만 일어납니다 (56-57 행). 만약 세포가 어떠한 이유해서이든 예측하는 것을 멈춘다면, 그 수상돌기의 연결은 약화시킵니다 (58-60 행).

```
54. for c, i in cells
55.     if learnState(s, i, t) == 1 then
56.         adaptSegments (segmentUpdateList(c, i), true)
57.         segmentUpdateList(c, i).delete()
58.     else if predictiveState(c, i, t) == 0 and predictiveState(c, i, t-1)==1 then
59.         adaptSegments (segmentUpdateList(c,i), false)
60.         segmentUpdateList(c, i).delete()
61.
```

구현을 위한 세부사항과 용어

여기서는 시간풀러 구현을 위한 세부사항과 용어들을 다루기로 하겠습니다. 각 셀은 두 개의 숫자를 이용하여 표기됩니다: 칼럼 번호 c 와 셀 번호 i. 셀들은 수상 돌기의 목록을 가지고 있는데, 각 수상돌기들은 시냅스 목록과 각 시냅스의 영속성 값을 가지고 있습니다. 셀의 시냅스에 변화가 생기면, 그 셀이 아랫단계의 입력에 의해 활성화되기 까지는 임시적인 것으로 표기합니다. 이러한 임시 변화는 segmentUpdateList 에 저장해놓습니다. 각 수상돌기에는 또한 논리표식과, 중요수상돌기(sequenceSegment)라는 변수가 있는데, 이들은 그 수상돌기가 다음 시간 단계에서 아랫단계의 입력을 예측하였는지를 의미합니다.

시간풀러에서 잠재적인 시냅스를 구현하는 것은 공간풀러에서의 구현과 조금 다릅니다. 공간풀러에서는, 잠재시냅스들의 모든 목록이 외부에 따로 기록되어 있었습니다. 마찬가지로 시간풀러에서도 각 수상돌기들이 자신만의 (아마도 매우 큰) 잠재적인 시냅스 목록을 가지고 있을 수 있습니다. 실제적으로, 각 수상돌기마다 이러한 긴 목록을 가지고 있는 것은 엄청난 메모리와 컴퓨터 능력을 필요로 합니다. 따라서 시간풀러에서, 우리는 학습이 일어나는동안 각 수상돌기에 무작위로 활성화된 시냅스를 더합니다 (변수 newSynapseCount 로 조절되는). 이러한 최적화는 모든 잠재적인 시냅스의 목록을 유지하는 것과 유사하지만, 각 수상돌기가 가지는 목록의 수는 훨씬 적으면서 새로운 시간 패턴을 배우는 능력은 그대로 유지할 수 있습니다.

또한 가코드는 서로 다른 시간 단계에서 세포의 상태를 추적하는 조그만 상태 함수를 사용합니다. 각 셀마다 세 개의 서로다른 상태를 유지합니다. activeState 와 predictiveState 배열은 각 시간 단계마다 각 셀들의 활성 상태와 예측상태를 추적합니다. learnState 배열은 학습이 이루어지는 동안 어떠한 셀 출력을 사용할지를 결정합니다. 새로 들어온 입력이 예측하지 못한 것이라면, 특정 칼럼에 속한 모든 세포들이 동시에 활성화됩니다. 이 셀들 중 오직 하나의 셀만이 (입력과 가장 잘 맞아들어가는) learnState 가 1 로 변합니다. 우리는 learnState 가 1 인 셀에만 시냅스를 더합니다

다음과 같은 데이터 구조가 시간코드의 가코드에 이용되었습니다:

cell(c,i)	i 와 c 로 표현되는 모든 세포들의 목록
cellsPerColumn	각 칼럼에 속한 셀들의 숫자
activeColumn(t)	아랫단계의 입력으로 인해 승리한 칼럼들의 목록 (이것이 공간풀러의 출력임)
activeState(c,i,t)	각 셀마다 하나의 값을 가지는 불린(boolean)벡터. 현재의 아랫단계로부터의 입력과 과거의 시간 맥락을 고려하여 칼럼 c 의 셀 i 의 시간 t 에서의 활성 상태를 나타냄. 만약 1 이며, 그 셀은 현재 아랫단계로부터 입력을 받을 뿐 아니라 적절한 시간 맥락을 가지고 있음을 의미.
predictiveState(c,i,t)	각 셀마다 하나의 값을 가지는 불린(boolean)벡터. 현재의 아랫단계로부터의 입력과 과거의 시간 맥락을 고려하여 칼럼 c 의 셀 i 의 시간 t 에서의 예측상태를 나타냄. 만약 1 이며, 그 셀은 현재의 시간 맥락을 바탕으로 아랫단계에서 입력이 들어올 것이라고 예측하고 있음을 의미.
learnState(c,i,t)	칼럼 c 에 속한 셀 i 가 학습 상태로 돌입할 것인지를 나타내는불린(boolean)벡터.
activation Threshold	수상돌기의 활성 임계치를 나타냄. 수상돌기에 연결되어 활성화된 시냅스의 숫자가 이 값보다 크면, 그 수상돌기가 활성화 되었다고 이야기함
learningRadius	시간풀러 셀들이 주변으로부터 연결을 받을 수 있는 면적
initialPerm	시냅스의 초기 영속성 값
connectedPerm	시냅스의 영속성 값이 이보다 크면, 연결된 것으로 간주
minThreshold	학습 과정에서의 최소한의 수상돌기 활성도
newSynapseCount	학습이 일어나는 동안 수상돌기에 더해질 수 있는 최대의 시냅스 숫자
permanenceInc	활성-기반 학습이 일어날 때 시냅스에서 증가하는 영속성 값
permanenceDec	활성-기반 학습이 일어날 때 시냅스에서 감소하는 영속성 값
segmentUpdate	주어진 수상돌기를 업데이트하는데 필요한 세 종류의 정보를 담고 있음: a) 수상돌기 인덱스(새로운 수상돌기이면 -1), b) 활성화된 시냅스들의 목록, c) 이 수상돌기를 중요수상돌기로 표기할 것인지를 나타내는 기호 (초기값은 false).
segmentUpdateList	segmentUpdate 데이터 구조의 목록. segmentUpdateList(c,i)는 칼럼 c 의 셀 i 의 변화 목록

아래 지원 루틴들이 위의 코드에서 사용되었음.

SegmentActive(s, t, state)

이 루틴은 특정 시간에 활성화된 수상돌기 s 에 속한 연결된 시냅스들의 숫자가 activationThreshold 보다 큰 경우에 true 를 반환합니다. 변수의 상태는 activeState 혹은 learnState 입니다.

getActiveSegment(c,i,t,state)

주어진 칼럼 c 의 세포 i 에 대해서, segmentActive(s,t,state)의 상태가 true 인 수상돌기의 인덱스를 반환합니다. 만약 다수의 수상돌기가 활성화되어 있으면, 중요수상돌기가 우선권을 가집니다. 중요수상돌기가 없다면, 가장 크게 활성화된 수상돌기가 우선권을 가집니다.

getBestMatchingSegment(c,i,t)

특정 시간 t 에 주어진 칼럼 c 의 세포 i 에 대해서, 가장 많은 숫자의 활성화된 시냅스를 가진 수상돌기를 찾습니다. 이 루틴은 다소 공격적으로 '가장 잘 어울리는수상돌기'를 찾습니다. 시냅스들의 영속성 값이 connectedPerm 인 경우도 포함합니다. 활성화된 시냅스들의 값이 activationThreshold 보다 낮은 경우도 포함합니다. 이 루틴은 수상돌기의 인덱스를 반환합니다. 만약 이 조건에 맞는 수상돌기가 없으면, 인덱스 -1 을 반환합니다.

getBestMatchingCell(c)

주어진 칼럼에 대해서, '가장 잘 어울리는 수상돌기' (위에서 정의된) 를 가지는 셀을 반환합니다. 만약 그러한 수상돌기가 없다면, 가장 적은 수의 수상돌기를 가지는 셀을 반환합니다.

getSegmentActiveSynapses (c,i,t, newSynapses=**false**)

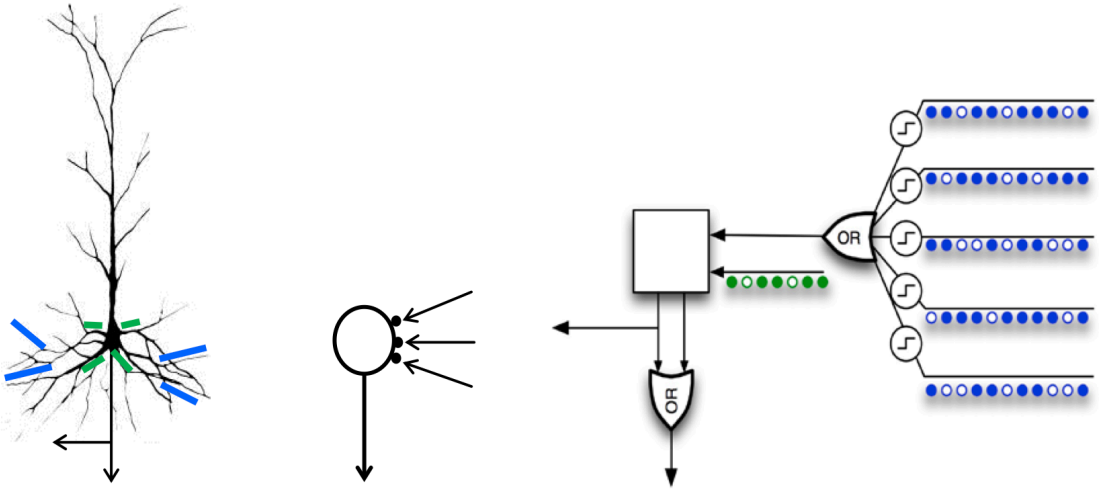
수상돌기 s 에서 변화시킬 목록들을 담은 데이터 구조인 segmentUpdate 를 반환합니다. activeSynapses 를 시간 t 에서 activeState 출력이 1 인 세포들로부터 활성화된 시냅스들의 목록이라고 해봅시다 . (s=-1 인 경우에는 이 목록은 비어있는데, 그러한 수상돌기가 없기 때문입니다.) newSynapses 는 선택적인 선언인데, 기본으로 false 값을 가집니다. 만약 newSynapses 가 true 라면, newSynapseCount-count(activeSynapses) 시냅스들이 activeSynapses 에 더해집니다. 이러한 시냅스들은 시간 t 에서 learnState 출력이 1 인 셀들의 집합으로부터 무작위로 선택됩니다.

adaptSegments(segment List, positiveReinforcement)

이 함수는 segmentUpdate 리스트를 반복하고 각 수상돌기들을 강화시킵니다. 모든 segmentUpdate 구성원마다, 다음과 같은 변화를 수행합니다.

만약 positiveReinforcement 가 true 라면, 활성 목록에 있는 시냅스의 영속성 값을 permanenceInc 값 만큼 증가시킵니다. 모든 다른 시냅스들은 permanenceDec 만큼 영속성 값이 감소합니다. 만약 positiveReinforcement 가 false 라면, 활성 목록에 있는 시냅스들의 영속성 값은 permanenceDec 만큼 감소합니다. 이 과정 이후에, segmentUpdate 에 아직 남아있는 시냅스들에는 initialPerm 값만큼 영속성 값을 더합니다.

부록 A: 생물학적 뉴런과 HTM 셀의 비교



위의 이미지에서 가장 왼쪽은 생물학적인 뉴런이, 가운데에는 간단한 형태의 인공 뉴런이, 그리고 가장 오른쪽에는 HTM 뉴런 혹은 "세포"가 나타나 있습니다. 이 부록에서는 HTM 셀에 대한 이해를 높이고, 실제 뉴런과 간단한 형태의 인공 뉴런과 비교함으로써, HTM 셀이 어떻게 작동하는지를 설명하고자 합니다.

실제 생물학적인 뉴런은 놀라울 정도로 복잡하고 다양합니다. 우리는 이 중에서 가장 일반적인 원리와 우리의 모델에 적용된 점들을 중점적으로 이야기할 것입니다. 비록 우리가 실제 뉴런의 많은 세부적인 면을 고려하지 않았지만, HTM 피질 학습 알고리즘에서 사용된 셀들은 대부분의 신경망에서 이용된 인공 뉴런에 비해 훨씬 더 실제에 가깝습니다. HTM 셀에 포함된 모든 구성 요소들이 HTM 영역을 작동하는데 필수적입니다.

생물학적 뉴런

뉴런은 뇌에서 정보를 전달하는 세포입니다. 위의 그림에서 맨 왼쪽에 있는 것이 전형적인 흥분성 뉴런(excitatory neuron)입니다. 이 뉴런의 특징은 시각적으로 두드러지게 나타나는 수상돌기 가지들입니다. 뉴런으로 들어오는 모든 흥분성 입력은 이 수상돌기 위에 배열되어있는 시냅스를 통해 들어옵니다. 최근에 뉴런에 대한 우리의 이해는 놀라울 정도로 발전하였습니다. 가장 큰 변화는 뉴런의 수상돌기가 단순히 정보를 세포체(cell body)로 전달하는데에 그치지 않는다는 점을 발견한 것입니다. 이제 우리는 수상돌기가 그들 스스로

복잡한 비선형적인 처리 기관임을 알고 있습니다. HTM 피질 학습 알고리즘은 이러한 비선형적인 성질을 이용하고 있습니다.

생물학적인 뉴런은 몇 가지 구조물로 이루어져 있습니다.

세포체 (Cell body)

세포체는 뉴런의 중심에 있으며, 상대적으로 적은 부피를 차지합니다. 세포의 출력을 담당하는 축삭돌기(axon)은 세포체로부터 비롯됩니다. 세포로 들어오는 입력은 수상돌기 위에 배열된 시냅스를 통해 전달되는데, 이 수상돌기가 세포체로 연결되어 있습니다.

몸쪽 수상돌기 (Proximal Dendrites)

세포체 가까이에 있는 수상돌기 가지들을 몸쪽 수상돌기라고 부릅니다. 앞의 모식도에서 몸쪽 수상돌기들이 초록색 선으로 표시되어 있습니다.

몸쪽 수상돌기에서 활성화된 여러개의 시냅스들은 선형적으로 그 값을 더하여 세포체로 전달합니다. 다섯 개의 활성화된 시냅스들은 한 개의 시냅스에 비해 약 5 배정도 세포체의 탈분극(depolarization)을 일으킵니다. 반면, 한 개의 시냅스가 아주 빠르게 반복되는 활동 전위에 의해 활성화되었다면, 두 번째, 세 번째 그리고 연이은 활동 전위들은 첫 번째 것보다는 세포체에 훨씬 적은 영향을 미칩니다.

그러므로, 수상돌기로 들어오는 입력들이 세포체에서 선형적으로 더해진다고 이야기할 수 있으며, 하나의 시냅스에 도달한 빠른 스파이크(spike)는 하나의 스파이크보다 오직 조금만 더 큰 영향을 끼칩니다.

대뇌피질의 어떤 영역으로 연결된 아랫단계의 입력은 보통 몸쪽 수상돌기를 통해 연결됩니다. 4 층의 뉴런이 이러한 역할을 담당하는 것으로 알려져 있는데, 이 층이 각 영역에서 첫 번째로 입력을 받아들이는 곳입니다.

먼쪽 수상돌기 (Distal Dendrites)

세포체로부터 좀 더 멀리에서 가지를 내는 돌기를 먼쪽 수상돌기라고 부릅니다. 앞의 그림에서 먼쪽 수상돌기들이 파란 선으로 표시되었습니다.

먼쪽 수상돌기는 몸쪽 수상돌기보다 더 얇습니다. 이들은 다른 수상돌기의 가지와 연결을 형성하며, 세포체에 직접 연결되어 있지 않습니다. 이러한 차이는 먼쪽 수상돌기의 독특한 전기-화학적 특성을 만들어냅니다. 먼쪽 수상돌기 위의 하나의 시냅스가 활성화되더라도, 세포체에는 거의 영향을 미치지 않습니다. 시냅스 주위에서 국소적으로 일어나는 탈분극은 세포체로 다가갈수럭 더 약해집니다. 오랫동안 이러한 현상은 미스테리였습니다. 시냅스의

대부분을 차지하는 먼쪽 수상돌기는 실제로는 그다지 중요한 역할을 하는 것처럼 보이지 않았습니다.

최근에서야 이 먼쪽 수상돌기가 반독립적(semi-independent)인 정보처리 단위라는 것이 알려졌습니다. 짧은 거리 안에서 동시에 충분한 수의 시냅스들이 활성화되면, 세포체까지 도달하여 커다란 효과를 낼 수 있는 수상돌기 스파이크가 만들어집니다. 예를 들어, 40μm 이내에서 20 개의 시냅스가 활성화되면, 수상돌기 스파이크가 생성됩니다.

따라서, 먼쪽 수상돌기는 마치 '임계치 동시발생 감지기(threshold coincidence detector)'처럼 작동한다고 이야기할 수 있습니다.

먼쪽 수상돌기위에 있는 시냅스들은, 대부분 같은 영역에서 근처에 있는 세포들로부터 기원합니다.

앞의 이미지에서 위쪽으로 길게 뻗어있는 커다란 수상돌기를 선단수상돌기(apical dendrite)라고 부릅니다. 한 이론에 따르면, 이러한 구조는 뉴런들이 어떤 영역 안의 여러 먼쪽 수상돌기의 위치를 알 수 있도록 하며, 이를 통해 주위를 지나는 축삭돌기와 쉽게 연결을 형성할 수 있습니다. 이러한 관점에서 보면, 선단수상돌기는 세포의 영향력을 확장시키는 역할을 합니다.

시냅스(Synapses)

전형적인 뉴런들은 수 천개의 시냅스를 가지고 있습니다. 이들 중 대부분은 (약 90%) 먼쪽 수상돌기 위에 있으며, 나머지는 몸쪽 수상돌기 위에 있습니다.

오랫동안, 학습은 시냅스의 "가중치"를 강화시키거나 약화시키는 과정을 통해 일어나는 것으로 생각되었습니다. 이러한 효과가 실제로 관찰되었지만, 각각의 시냅스들은 다소 확률적으로 작동합니다. 시냅스가 활성화되었다고 해서, 반드시 신경전달물질을 내보내는 것은 아닙니다. 따라서 실제 뇌에서 사용되는 알고리즘은 어떤 정교한 기준이나 각 시냅스 가중치 값의 정확도에 의존할 수가 없습니다.

게다가, 모든 시냅스들이 빠르게 형성되거나 사라집니다. 이러한 유연성은 학습의 여러 놀라운 능력이나 지식의 빠른 습득 능력을 잘 설명해줍니다. 시냅스는 축삭돌기와 수상돌기가 특정 거리 이내에 있을 때만 형성될 수 있는데, 이러한 사실이 "잠재적" 시냅스의 개념과 관련되어 있습니다. 이러한 가정을 바탕으로, 학습은 잠재적 시냅스로부터 실제 유효한 시냅스를 형성하는 것과 큰 관련이 있다고 할 수 있습니다.

뉴런의 출력(Neuronal Output)

하나의 뉴런의 출력은 스파이크 혹은 "활동전위(action potential)"인데, 축삭돌기를 통해 전달됩니다. 축삭돌기는 세포체를 떠나 거의 대부분 두 개로 나누어 집니다. 하나의 가지는 수평방향으로 뻗어나가 근처의 다른 세포들과 많은 연결을 만듭니다. 다른 가지는 다른 층의 세포나 뇌의 다른 부분들로 투사됩니다. 위의 그림에 제시된 뉴런 그림에서 축삭돌기는 나타나있지 않습니다. 대신 하나의 선과 두 개의 화살표가 축삭돌기를 표현하고 있습니다.

뉴런의 실제 출력은 항상 스파이크지만, 이것을 해석하는데에는 여러 관점이 있습니다. 가장 잘 알려지 관점은 (특히 대뇌피질에서) 스파이크의 발생율(rate)이 중요하다는 것입니다. 따라서 세포의 출력은 일종의 스칼라 값으로 볼 수 있습니다.

어떤 뉴런들 역시 "폭발적(bursting)"인 행태를 보여주는데, 이는 몇 개의 스파이크가 짧고 빠르게 연달아 일어난다는 점에서 일반적인 스파이크 패턴과 차이가 있습니다.

지금까지의 뉴런에 대한 설명은 개략적인 소개였습니다. HTM 셀들의 특징과 일치하는 부분들에 중점을 두었기 때문에, 많은 세부적인 사항들이 빠져있습니다. 또한 지금 설명한 모든 특징들이 모두 확실하게 인정받고있는 것은 아닙니다. 그럼에도 불구하고, 우리의 모델에서 필요한 사항들을 설명에 포함시켰습니다. 뉴런에 대해 지금까지 알려진 내용들을 설명하기 위해서는 수십 권의 책으로도 부족하며, 지금 이 순간에도 활발한 연구가 이루어지고 있습니다.

간단한 형태의 인공 뉴런

이 부록의 처음에 제시된 그림의 가운데에는 많은 고전 인공지능 신경망에서 사용하는 뉴런을 닮은 구조물이 나타나있습니다. 이 인공뉴런은 가중치값을 가지는 몇 개의 시냅스를 가지고 있습니다. 각 시냅스들은 스칼라 활성화값을 받아 거기에 시냅스의 가중치를 곱합니다. 모든 시냅스들의 출력이 비선형적인 방식으로 더해지면, 인공뉴런의 출력을 만들어냅니다. 학습은 시냅스의 가중치값을 조정하는 방식(아마도 비선형적인)으로 이루어집니다.

이러한 종류의 인공 뉴런은, 많은 변형이 있지만, 실제 중요한 계산 응용도구에서 유용한 것으로 알려져있습니다. 하지만, 그것은 실제 생물학적인 뉴런이 가지는 복잡성과 계산능력을 따라가지 못합니다. 우리가 실제 뉴런들이 뇌에서 어떻게 조화롭게 작동하는지를 이해하고 모델링하기 위해서는 훨씬 더 정교한 뉴런 모델이 필요합니다.

HTM 셀

우리의 모식도에서, 가장 오른쪽에 나타나 있는 것이 HTM 피질 학습 알고리즘에서 사용되는 셀입니다. HTM 셀은 실제 뉴런의 여러 중요한 능력들을 이용하고있지만, 몇 가지 부분은 간소화시켰습니다.

몸쪽 수상돌기 (Proximal Dendrite)
각 HTM 셀은 하나의 몸쪽 수상돌기를 가집니다. 모든 아랫단계로부터의 입력은 시냅스(초록색 점으로 표시된)를 통해 셀로 들어옵니다. 시냅스의 활성도는 선형적으로 더해져서 셀의 피드포워드 활성화를 일으킵니다.

우리는 하나의 칼럼 내의 모든 셀들이 같은 피드포워드 응답을 가지도록 하였습니다. 실제 뉴런에서는 억제 세포(inhibitory cell)가 이와 유사한 역할을 하는 것으로 보입니다. HTM 에서는 단순히 한 칼럼 안의 모든 세포들이 하나의 몸쪽 수상돌기를 공유하도록 하였습니다.

주위의 셀들과의 경쟁해서 한번도 이기지 못하는 셀들이 없도록 하기 위해서, HTM 셀은 주변의 셀보다 상대적으로 덜 이기는 셀에 대해서, 피드포워드 활성화에 가중치를 부여합니다. 따라서 셀들 사이에는 안정적인 경쟁이 일어납니다. 다시 한번 강조하면, HTM 에서 이러한 경쟁은 셀 사이가 아닌, 칼럼 사이에서 일어납니다. 이러한 경쟁 과정이 모식도에는 나타나있지 않습니다. 마지막으로, 몸쪽 수상돌기는 어떤 영역에 들어온 모든 입력들의 부분집합인, 잠재적인 시냅스들의 집합과 연관되어 있습니다. 셀이 학습하는 과정에서, 몸쪽 수상돌기 위에 있는 모든 잠재적 시냅스의 "영속성"값이 증가하거나 감소합니다. 오직 임계치를 넘은 잠재적 시냅스들만이 유효한 시냅스가 될 수 있습니다.

앞에서도 이야기 했듯이, 잠재적 시냅스에 대한 개념은, 충분히 가까운 곳에 위치한 축삭돌기와 수상돌기만이 시냅스를 형성한다는 생물학적 사실에서 가져온 것입니다. 우리는 이러한 개념을 HTM 셀의 커다란 잠재적 시냅스 집합으로 확장시켰습니다. 생물학적 뉴런의 수상돌기와 축삭돌기는 학습과정에서 자라나거나 줄어들 수 있으며, 마찬가지로 잠재적 시냅스들의 집합도 변화합니다. HTM 셀의 잠재적 시냅스 집합을 크게 만듬으로써, 우리는 축삭돌기와 수상돌기가 자라나는 것과 유사한 결과를 얻을 수 있습니다. 이러한 잠재적 시냅스의 집합은 모식도에 나타나있지 않습니다.

칼럼들 사이의 경쟁과, 잠재적 시냅스 집합을 통한 학습, 칼럼의 활성을 가중시키거나 억제함으로써 HTM 뉴런들의 영역은 실제 뇌와 비슷한 놀라운 유연성을 보여줍니다. HTM 영역은, 입력이 변화하거나 칼럼의 숫자가 증가, 감소하더라도 어떤 칼럼이 무엇을 표현할지를 자동적으로 조절해나갈 수 있습니다 (몸쪽 수상돌기 위에 속한 시냅스들의 변화를 통해) .

먼쪽 수상돌기 (Distal Dendrites)

각 HTM 셀들은 먼쪽 수상돌기의 목록을 가지고 있습니다. 각 수상돌기들은 임계치 탐지기처럼 작동합니다. 어떤 수상돌기 위의 활성화된 시냅스의 숫자 (앞의 모식도에서 파란 점으로 표시된)가 임계치를 넘어가면, 그 수상돌기가 활성화되며, 관련된 셀들이 예측상태로 들어갑니다. 셀의 예측상태는 그 수상돌기의 활성정도를 OR 연산자를 통해 결정합니다.

수상돌기는 특정 시간에서 함께 활성화된 셀들과 연결을 형성함으로써 어떤 영역의 상태를 기억할 수 있습니다. 또한 수상돌기는 바로 전의 아랫단계로부터의 입력을 통해 활성화된 세포의 상태도 기억할 수 있습니다. 따라서 수상돌기는 자신이 속한 셀이 활성화될 것인지에 대한 예측과 관련되 상태를 기다릴 수 있습니다. 수상돌기의 일반적인 임계치는 15 입니다. 만약 어떤 수상돌기 위의 15 개의 유효한 시냅스가 한 번에 활성화된다면, 그 수상돌기가 활성화됩니다. 이는 주위의 수 백, 수 천개의 셀들이 활성화되었음을 의미하지만, 오직 15 개의 연결만으로 이 커다란 패턴을 충분히 인식할 수 있습니다.

각 먼쪽 수상돌기 역시 관련된 잠재적인 시냅스 목록을 가지고 있습니다. 이 잠재적 시냅스의 집합은 그 영역에 속한 모든 셀들의 집합입니다. 수상돌기가 학습해나감에 따라, 이 모든 가능한 잠재적 시냅스의 영속성 값이 증가하거나 감소합니다. 오직 임계치를 넘은 잠재적 시냅스만이 유효한 상태를 가집니다.

어떤 실제 구현 모델에서, 우리는 하나의 셀당 고정된 숫자의 수상돌기만을 이용하였습니다. 또 다른 구현 모델에서는, 훈련 과정동안 수상돌기들을 더하거나 제거하였습니다. 두 가지 방법 모두 잘 작동하였습니다. 만약 우리가 셀마다 고정된 수의 수상돌기를 가진다면, 같은 수상돌기마다 서로 다른 시냅스 집합을 저장하는 것이 가능합니다. 예를 들어, 우리가 수상돌기 위에 20 개의 유효한 시냅스를 가지고 있고, 임계치가 15 라고 가정해봅시다. (보통은 노이즈에 대한 저항성을 가지기 위해서, 임계치는 시냅스의 숫자보다 적게 설정합니다.) 그 수상돌기는 이제 근처 셀들의 하나의 특정 상태를 인식할 수 있습니다. 만약 우리가 같은 수상돌기 위해 위에 20 개의 시냅스를 추가하여 근처 셀들의 전혀 다른 상태를 표현하도록 하면 어떨까요? 이러한 방식은 오류를 일으킬 수 있는데, 그 수상돌기가 하나의 패턴으로부터는 8 개의 활성화된 시냅스를 형성하고, 다른 패턴으로부터는 7 개의 활성화된 시냅스를 가질 수 있기 때문입니다. 실험적으로 우리는 오류가 발생하기 전까지, 하나의 수상돌기에 20 개의 다른 패턴을 저장할 수 있음을 발견하였습니다. 따라서 수 십개의 수상돌기를 가진 HTM 셀은 많은 다른 예측에 참여할 수 있습니다.

시냅스(Synapses)

HTM 셀 위의 시냅스의 상태는 이진값(binary)을 이용합니다. HTM 모델에서는 시냅스 상태에 스칼라 값을 이용하지 않는데, 희소하게 분포된 표상 덕분에 아직은 스칼라 값을 이용할 필요가 없어 보입니다.

하지만, HTM 셀의 시냅스들은 학습과정동안 조절되는 "영속성"이라는 스칼라 값을 가지고 있습니다. 0.0 의 영속성 값은 잠재적인 시냅스가 유효하지 않으며, 유효한 시냅스가 되기 위한 어떠한 진전도 없었음을 의미합니다. 임계치(보통 0.2)를 넘는 영속성 값을은 그 시냅스가 방금 연결되었으며, 쉽게 연결이 끊어질 수 있음을 의미합니다. 0.9 와 같이 높은 영속성 값은, 시냅스가 연결되었음을 의미하며, 쉽게 끊어지지 않습니다.

HTM 셀의 몸쪽, 먼쪽 수상돌기에서 유효한 시냅스들의 숫자는 고정되어 있지 않습니다. 예를 들어, 먼쪽 수상돌기 위의 유효한 시냅스들의 숫자는 데이터의 시간적인 구조에 의존합니다. 영역으로 들어오는 입력에 지속적인 시간 패턴이 없으면, 먼쪽 수상돌기 위의 모든 시냅스들은 낮은 영속성 값을 가지며, 매우 적은 수의 시냅스들만이 유효한 상태를 가집니다. 입력에 많은 시간적 구조가 있다면, 높은 영속성 값을 가진 많은 유효한 시냅스를 발견할 수 있습니다.

셀의 출력(Cell Output)

HTM 셀은 두 개의 서로다른 이진 출력을 가집니다: 1)셀이 아랫단계의 입력(몸쪽 수상돌기를 통한)을 통해 활성화된 경우, 2)셀이 같은 단계의 연결(먼쪽 수상돌기를 통한)을 통해 활성화된 경우. 전자를 "활성상태(active state)"라고 부르며, 후자를 "예측상태(predictive state)"라고 부릅니다.

앞에 나타난 모식도에서, 이 두 가지 출력이 사각형의 세포체에서 나오는 두 개의 선으로 표시되어 있습니다. 왼쪽의 선이 피드포워드 활성상태를 나타내며, 오른쪽 선이 예측상태를 의미합니다.

오직 피드포워드 활성상태만이 영역 내의 다른 세포와 연결됨으로써, 예측이 항상 현재의 입력(더하기 맥락)에 기반하여 일어나도록 합니다. 우리는 예측을 바탕으로 예측을 만들어내기를 원하지 않습니다. 만약 그렇게 한다면, 몇 단계의 과정만 지나도 영역 내의 거의 모든 셀들이 예측상태에 들어가게 될 것입니다.

특정 영역의 출력은 모든 셀들의 상태를 표현하는 벡터입니다. 이 벡터들이 계층구조에서 다음 영역(만약 있다면)의 입력이 됩니다. 활성상태와 예측상태를 OR 연산자를 통해 계산하여 출력값이 만들어집니다. 활성상태와 예측상태를 조합함으로써, 한 영역의 출력은 입력보다 더

안정화(느리게 변화하는) 될 수 있습니다. 이러한 안정하는 어떤 영역의 추론을 만들어내는 중요한 특징입니다.

읽어보면 좋은 것들

신경과학(neuroscience)를 공부하기 위해 무엇을 더 읽으면 좋은지에 대한 질문을 많이 받습니다. 신경과학의 분야는 너무 넓어서, 일반적인 개론을 위해서도 여러 다른 자료를 살펴야 합니다. 여러분이 대학에 속해있지 않다면,새롭게 출판된 논문들을 읽기도, 접근하기도 어렵습니다.

이 부록에서 다룬 주제들과 관련하여, 관심있는 독자들이 살펴볼만한 좋은 책이 두 권 있습니다.

> Stuart, Greg, Spruston, Nelson, Häusser, Michael, *Dendrites, second edition*
> (New York: Oxford University Press, 2008)

이 책에는 수상돌기의 모든 것이 잘 설명되어 있습니다. 특히 16 장은 HTM 피질 학습 알고리즘에서 사용된 수상돌기의 비선형적인 특징에 대해 다루고 있습니다. 저자 Bartllet Mel 은 관련된 분야에서 누구보다 많은 생각과 연구를 하였습니다.

> Mountcastle, Vernon B. Perceptual Neuroscience: *The Cerebral Cortex*
> (Cambridge, Mass.: Harvard University Press, 1998)

이 책은 대뇌피질에 대한 아주 훌륭한 소개서입니다. 여러 장(chapter)들이 세포의 종류와 연결을 다루고 있습니다. 최근에 알려진 수상돌기의 특징을 알기에는 조금 오래되었지만, 피질 뉴런들과 그 연결이 아주 잘 설명되어 있습니다.

부록 B: 대뇌피질의 층과 HTM 영역 사이의 비교

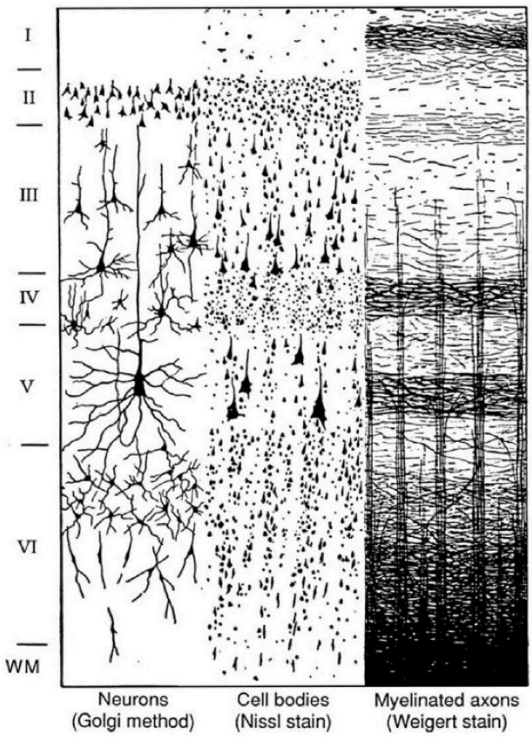
이 부록은 HTM 영역과 생물학적 대뇌피질의 영역을 서로 비교하는 내용을 담고 있습니다.

구체적으로, 이 부록은 HTM 피질학습알고리즘이 자신의 칼럼과 셀들을 가지고 어떻게 작동하는지를, 대뇌피질의 층과 칼럼구조를 통해 설명하고 있습니다. 많은 사람들이 대뇌피질의 "층(layers)"이라는 개념과 그것이 HTM 의 층과 어떠한 관련이 있는지 혼돈스러워 합니다. 이 부록이 이러한 혼란을 해결해줄 뿐만 아니라, HTM 피질학습알고리즘과 관련한 생물학적 특징에 더 많은 통찰을 주리라 기대합니다.

대뇌신피질의 회로 구조

인간의 대뇌피질은 약 1,000cm^2 의 면적과 2mm 두께를 가지는 한 장의 신경조직입니다. 이를 쉽게 떠올리기 위해, 천으로 된 식탁 냅킨을 생각해봅시다; 실제 대뇌피질과 유사한 두께와 면적을 가지고 있습니다. 대뇌피질은 수 십개의 기능적인 영역으로 나누어져 있으며, 어떤 부분은 시각을, 다른 부분은 청각을, 또 다른 부분은 언어 등을 담당합니다. 현미경으로 보면, 이 서로다른 영역들의 물리적 특징은 놀라울 정도로 비슷합니다.

대뇌피질의 각 영역들은 몇 개의 구성 원리를 가지고 있습니다.



층(Layers)

대뇌피질은 일반적으로 여섯 개의 층을 가지고 잇습니다. 다섯 개의 층은 세포를 포함하고 있고, 나머지 하는 주로 연결들로 이루어져 있습니다. 100 여년 전에 발견된 이 층 구조는 발전된 염색 기술 덕분이었습니다. 위에 나타난 이미지 (Cajal 이 그린)는 서로 다른 3 개의 염색 기법이 적용된 대뇌피질의 작은 조각을 보여주고 있습니다. 수직 축은 대뇌피질의 두께이며, 약 2mm 입니다. 이미지의 왼쪽은 6 개의 층을 보여주고 잇습니다. 가장 위에 있는 층이, 1 층이며, 세포들이 없는 층입니다. 가장 아래에 "WM"이라는 표시는 백질(white matter)의 시작을 의미하며, 세포들로부터 나온 축삭돌기가 대뇌피질이나 뇌의 다른 부분으로 여행하는 부분입니다.

이미지의 오른쪽 부분은 유수축삭돌기(myelinated axons)만을 염색한 그림입니다 (수초(myelination)는 축삭돌기의 일부분을 감싸고 있는 지방층입니다). 이 그림의 일부에서 대뇌피질의 핵심 구성 원리 두 가지인, 층과 칼럼을 볼 수 있습니다. 대부분의 축삭돌기들은 뉴런의 세포체를 떠나는 즉시 두 개로 분리됩니다. 하나의 가지는 대부분 수평한 방향으로 이동하고, 다른 가지는 주로 수직방향으로 이동합니다. 수평 방향으로 뻗은 가지는 같은 혹은 근처의 층에 있는 다른 세포로 연결되며, 위의 염색 방법을 통해 눈으로 확인할 수 있습니다. 이 그림이 대뇌피질의 단면을 그린 것임을 기억하기 바랍니다. 대부분의 축삭돌기들은 이 그림 평면을 뚫고 나오거나 들어가는 방향으로 이동하며, 따라서 실제 축삭돌기의 길이는 그림에 나타난 것보다 훨씬 더 깁니다. 대뇌피질의 매 세제곱밀리미터마다 약 2 에서 4 킬로미터 사이의 축삭돌기와 수상돌기가 있는 것으로 생각되고 있습니다.

이미지의 가운데 부분에서 사용된 염색 방법은 오직 세포체만을 보여주며, 축삭돌기나 수상돌기는 나타나있지 않습니다. 여기서 뉴런들의 크기와 밀도가 층에 따라 다름을 확인할 수 있습니다. 이 특정 그림에서 칼럼 구조의 특징은 잘 드러나있지 않습니다. 1 층에 몇 개의 뉴런들이 있는게 보일 것입니다. 1 층에 있는 뉴런들의 수는 너무 적어서 이 층은 세포가 없는 층으로 불립니다. 신경과학자들은 대뇌피질의 매 세제곱밀리미터마다 약 100,000 개의 뉴런이 있는 것으로 추정하고 있습니다.

이미지의 맨 왼쪽은 단 몇 개 뉴런들의 세포체와 축삭돌기 그리고 수상돌기를 염색한 그림입니다. 여기서 수상돌기 "나무"의 크기가 각 층에 있는 세포마다 크게 다름을 확인할 수 있습니다. 몇 개의 "선단수상돌기(apical dendrites)" 들이 세포체에서 나와 다른 층과 연결을 이루고 있습니다. 이 선단수상돌기 위치와 목적지는 각 층마다 차이가 있습니다.

요약하면, 대뇌피질의 층과 칼럼 구조는 실제 신경 조직을 염색하여 현미경으로 관찰하면 명확하게 드러납니다.

다른 영역들 간의 층 구조의 차이(Variations of layers in different regions)

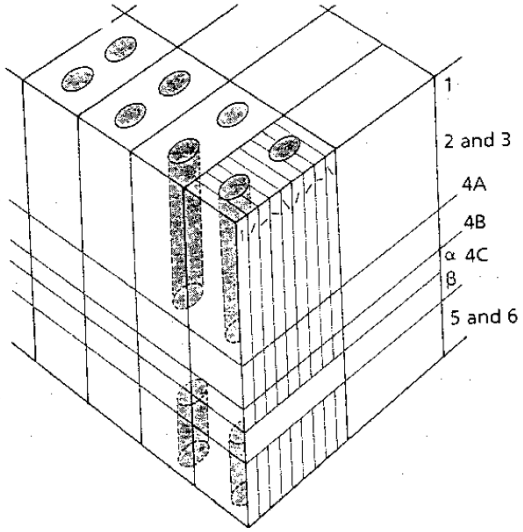
대뇌피질의 다른 영역들에 속한 층들의 두께에는 조금씩 차이가 있으며, 층의 갯수도 조금씩 다릅니다. 어떠한 동물을 대상으로 연구하였는지, 어떠한 영역을 살펴보았는지, 누가 관찰을 하였는지에 따라서 차이가 생깁니다. 예를 들어, 위의 그림에서는, 2 층과 3 층이 쉽게 구별되지만, 항상 그런 것은 아닙니다. 어떤 과학자들은 그들이 연구한 영역에서 두 개의 층을 잘 구별할 수 없다고 하였으며, 따라서 2 층과 3 층은 주로 함께 묶여서 2/3 층으로 불리기도 합니다. 다른 과학자들은 전혀 다르게, 3A, 3B 와 같은 하부 층으로 정의하기도 합니다.

4 층은 감각기관에 가장 가까운 대뇌피질 영역 중에서 가장 잘 정의된 부분입니다. 어떤 동물의 (인간과 원숭이 같은) 시각정보를 가장 먼저 처리하는 영역에서, 4 층은 명확하게 구별됩니다. 하지만 다른 동물에서는 명확히 구별되지 않습니다. 또한 감각기관과 계층구조 상에서 멀리 있는 영역에서는 대게 4 층이 관찰되지 않습니다.

칼럼(Columns)

대뇌피질의 두 번째 중요한 구성 원리는 칼럼구조입니다. 어떤 칼럼구조는 염색 상에서 확인할 수 있지만, 칼럼의 존재에 대한 대부분의 증거는, 다른 입력들에 대해 세포들이 어떻게 반응하는지를 통해 얻을 수 있습니다.

탐침을 이용하여 무엇이 뉴런을 활성화시키는지를 관찰하면서, 과학자들은 서로 다른 층에 걸쳐 수직으로 배열되어 있는 뉴런들이 같은 입력에 반응한다는 것을 발견하였습니다.



위에 제시된 모식도는 V1(망막으로부터 전달된 시각정보를 가장 먼저 처리하는 대뇌피질 영역)에 있는 세포들의 일부 응답 특성을 보여주고 있습니다.

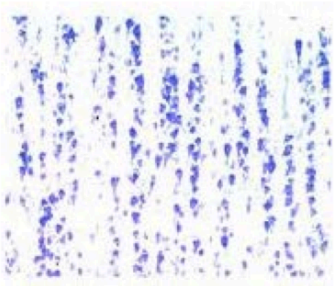
최초의 발견 중 하나는, V1 에 있는 대부분의 세포들이 망막의 특정 영역에 제시된 다른 기울기의 선과 모서리에 반응한다는 사실입니다. 칼럼 안에서 수직으로 배열되어 있는 세포들은 같은 기울기를 가진 모서리에 모두 동일하게 반응하였습니다. 위의 그림을 자세히 보면 가장 윗 부분에, 다른 기울기를 가르키는 작은 선들의 집합이 배열되어 있음을 확인할 수 있습니다. 이 선들이 그 위치에 있는 세포들이 어떠한 기울기에 반응할지를 알려주고 있습니다. 수직으로 배열되어 있는 세포들이 (얇은 수직 줄무늬 안에 있는) 같은 기울기의 선들에 반응합니다.

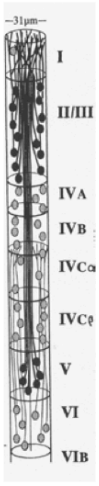
V1 에서는 칼럼구조의 다른 특징들도 발견되는데, 그림에는 두 가지가 나타나 있습니다. "안구우위칼럼(ocular dominance column)"은 왼쪽 눈과 오른 쪽 눈에 대한 반응성이 비슷한 세포들이 기둥으로 모여있는 구조를 말합니다. 그리고 주로 색상에 주로 민감하게 반응하는 "블럽(blobs)"들도 있습니다. 안구우위칼럼은 그림에서 큰 블럭으로 표시되어 있으며, 각각의 안구우위칼럼은 여러 개의 기울기 칼럼들을 포함하고 있습니다 ."블럽"들은 짙은 타원으로 표시되어 있습니다.

대뇌피질의 일반적인 원리는, 기울기와 안구우위에서처럼, 몇 개의 다른 응답 특성들이 서로 겹쳐있다는 것입니다. 만약 우리가 대뇌피질 표면 위를 수평으로 움직이다면, 세포들이 변화함에 따라 응답특성의 조합이 나타남을 볼 수 있습니다. 하지만, 수직으로 배열된 뉴런들은 같은 종류의 응답특성을 공유하고 있습니다. 이러한 수직 배열은 청각과 시각, 체성감각에서 모두 마찬가지입니다. 모든 대뇌피질에서도 마찬가지인지는 신경과학자들 사이에 논쟁이 있지만, 대부분의 영역이 이러한 특징을 보이고 있습니다.

미니칼럼(Mini-columns)

대뇌피질에서 가장 작은 칼럼구조는 '미니칼럼'입니다. 이 미니칼럼들은 지름이 약 30μm 이며, 5 개의 세포층을 거치면서 그 안에 80-100 개의 뉴런들을 가지고 있습니다. 대뇌피질 전체가 이 미니칼럼들로 이루어져 있습니다. 겹겹이 쌓여있는 아주 작은 스파게티 면 조각을 상상하면 비슷한 모습입니다. 미니칼럼들 사이에는 몇 개의 세포들이 들어찬 아주 작은 틈이 있으며, 이로 인해 염색 이미지에서 그 모습이 나타나기도 합니다.





염색 이미지의 왼쪽에 있는 것이 대뇌피질 절편(slice) 일부의 뉴런의 세포체들을 보여주고 있습니다. 이 그림에서 미니칼럼들의 수직 구조가 명확하게 드러납니다. 오른쪽에는 미니칼럼의 모식도가 나타나있습니다(Peters 와 Yilmez 의 그림). 실제는 이보다 더 얇습니다. 칼럼의 각 층마다 여러 개의 뉴런들이 있음을 주목합시다. 미니칼럼 내의 모든 뉴런들이 동일한 입력에 반응합니다. 예를 들어, 바로 전에 제시한 V1 의 단면 그림에서, 미니칼럼 안의 뉴런들은, 특정 안구우위 선호도를 가지고 특정 방향으로 기울어진 선들에 반응합니다. 근처의 미니칼럼에 속한 세포들은 조금 다른 방향이나 다른 안구우위선호도에 반응을 보일 것입니다.

억제뉴런이 미니칼럼을 정의하는데 있어 중요한 역할을 합니다. 위의 그림이나 모식도에 나타나있지는 않지만, 억제뉴런들은 미니칼럼 사이의 직선로로 축삭돌기를 뻗어, 부분적으로 이들을 물리적으로 분리시킵니다. 또한 억제뉴런은 미니칼럼에 속한 모든 셀들이 동일한 입력에 반응할 수 있도록 돕는 것으로 생각되고 있습니다.

미니칼럼은 HTM 피질학습알고리즘에서 시험적으로 사용된 칼럼의 형태입니다.

칼럼 반응의 예외사항 (An exception to columnar responses)

HTM 피질학습알고리즘과 관련한 칼럼 반응에는 한가지 예외가 있습니다. 과학자들은 보통 실험동물들에게 간단한 자극을 주어서 어떠한 세포가 반응하는지를 찾습니다. 예를 들어, 동물 시각영역의 작은 부분에 하나의 선을 보여줌으로써, V1 에 속한 세포의 반응 특성을 결정할 수 있습니다. 간단한 입력을 이용하면, 연구자들은 세포들이 언제나 같은 입력에 반응한다는 사실을 찾을 수 있습니다. 하지만, 이러한 간단한 입력이 일상의 화면들 속에 함께 있는 경우에는, 세포들이 더 민감해집니다. 따로 분리된 수직선에는 믿을만한 반응을 보여주던 세포가, 일상에서 복잡하게 움직이는 화면 안의 수직선에는 항상 반응하지 않을 수도 있습니다.

HTM 피질학습알고리즘에서, 하나의 칼럼에 속한 모든 HTM 셀들은 같은 피드포워드 입력을 받지만, 학습된 시간적 시퀀스에서는 HTM 칼럼 안의 오직 하나의 셀만이 활성화됩니다. 이러한 방법을 통해 다양한 계층의 시퀀스들을 표현할 수 있으며, 이러한 방법은 뉴런들이 가지고 있는 특성과 유사합니다. 맥락이 없는 단순한 입력은 칼럼 내의 모든 세포들을 활성화 시킵니다. 학습된 시퀀스 안의 같은 입력에서는 오직 하나의 셀만 활성화 됩니다.

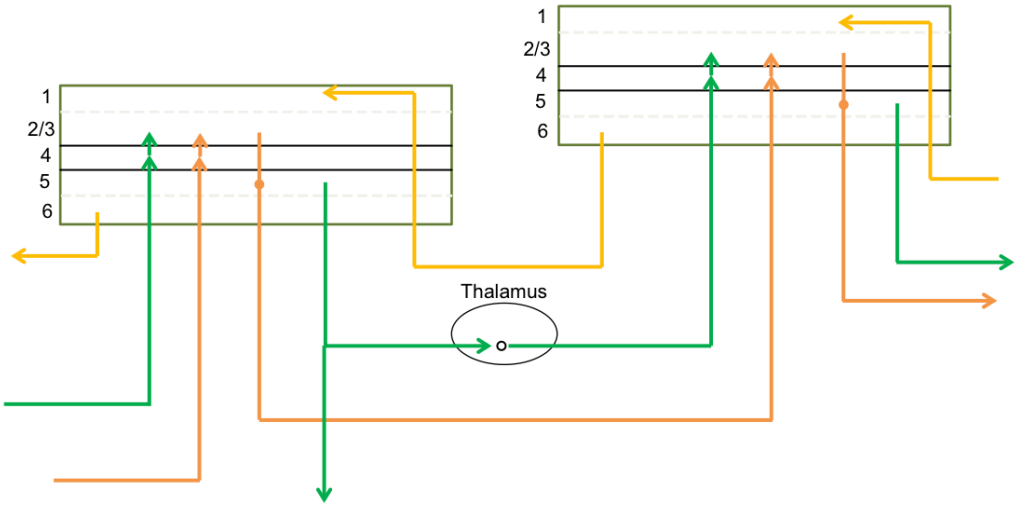
하지만 미니칼럼 내에서도 오직 하나의 뉴런들이 한 번에 활성화된다고 말할 수는 없습니다. HTM 피질학습알고리즘은 하나의 칼럼 내에서, 예측하지못한 입력에는 같은 층 안의 모든 뉴런들이 활성화되며, 예측한 입력에는 뉴런의 일부만이 활성화될 것이라고 제안합니다.

왜 층과 칼럼들이 있는 것인가요?

왜 대뇌피질에 층들과 칼럼들이 있는지는 아무도 정확히 모릅니다. HTM 이론은 하지만, 답을 제시하고 있습니다. HTM 피질학습알고리즘은 칼럼 내의 세포 층들이, 다양한 계층의 상태변화를 저장하는 대용량 메모리가 될 수 있음을 보여주고 있습니다. 더 간단히 이야기하면, 셀로 구성된 층은 많은 시퀀스들을 학습할 수 있습니다. 같은 피드포워드 응답을 공유하는 셀의 기둥들은 다차원-변화를 학습하는데있어 핵심적인 메커니즘입니다.

이러한 가설은 왜 칼럼이 필수적인지 설명해주지만, 왜 5 개의 층이 있는지도 알려줄 수 있을까요? 만약 한 개의 피질 층이 시퀀스를 배우고 예측을 만들 수 있다면, 왜 대뇌피질에는 5 개의 층이 있는 것일까요?

우리는 대뇌피질에서 관찰되는 서로 다른 층들이 모두 같은 기본 원리를 바탕으로 시퀀스를 학습하지만, 각 층에서 학습된 시퀀스들은 서로 다른 방식으로 이용될 것이라 생각하고 있습니다. 이것에 대해서 많은 것을 이해하고 있지는 않지만, 일반적인 아이디어는 이야기할 수 있습니다. 그 전에, 각 층의 뉴런들이 어떻게 연결되어 있는지를 설명하는 것이 도움이 될 것 같네요.



위에 나타난 모식도는 두 개의 대뇌피질 영역과, 두 영역 사이의 주요 연결을 나타내고 있습니다. 이러한 연결은 서로가 서로를 연결하고 있는 대뇌피질의 두 영역에서는 모두 볼 수 있는 구조입니다. 왼쪽 박스는 오른쪽의 영역(박스) 보다 계층구조에서 더 낮은 위치에 있습니다. 따라서 피드포워드 정보는 왼쪽에서 오른쪽으로 전달됩니다. 아래 방향의 화살표는 뇌의 다른 영역으로 투사됨을 의미합니다. 피드백 정보는 오른쪽에서 왼쪽으로 전달됩니다. 각 영역은 층들로 나뉘어져 있으며, 2 층과 3 층은 함께 2/3 층으로 표현되어 있습니다.

색이 있는 선은 다른 층에 속한 뉴런들의 출력을 나타냅니다. 이는 그 층의 뉴런들로부터 나오는 축삭돌기의 묶음과 같습니다. 축삭돌기들이 바로 둘로 나뉜다는 점을 기억합시다. 하나의 가지는 수평 방향으로 향하는데, 한 영역 내의 같은 층을 우선적으로 뻗어나갑니다. 따라서 각 층의 모든 세포들은 고도로 연결되어있습니다. 뉴런과 수평 연결은 모식도에 나타나있지 않습니다.

여기에는 두 개의 피드포워드 경로가 있는데, 주황색으로 표시된 직통로(direct path)와 초록색으로 표시된 우회로(indirect path)가 있습니다. 4 층은 주로 피드포워드 입력을 받는 층이며, 양쪽 피드포워드 경로로부터 입력을 받습니다. 4 층은 3 층으로 투영됩니다.

3 층 역시, 직통 피드포워드 경로가 시작되는 곳입니다. 따라서 직통로는 4 층과 3 층에 한정되어 있습니다.

어떤 피드포워드 연결은 4 층을 뛰어넘어 바로 3 층으로 전달됩니다. 그리고, 위에서도 언급했듯이, 감각 입력과 멀리있는 영역일수록 4 층은 명확히 나타나지 않습니다. 그 지점에서, 직통 피드포워드 경로는 3 층에서 시작하여 다른 영역의 3 층으로 연결됩니다.

두 번째 피드포워드 경로(초록색으로 표시된)는 5 층에서 비롯됩니다. 3 층의 세포는 다음 영역으로 뻗어나가는 도중에 5 층의 세포와 연결됩니다. 피질 층을 벗어나면, 5 층 세포에서 나온 축삭돌기는 다시 나뉘어집니다. 하나의 가지는 전달되어 운동(motor)을 일으키는뇌의 피질하부(subcortical) 영역으로 뻗어나갑니다. 이 축삭돌기들이 운동 명령(아래로 향한 화살표)을 일으키는 것으로 생각되고 있습니다. 다른 가지는 뇌의 시상(thalamus)라고 불리는 영역으로 연결되는데, 이곳은 일종의 관문과 같은 역할을 하는 곳입니다. 시상은 정보를 다음 영역으로 전달하거나 그것을 막는 역할을 합니다.

마지막으로, 일차 피드백 경로(노란색으로 나타난)가 6 층에서 시작하여 1 층으로 전달됩니다. 2,3 그리고 5 층의 세포들이 그들의 선단수상돌기(그림에는 나타나지 않은)를 통해서 1 층으로 연결됩니다. 6 층은 5 층으로부터 입력을 받습니다.

이러한 설명은 우리가 알고 있는 층과 층 사이의 연결을 간략히 요약한 것입니다. 하지만 모든 층들이 시퀀스를 학습한다면, 왜 여러 개의 층이 있는지에 대한 우리의 가설을 이해하는데에는 충분하다고 생각됩니다.

서로 다른 층들이 어떠한 일을 하는지에 대한 가설

우리는 3, 4 그리고 5 층이 모두 피드포워드 층이며, 모두 시퀀스를 학습한다고 제안합니다. 4 층은 일차원(first order)의 시퀀스를 학습니다. 3 층은 다차원(variable order)의 시퀀스를 학습합니다. 그리고 5 층은 다차원 시퀀스의 시간정보를 함께 학습합니다. 이와 관련한 세부사항들을 살펴보기로 합시다.

4 층(Layer 4)

HTM 피질학습알고리즘을 이용해서 일차원 시퀀스를 학습시키는 것은 간단합니다. 하나의 칼럼 내의 셀들이 서로를 억제할 필요가 없다면, 즉, 칼럼 안의 셀들이 이전 입력의 맥락을 분별할 필요가 없다면, 바로 일차원의 학습이 일어납니다. 대뇌피질에서 이러한 상황은 같은 칼럼 내의 세포들 사이에 작용하는 억제 효과를 제거함으로써 일어날 수도 있습니다. 우리의 HTM 피질학습알고리즘의 컴퓨터 모델에서는, 단지 하나의 칼럼에 하나의 셀을 할당하기만 하면, 비슷한 결과를 얻을 수 있습니다.

일차원의 시퀀스는 어떤 입력의 공간적 변형에 대해 변하지않는 표상을 만들기 위해 필요합니다. 시각을 예로 들면, x-y 이동, 크기 변화, 그리고 회전 등은 모두 공간 변형에 해당합니다. HTM 1 차원의 메모리를 가지는 HTM 영역에 움직이는 물체를 학습시키면, 서로 다른 공간 패턴들을 동일한 것으로 학습합니다. 이러한 HTM 셀은 대뇌피질의 소위

"복잡세포(complex cells)"처럼 활동합니다. 이 HTM 셀은 공간 변형 과정에서 계속 활성화(예측상태로)되어 있을 것입니다.

Numenta 에서 우리는 이러한 원리가 예상한대로 작동하는지 확인하는 시각실험을 수행하였습니다. 그리고 각 단계 안에 어떠한 공간 불변량이 만들어짐을 확인하였습니다. 하지만 이 실험들의 세부사항은 이 부록의 목적을 넘어서기 때문에 다루지 않도록 하겠습니다.

4 층에서 일차원 시퀀스를 학습하는 것은 4 층에서 복잡세포를 찾는 것과 마찬가지이며, 왜 대뇌피질의 더 높은 단계에 있는 영역들에서 4 층이 잘 관찰되지 않는지를 설명해줍니다. 계층구조를 따라 올라가다보면, 어디에선가 더 이상의 공간 불변량을 배울 수 없는 지점이 생기는데, 이미 그 단계에서는 표상들이 불변량으로 변해버렸기 때문입니다.

3 층

3 층은 우리가 2 장에서 언급한 HTM 피질학습알고리즘과 유사합니다. 이곳은 다차원의 시퀀스들을 학습하고 입력보다 더 안정된 예측을 만들어냅니다. 3 층은 항상 계층구조의 위 단계로 정보를 전달하는데, 이로써 계층구조내의 시간적 안정성이 증대됩니다. 다차원 시퀀스 메모리는, 3 층에서 처음 관찰된 적이 있는 소위 "방향-지향적 복잡세포(directionally-tuned complex cells)"와 유사합니다. 방향-지향적 복잡세포는, 왼쪽으로 움직이는 선 vs. 오른쪽으로 움직이는 선과 같이 시간적 맥락을 분류해냅니다.

5 층

마지막 피드포워드 층은 5 층입니다. 우리는 5 층과 3 층이 유사하지만, 3 가지 차이점이 있을 것으로 제안합니다. 첫 번째 차이는 5 층이 시간 개념을 더한다는 것입니다. 3 층은 "무엇"이 다음에 일어날지를 예측하지만, 그것이 "언제" 일어날지는 말해주지 않습니다. 하지만, 많은 작업들이 시간 정보를 필요로 합니다. 예를 들어, 대화를 인식하기 위해서는 단어 사이의 상대적인 간격이 중요한 요소입니다. 운동을 만들어내는 것도 좋은 예입니다; 근육들 사이의 조화된 타이밍이 필수적입니다. 우리는 5 층의 뉴런들이 오직 예상된 시간에만 다음 상태를 예측한다고 제안합니다. 이러한 가설을 뒷받침하는 몇 가지 생물학적인 근거가 있습니다. 하나는 대뇌피질의 운동 출력 층이 바로 5 층이라는 것입니다. 다른 근거는 5 층이 시상의 일부로부터 기원한 1 층에서 입력을 받는다는 점입니다 (모식도에는 나타나지 않았습니다). 우리는 이러한 사실들이 어떻게 시간 정보가 1 층으로 들어오는 시상 입력을 통해 여러 세포들에 부호화되고 퍼져있는지를 알려준다고 생각합니다 (모식도에는 나타나지 않았습니다).

3 층과 5 층의 두 번째 차이는, 우리가 3 층이 가능한 더 멀리 예측을 만들어내서 시간적 안정성을 가지기 원한다는 점입니다. 제 2 장에 기술된 HTM 피질학습알고리즘이 이와 같이 작동합니다. 반면, 우리는 5 층이 바로 다음 요소 (특정 시간에)를 예측하기 원합니다. 이러한

차이가 실제로 HTM 에 모델링되어 있지는 않지만, 여러 변이들이 연관된 시간 정보와 함께 저장된다면, 자연스럽게 일어날 것으로 생각됩니다.

3 층과 5 층의 세 번째 차이는 모식도에서 확인할 수 있습니다. 5 층에서 나온 출력은 항상 피질하부의 운동센터로 투사되며, 피드포워드 경로는 시상에 의해 조절됩니다. 5 층의 출력은 때때로 다음 영역으로 전달되기도 하지만, 어떤 경우에는 그 전달이 막히기도 합니다. 우리는 (그리고 다른 사람들은) 이러한 관문조절이 은밀한 주의집중(covert attention- 어떠한 행동도 만들어내지않고 입력에 집중하는 경우)과 관련이 있을 것이라고 생각합니다.

요약하면, 5 층은 구체적인 시간, 주의집중 그리고 운동 능력을 조합합니다. 이러한 일들이 어떻게 함께 일어나는지는 아직 미스테리입니다. 중요한 점은 HTM 피질학습알고리즘의 변형 모델은 구체적인 시간정보를 쉽게 포함할 수 있을 뿐 아니라, 대뇌피질의 분리된 층이 있어야함을 보여줍니다.

2 층과 6 층

아랫단계의 영역으로 정보를 되먹임하는 축삭돌기들은 6 층에서 기원합니다. 2 층에 대해서는 많은 것이 알려져있지 않습니다. 위에서도 언급했듯이, 2 층이 3 층으로부터 완전히 독립적으로 존재하는가에 대해서는 논란이 있습니다. 우리는 이러한 질문에 대해서는 더 논의하지 않을 생각이며, 2 층과 6 층이, 다른 모든 층들과 마찬가지로, 방대한 수평 연결과 칼럼 응답 특성을 가지고 있으며, 따라서 변형된 HTM 피질학습알고리즘에서 구현될 수 있을 것으로 생각합니다.

HTM 영역은 대뇌피질에서 어느 부분에 해당하나요?

우리는 HTM 피질학습알고리즘을 두 가지 방면에서 구현했습니다. 하나는 칼럼마다 여러 개의 셀을 할당하여 다차원 메모리를 만들고, 다른 방면에서는 칼럼마다 하나의 셀만 할당하여 일차원 메모리를 만들었습니다. 우리는 이 두 방식이 대뇌피질의 3 층, 4 층에 해당한다고 생각합니다. 우리는 이 두가지 다른 방식을 하나의 HTM 영역으로 통합하는 시도는 하지 않았습니다.

비록 HTM 피질학습알고리즘(칼럼 당 여러 개의 셀)이 대뇌피질의 3 층에 가장 가깝지만, 우리의 모델을 유연하게 바꾸어볼 수 있습니다- 실제 뇌에서는 할 수 없는. 따라서 특정 대뇌피질 층에 해당하지 않는 통합된 세포 층들도 융합해서 만들어낼 수 있습니다. 예를 들어, 우리의 모델에서는, 수상돌기 위에 어떤 시냅스들이 형성되어 있는지 그 순서를 알 수 있습니다. 이 정보를 이용하여 미래에 일어날 모든 일들에 대한 훨씬 일반적인 예측을 만들어낼 수도 있습니다. 아마 더 구체적인 시간 정보도 더할 수 있을 것입니다. 따라서 3 층과 5 층의 기능을 통합하는 하나의 층으로 구성된 HTM 영역을 만들 수 있습니다.

요 약

HTM 피질학습알고리즘은 우리가 대뇌피질의 신경학적 구조의 기본 단위로 생각하는 것을 담고 있습니다. 그것은 어떻게 수평으로 연결된 뉴런들이 희소하게 분포된 표상의 시퀀스를 학습하는지 보여줍니다. HTM 피질학습알고리즘의 여러 변형 모델이 관련있는 대뇌피질의 여러 다른 층에서 다른 목적을 가지고 사용될 수 있습니다.

우리는 대뇌피질로 들어가는 피드포워드 입력이, 4 층 혹은 3 층에 상관없이, 주로 몸쪽 수상돌기로 뻗어있으며, 억제 뉴런의 도움을 통해 입력의 희소분포표상을 만들어낸다고 제안합니다. 우리는 2,3,4,5 그리고 6 층의 세포들이 이러한 희소분포표상을 공유한다고 제안합니다. 이러한 일은 여러 층에 퍼져있는 칼럼 안의 모든 세포들이 같은 피드포워드 입력에 반응하도록 강요함으로써 일어납니다.

우리는 또한 4 층의 세포들이, HTM 피질학습알고리즘을 이용해서, 공간적인 변형에 대해 불변량을 만들어내는 일차원 시간 변이를 배울 수 있다고 생각합니다. 3 층의 세포들은 HTM 피질학습알고리즘을 이용하여 다차원의 시간 변이를 배우고 피질 계층구조를 통해 전달되어 올라오는 안정된 표상을 만들어낼 수 있습니다. 우리는 2 층과 4 층에 대해서는 구체적인 가설을 가지고 있지 않습니다. 하지만, 이 층들에서 관찰되는 전형적인 수평 연결은 이들 역시, 어떤 형태의 시퀀스 메모리를 학습하고 있을 가능성이 있음을 시사합니다.

용어 사전

알림: 여기에 제시된 정의는 이 문서에서 용어들이 어떻게 사용되었는지를 나타내며, 일반적인 분야에서는 다른 의미를 가지고 있을 수도 있습니다. 대문자로 표시된 용어는 이 용어 사전에서 다르게 정의된 용어를 가르킵니다.

활성상태(Active State)	피드포워드 입력에 의해 활성화된 경우의 셀(Cell)의 상태
아랫단계로부터의 (Bottom-Up)	피드포워드의 동의어
셀(Cells)	HTM 에서 뉴런과 동일한 개념 *HTM 영역에서 셀은 칼럼 안에 속해 있습니다.*
우연적 활성(Coincident Activity)	두 개 혹은 그 이상의 셀들이 동시에 활성화됨
칼럼(Column)	하나의 HTM 영역에서 한 개 혹은 그 이상의 셀들이 모여 하나의 단위로 활동하는 구조 *칼럼 안의 셀들은 같은 피드포워드 입력을 표현하지만, 서로 다른 맥락을 나타냅니다.*
수상돌기(Dendrite Segment)	셀과 칼럼과 연관되어있는 시냅스들의 집합체 단위 *HTM 에는 두 종류의 수상돌기가 있습니다. 하나는 셀과 같은 단계에 속한 연결과 관련이 있습니다. 수상돌기 위의 활성화된 시냅스의 수가 임계치를 넘으면, 연관되어 있는 셀이 예측상태로 들어갑니다. 다른 종류의 수상돌기는 칼럼으로의 피드포워드 연결과 관련이 있습니다. 활성화된 시냅스의 숫자가 더해져서 칼럼의 피드포워드 활성을 일으킵니다.*
적정밀도(Desired Density)	하나의 영역으로 들어온 피드포워드 입력에 의해 활성화되는 칼럼들의 적절한 비율 *이 비율은 정해진 반지름 이내에서만 적용되는데, 그 반지름은 피드포워드 입력의 팬 아웃(fan-out)에 의해 변화합니다. 특정 입력에 의해 이 비율이 바뀔 수 있기 때문에, "적정"이라는 용어를 사용하였습니다.*
피드포워드(Feed-Forward)	입력으로부터 멀어지는 방향 또는 계층구조에서 낮는 단계에서 더 높은 단계로 이동하는 것(Bottom-Up 으로 부르기도 함)
피드백(Feedback)	입력을 향하는 방향 또는 계층구조의 높은 단계에서 더 낮은 단계로 이동하는 것(Top-down 으로 부르기도 함)
일차원 예측 (First Order Prediction)	이전 입력과 상관없이 현재 들어오는 입력에만

	기반하여 예측을 만들어 냄- 다차원 예측과 비교
Hierarchial Temporal Memory (HTM)	대뇌피질의 구조와 알고리즘 기능 일부를 모방한 기술
계층구조(Hierarchy)	각 구성요소 사이의 연결이 피드포워드와 피드백으로 명확히 구별되는 네트워크
HTM 피질학습알고리즘 (HTM Cortical Learning Algorithms)	공간풀러, 시간풀러 그리고 HTM 영역내에서 일어나는 학습과 망각 기능들의 집합. HTM 학습알고리즘이라고도 불림
HTM 네트워크(HTM Network)	HTM 영역들의 계층구조
HTM 영역(HTM Region)	HTM 의 기억과 예측의 핵심 단위 *HTM 영역은 기둥 안에 배열된 고도로 연결된 셀들의 층으로 이루어져있다. 현재 하나의 HTM 영역은 한 개의 셀 층으로 이루어져있지만, 대뇌피질에서는 (궁극적으로 HTM 에서도), 하나의 영역이 여러 층의 셀을 가진다. 계층구조 내에서 그 위치를 이야기할 때는, '단계'라고 불리기도 한다.*
추론(Inference)	이전에 배운 패턴들과 유사한 공간, 시간 입력 패턴을 인식하는 것
억제반경(Inhibition Radius)	하나의 기둥이 억제하는 주위 영역의 넓이를 결정함
가쪽연결(Lateral Connections)	같은 영역에 속한 셀들 사이의 연결
단계(Level)	계층구조의 맥락 안에서의 HTM 영역
뉴런(Neuron)	뇌에서 정보를 처리하는 세포 *이 문서에서는, 생물학적인 세포를 의미할 때만 뉴런이라는 단어를 사용하며, HTM 의 계산 단위를 언급할 때에는 "셀"이라는 단어를 사용함.*
영속성(Permanence)	잠재적 시냅스의 연결 상태를 나타내는 스칼라 값. *임계치 미만의 영속성 값을 가지면, 그 시냅스는 아직 형성되지 않은 상태. 임계치 이상의 영속성 값은 시냅스가 유효함을 나타냄. HTM 영역에서, 학습은 잠재적 시냅스의 영속성 값을 변화시킴으로써 일어납니다.*
잠재적 시냅스(Potential Synapse)	모든 셀들의 소집합으로써 특정 수상돌기와 시냅스를 형성할 수 있음. *오직 잠재적 시냅스의 일부 집합만이 그들의 영속성 값에 따라 유효화될 수 있습니다.*
예측(Prediction)	가까운 미래에 피드포워드 입력에 의해 활성화될 것 같은 셀들을 활성화시킴 (예측상태로) *HTM 영역은 동시에 미래에 가능한 여러 입력들을 예측함*
수용영역(Receptive Field)	기둥 혹은 셀과 연결되는 입력의 집합

	HTM 영역으로 들어오는 입력이 2 차원 비트의 배열로 구성되어있는 경우, 수용영역은 입력 공간에서 반지름으로 나타낼 수 있음.
센서(Sensor)	HTM 네트워크로 들어오는 입력의 근원
희소분포표상(Sparse Distributed Representation)	여러 개의 비트로 구성되어 있지만 , 일부의 셀들만 활성화하여 표현하는 것. 하나의 비트만으로는 의미를 전달하기 힘든 경우에 사용.
공간풀링(Spatial Pooling)	입력에 대한 희소분포표상을 형성하는 프로세스 *공간풀링의 특징 중 하나는 겹치는 입력 패턴들을 같은 형태의 희소분포표상으로 투사하는 것이다.*
부차추출(Sub-Sampling)	커다란 패턴에서 활성화된 일부의 집합만으로 크게 퍼져있는 패턴을 인식하는 것
시냅스(Synapse)	학습과정 동안에 셀들 사이에 형성된 연결
시간풀링(Temporal Pooling)	출력 표상이 입력보다 더 안정된 시퀀스의 표상을 형성하는 프로세스.
탑다운(Top-Down)	피드백의 동의어
다차원 예측(Variable Order Prediction)	이전 맥락의 다양한 면을 고려하여 예측을 만들어내는 것 -일차원 예측과 비교 *"다차원"이라고 부르는 이유는 이전의 맥락을 유지하기위해 필요한만큼만 메모리가 할당되기 때문. 따라서 다차원 예측을 구현하는 메모리 시스템은 기하급수적으로 메모리가 늘어나지 않으면서도, 시간을 거슬러 형성된 맥락을 이용할 수 있습니다.*