Иерархическая темпоральная память (HTM)

и ее кортикальные алгоритмы обучения

Версия 0.2.1 от 12 сентября 2011г.

(C) Numenta, Inc. 2011

Использование программного обеспечения и интеллектуальной собственности компании Numenta, включая все идеи содержащиеся в данном документе, может осуществляться свободно (бесплатно) только для некоммерческих целей. Подробнее смотри (на английском языке):
http://www.numenta.com/software-overview/licensing.php

Разрешение (лицензия) компании Numenta на перевод

Copyright (c) 2010, 2011 Numenta, Inc.

Все права зарезервированы.

Приведенные здесь текст, алгоритмы, примеры исходных кодов, псевдокоды алгоритмов и прочие материалы были переведены, либо основаны на исходных материалах о технологии иерархической темпоральной памяти (HTM) опубликованных компанией Numenta, Inc. Компания Numenta оставляет за собой все права на исходные документы и все патентные права относящиеся к HTM и алгоритмам переведенным и опубликованным ниже. Компания Numenta согласна не рассматривать как нарушение своих патентных прав использование отдельных HTM систем или разработок на их основе при условии, что указанные действия производятся исключительно в исследовательских целях, а не в коммерческих или производственных интересах. Любое коммерческое или производственное использование технологии HTM нарушает патентные права компании Numenta и требует коммерческой лицензии.

Основываясь на вышесказанном, компания Numenta предоставляет вам права на использование представленных здесь алгоритмов и материалов только в исследовательских целях, но не для коммерческого либо производственного применения. В рамках данной лицензии термин «коммерческое либо производственное применение» включает в себя также и обучение HTM сетей с целью дальнейшего применения их, или приложений на их основе, для коммерческих или производственных целей, а также использование, либо предоставление другим права использования, результатов работы технологии HTM в коммерческих или производственных целях. Любое распространение, публикация или копирование данного документа должно включать в себя полный текст данной лицензии на перевод, как на английском языке, так и на языке перевода.

НИКАКИХ ДОПОЛНИТЕЛЬНЫХ ИЛИ ПОДРАЗУМЕВАЕМЫХ РАЗРЕШЕНИЙ НА ЛЮБЫЕ ПАТЕНТНЫЕ ПРАВА ДАННОЙ ЛИЦЕНЗИЕЙ НЕ ПРЕДОСТАВЛЯЕТСЯ. КОМПАНИЯ NUMENTA ОСОБО ПОДЧЕРКИВАЕТ ОТСУТСТВИЕ ЛЮБЫХ ОБЯЗАТЕЛЬСТВ ИЛИ ОТВЕТСТВЕННОСТИ ЗА КАЧЕСТВО ИЛИ АККУРАТНОСТЬ ЛЮБЫХ ПЕРЕВОДОВ СДЕЛАНЫХ В РАМКАХ ДАННОЙ ЛИЦЕНЗИИ.

Numenta Translation License

Copyright (c) 2010, 2011 Numenta, Inc.

All rights reserved.

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.

Сначала прочтите это!

Это предварительная версия данного документа. Некоторые вещи в ней пока просто отсутствуют, и мы должны четко уведомить вас об этом.

Что ЕСТЬ в этом документе:

Этот документ детально описывает новые алгоритмы обучения и предсказания состояний, разработанные в компании Numenta в 2010 году. Эти новые алгоритмы описаны здесь достаточно детально, чтобы программисты смогли их полностью понять и, по желанию, самостоятельно имплементировать. Документ начинается с водной (теоретической) главы [после предисловия]. Если вы и до этого следили за работами компании Numenta и читали некоторые из наших предыдущих документов, то эта вводная глава будет для вас знакомой. Остальной материал является новым.

Чего НЕТ в этом документе:

Есть несколько тем, относящихся к имплементации описанных новых алгоритмов, которые не вошли в эту предварительную версию документа:

- Хотя большинство аспектов описанных алгоритмов было нами уже имплементировано и протестировано в виде программных приложений, результаты этих тестов пока не были включены в данный документ.

- Отсутствуют описания, как эти алгоритмы могут быть применены к решению практических задач. Отсутствует описание, как вы могли бы конвертировать исходные данные с сенсоров или из базы данных в разреженные представления, подходящие для этих алгоритмов.

- Эти алгоритмы самообучения могут работать в режиме on-line поступления данных. Однако, ряд деталей, необходимых для полной имплементации этого режима (для некоторых редких случаев), здесь не описаны.

- Нами запланировано несколько дополнительных разделов, которые пока не включены в данный документ. Туда должны войти приложения с обсуждением свойств разреженных пространственных представлений, разделы о практических приложениях и примерах, а также ссылки на первоисточники.

Мы решили опубликовать данный документ в его текущем виде, поскольку мы считаем, что описанные в нем алгоритмы могут быть интересны и для других людей. Отсутствующие части данной версии документа не должны затруднить понимание и эксперименты с описанными алгоритмами для заинтересованных исследователей. Мы будем регулярно обновлять данный документ, отражая в нем наши достижения.

Дополнительно: ссылки на оригинал документа

Данный документ был взят с официального сайта компания Numenta (www.numenta.com). Его оригинальное название – «Hierarchical Temporal Memory including HTM Cortical Learning Algorithms»; текущая ссылка на него (формат PDF):

http://numenta.com/htm-overview/education/HTM_CorticalLearningAlgorithms.pdf

Дополнительно: обращение переводчика

Автор перевода будет благодарен за все присланные замечания и предложения направленные на улучшение качества и удобочитаемости перевода данного документа. Прошу присылать их на мой почтовый адрес mih.net(эт)yandex.ru.

Все примечания переводчика в данном тексте выделялись [квадратными скобками].

Содержание

Предисловие

Существует множество вещей, которые люди делают очень легко, а компьютеры на это сейчас просто не способны. Такие задачи, как распознавание визуальных образов, понимание разговорной речи, распознавание и манипулирование объектами с помощью только осязания, равно как и навигация в сложном окружающем мире, довольно просто решаются людьми на практике. Тем не менее, не смотря на десятилетия упорных исследований, существует мало заметных алгоритмов для реализации подобных человеческих навыков с помощью компьютеров.

В человеке все эти способности обеспечиваются, в основном, с помощью коры головного мозга. Иерархическая темпоральная память (Hierarchical Temporal Memory – **HTM**) это технология, моделирующая то, как в коре головного мозга выполняются указанные функции. HTM нацелена на построение машин, которые достигнут или даже превзойдут уровень человека в решении множества когнитивных задач.

Данный документ описывает указанную технологию HTM. Глава 1 дает вводный обзор HTM, подчеркивая важность иерархической организации, разреженность пространственных представлений, и запоминание временных событий. Глава 2 детально описывает алгоритмы обучения регионов HTM. Главы 3 и 4 содержат в себе псевдокод для алгоритмов обучения HTM поделенный на две части: для пространственного группировщика и для темпорального группировщика. После прочтения глав со 2-ой по 4-ую данного документа, опытный инженер-программист сможет самостоятельно воспроизвести указанные алгоритмы и поэкспериментировать с ними. Кроме того, мы надеемся, что некоторые из читателей пойдут еще дальше и разовьют для себя нашу работу.

Назначение документа, его аудитория

Данный документ предназначен для технически образованной аудитории. Хотя мы и не требуем от вас никаких предварительных знаний по нейрофизиологии, мы все же полагаем, что вы в состоянии понимать простые математические и компьютерные концепции. Мы создавали данный документ так, чтобы он мог быть назначен в качестве дополнительного чтения, на курсах соответствующих ВУЗ-ов. Нашими основными воображаемыми читателями мы предполагали студентов компьютерных или когнитивных наук; либо же разработчиков программных приложений, которые интересуются созданием искусственных когнитивных систем, работающих по тем же принципам, что и человеческий мозг.

Но даже читатели без технического образования все равно могут получить пользу от чтения отдельных разделов данного документа, особенно главы 1 «Введение в HTM».

Предыдущие аналогичные документы

Часть теории HTM была уже описана в книге [Джефа Хокинза] «О Интеллекте» («On Intelligence», 2004г.), в пояснительных документах (white papers), опубликованных компанией Numenta, а также в ряде рецензируемых статей, написанных сотрудниками компании Numenta. Однако мы не предполагаем, что вы читали что-либо из этого предварительного материала, большая часть которого была включена и переработана в данном документе. Обратите внимание, что алгоритмы обучения HTM, описанные в главах 2- 4, нигде ранее нами не публиковались. Эти новые алгоритмы заменили собой первое поколение алгоритмов, называемых Zeta-1. Какое-то короткое время мы их называли алгоритмами «Распределенных Представлений Фиксированной плотности» («Fixed-density Distributed Representations» – FDR) , но мы больше не используем такую терминологию. Теперь мы их называем «кортикальными алгоритмами обучения HTM» (HTM Cortical Learning Algorithms) или просто алгоритмами обучения HTM.

Мы рекомендуем вам прочитать книгу «О Интеллекте», написанную одним из основателей компании Numenta Джефом Хокинзом (Jeff Hawkins) совместно с

Сандрой Блэйксли (Sandra Blakeslee). Хотя в указанной книге и не упоминается напрямую о технологии НТМ, в ней дается простое, не техническое, описание базовой теории НТМ и основных знаний из нейрофизиологии, на которых она основана. Однако, во время написания указанной книги, хотя мы уже понимали основные принципы работы НТМ, но мы тогда совсем еще не знали, как их реализовать в алгоритмах. Таким образом, данный документ можно рассматривать как продолжение работы начатой в книге «О Интеллекте».

О компании Numenta

Компания Numenta, Inc. (www.numenta.com) была создана в 2005 году для разработки технологии НТМ как для коммерческого, так и для научно-исследовательского применения. Для достижения этой цели мы полностью документируем и публикуем все наши достижения и открытия. Кроме того, мы публикуем наше программное обеспечение в форме, позволяющей другим людям применять его и в своих исследовательских и в коммерческих разработках. Мы с самого начала так разрабатывали и структурировали наше ПО, чтобы впоследствии, на его основе, стимулировать появление независимого сообщества разработчиков приложений. Использование ПО и интеллектуальной собственности компании Numenta бесплатно для исследовательских целей. Мы планируем получать доход от продажи поддержки, лицензий на ПО и лицензий на интеллектуальную собственность для коммерческих применений. Мы всегда будем стремиться сделать наших партнеров-разработчиков настолько же успешными, насколько и мы сами.

Компания Numenta расположена в городе Редвуд (Redwood City), штат Калифорния. Она финансируется из частных источников.

Об авторах документа

Данный документ является плодом коллективных усилий сотрудников компании Numenta. Имена основных авторов каждого раздела документа перечислены в истории изменений документа.

История изменений документа

В нижеследующей таблице нами отмечены основные изменения между версиями данного документа. Малые изменения, вроде небольших пояснений или форматирования, не отмечены.

Версия	Дата	Изменения	Основные авторы
0.1	9 ноября 2010г.	1. написаны первые варианты Предисловия, глав 1,2,3,4 и Глоссария.	Джеф Хокинз (Jeff Hawkins), Субутай Ахмад (Subutai Ahmad), Донна Дубински (Donna Dubinsky)
0.1.1	23 ноября 2010г.	1. В главе 1 раздел о Регионах был отредактирован, чтобы уточнить термины, такие как уровни, колонки и слои. 2. Написан первый вариант Приложения А.	Джеф Хокинз (Jeff Hawkins)

0.2	10 декабря 2010г.	1. Глава 2: различные уточнения. 2. Глава 4: исправлены ссылки на строчки кода; изменен код в строках 37 и 39. 3. Приложение Б: написан первый вариант.	Джеф Хокинз (Jeff Hawkins) Субутай Ахмад (Subutai Ahmad), Джеф Хокинз (Jeff Hawkins)
0.2.1	12 сентября 2011г.	1. Сначала прочтите это: удалена ссылка на 2011г. 2. Предисловие: Удален раздел о выпуске ПО.	

Глава 1. Введение в НТМ

Иерархическая темпоральная память (Hierarchical Temporal Memory - HTM) является технологией машинного самообучения, которая нацелена на повторение структурных и алгоритмических свойств коры головного мозга.

Именно кора головного мозга млекопитающих является местом где находятся мысли и сознание. Высокоуровневое зрение, слух, осязание, движения, язык и планирование, все это создается корой головного мозга. Наблюдая такое разнообразие когнитивных функций, вы можете подумать, что в коре мозга заложен и эквивалентный ему набор специализированных нейрональных алгоритмов. Но это, пожоже, не так. Кора мозга демонстрирует нам на удивление постоянный паттерн нейрональных связей. Биологические данные позволяют нам предположить, что в них реализова некоторый общий набор алгоритмов, который позволяет выполнить множество различных функций сознания.

Технология НТМ включает в себя набор теоретических разработок (theoretical framework) для понимания принципов работы коры мозга и ее многочисленных возможностей. И хотя, на сегодняшний день, мы имплементировали в ПО лишь малую часть указанной теоретической базы, мы надеемся, что со временем все больше и больше ее теоретических построений будет реализовано на практике. Тем не менее, уже сейчас, как мы полагаем, нами воссоздана значительная часть основных механизмов деятельности коры, которые являются ценными сами по себе для множества коммерческих и научных применений.

Программирование НТМ совершенно не похоже на традиционное программирование компьютеров, когда программисты создают жесткие программы для решения конкретных, четко очерченных, проблем. В отличии от них НТМ обучается путем работы с входным потоком сенсорных данных. И ее конечные способности во многом определяются тем, что было ей продемонстрировано.

НТМ можно рассматривать и как новый тип нейронных сетей. Ведь согласно определению, любая система, которая пытается моделировать архитектурные особенности коры мозга, является нейронной сетью. Однако, сам по себе термин «нейронная сеть» не очень хорош, поскольку его применяют к слишком широкому классу различных систем. В модели НТМ нейроны (которые в НТМ называют просто клетками) организованы в колонки, слои, регионы и иерархии регионов. Все эти детали очень важны, как вы увидите сами, и про НТМ можно смело сказать, что это совершенно новый вид нейронных сетей.

Как и следует из ее названия (Иерархическая темпоральная память), технология НТМ в основе своей является некоторой системой памяти. Сети НТМ обучаются много-много раз на изменяющихся входных данных и механизм их работы основан на запоминании большого множества паттернов и их последовательностей. Однако, способ, с помощью которого эти данные сохраняются и «вспоминаются» совершенно отличается от модели используемой в современном программировании. Классическая компьютерная память имеет линейную организацию и никак не связана («не имеет понятия о») с категорией времени. Программисты, при этом, сами могут организовать любой вид организации данных и их структуру поверх этой линейной компьютерной памяти и контролировать, как и где в них будет храниться информация. По сравнению с этим, память НТМ накладывает гораздо большие ограничения в своем использовании и неотъемлемо связана с параметром времени. В ней информация всегда хранится в распределенном виде. Пользователь НТМ указывает только параметры иерархии и чему она будет обучаться, а НТМ уже сама контролирует где и как будет храниться информация.

И хотя сети НТМ настолько отличаются от классических компьютеров, мы вполне можем использовать последние для моделирования работы НТМ, если только мы реализуем в них основные функции иерархии, времени и разреженных пространственных представлений (подробно описанных далее). Мы предполагаем, что со временем будет создано и специализированное аппаратное обеспечение для более эффективной реализации работы прикладных сетей НТМ.

В данном документе мы будем часто иллюстрировать свойства НТМ с помощью

примеров позаимствованных из областей человеческого зрения, осязания, слуха, разговорного языка и поведения. Такие примеры очень полезны, поскольку они позволяют читателям легко схватывать их суть. Однако очень важно помнить, что возможности НТМ не привязаны к каким-то конкретным видам данных. Ее можно использовать в работе не только с человеческими сенсорными потоками, но и, например, с данными от радаров, инфракрасного («ночного») зрения, или даже потока данных с финансовых рынков, метеорологических данных, данных о веб трафике или с чистым текстом. Описанные здесь механизмы обучения и предсказания, заложенные в НТМ, могут быть применены к самым различным типам задач.

Принципы НТМ

В этом разделе мы рассмотрим некоторые базовые принципы НТМ: почему в ней так важна организационная иерархия, как структурированы регионы НТМ, почему данные в ней хранятся в виде разреженных пространственных представлений и почему понятие о времени здесь настолько критически важно.

Иерархия

Сеть НТМ состоит из регионов организованных в иерархию. Регион НТМ – это основной ее строительный блок, функциональная единица ее памяти и способности к предсказанию, который мы рассмотрим более подробно в следующей главе. Типичным случаем является, когда один регион представляет из себя один уровень в иерархии НТМ. И по мере восхождения вверх по этой иерархии всегда присутствует конвергенция данных, когда многие элементы дочернего (нижнего) региона соединяются (конвергируют) на одном элементе родительского (верхнего) региона. Тем не менее, благодаря наличию обратных связей, информация также и дивергирует обратно (разделяется) при движении вниз по уровням иерархии. (В данном случае термины «регион» и «уровень» являются практически синонимами. Мы используем термин «регион» когда описываем его функциональный аспект, тогда как термин «уровень» используется специально для описания места региона в иерархии НТМ.)

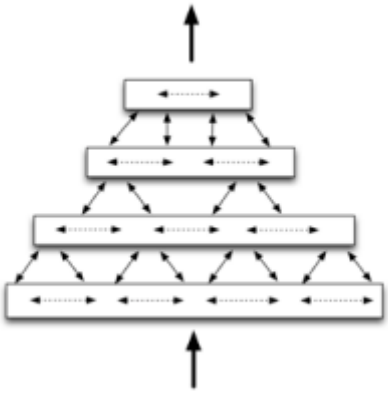


Рисунок 1.1. Упрощенная диаграмма четырех регионов НТМ, организованных в четырехуровневую иерархию. Показаны потоки информации внутри уровня, между уровнями и извне/наружу для всей этой иерархии.

Кроме того, возможно скомбинировать вместе несколько НТМ сетей. Такого типа структура имеет смысл, когда у вас есть потоки данных от более чем одного вида сенсоров. Например, одна из сетей может обрабатывать аудио информацию, а другая – визуальную информацию. А конвергенция от каждой из сетей будет происходить только в верхней части иерархии.

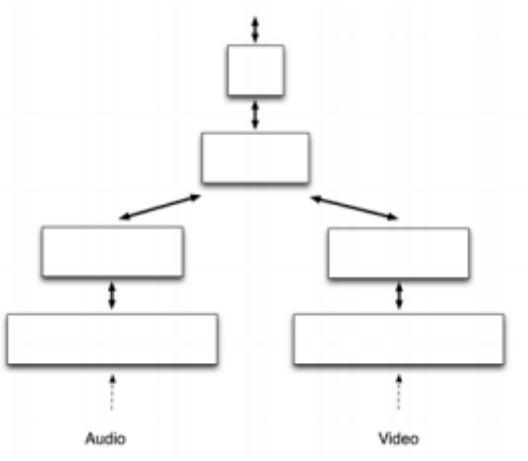


Рисунок 1.2. Конвергенция сетей от различных сенсоров.

Преимуществом иерархической организации является ее высокая эффективность. Она существенно сокращает время обучения и необходимые объемы памяти, поскольку паттерны выученные на каждом уровне иерархии используются многократно в комбинациях на более высоких уровнях. Для иллюстрации, давайте рассмотрим обработку визуальной информации. На самом нижнем уровне иерархии ваш мозг хранит информацию о маленьких кусочках визуального поля, таких как линии (границы) и углы. Линия (граница) это один из основных элементов любого визуального образа. Такие паттерны низкого уровня рекомбинируются на средних уровнях в более сложные компоненты, такие как кривые и текстуры. Например, простая дуга может быть и верхней гранью человеческого уха, и верхом рулевого колеса автомобиля или же ободком кружки кофе. Далее, эти паттерны среднего уровня потом комбинируются в представления высокоуровневых объектов, таких как лица, машины или дома. Чтобы запомнить такой высокоуровневый объект, вам не нужно заново изучать все его мелкие компоненты.

В качестве другого, аналогичного примера, можно напомнить, что когда вы запоминаете новое слово, вам не нужно заново изучать его буквы, слоги или фонемы.

Совместное использование репрезентаций в иерархии также ведет ко все большему обобщению в ожидаемом поведении. Когда вы видите новое животное и видите у него рот и зубы, то вы можете предсказать, что это животное ест с помощью своего рта и что оно может вас укусить. То есть иерархия позволяет совершенно новому объекту унаследовать некоторые известные свойства своих субкомпонент.

Сколько всего может запомнить один уровень в HTM иерархии? Или, поставив вопрос по другому, сколько уровней в иерархии нам необходимо? Здесь существует некоторый компромисс между тем сколько памяти отведено на каждый уровень и сколько всего уровней потребуется. К счастью, HTM автоматически запоминает самые лучшие репрезентации на каждом своем уровне, при заданной исходной статистике входных данных, и при выделенных ей на это ресурсах. Если вы дадите ей больше памяти на некоторый уровень, то они сформируют у себя более сложные и большие репрезентации, а следовательно, вам возможно потребуется меньше уровней HTM. А если вы дадите меньше памяти каждому уровню HTM, то они будут формировать репрезентации которые проще и меньше, что может потребовать большего количества необходимых уровней в иерархии.

До сего момента мы рассматривали такие сложные задачи, как распознавание визуальных образов. Однако многие практические задачи гораздо проще зрения и даже один регион HTM может оказаться для них вполне достаточен. Например, мы применили HTM для предсказания, куда посетитель веб сайта вероятнее всего кликнет дальше на странице. Решение этой проблемы потребовало направить на HTM поток данных о веб кликах посетителей. В ней практически не было никакого

пространственного аспекта, а решение требовалось в области временной статистики, то есть предсказания на основе распознания типичного профиля посетителя. Темпоральные алгоритмы самообучения НТМ просто идеальное средство для решения подобных задач.

Подводя итог, можно сказать, что иерархии сокращают время обучения, использование памяти и приводят к некоторой большей обобщенности данных. Тем не менее, многие задачи на простое предсказание могут быть решены и с помощью одного региона НТМ.

Регионы

Само понятие регионов, связанных между собой в иерархию, было нами позаимствовано из биологии. Кора головного мозга представляет собой тонкую нейронную ткань толщиной примерно 2 мм. Нейрофизиологи делят кору на различные зоны, или области, базируясь в основном на их связях между собой. И именно эти связи областей мозга и определяют их иерархию.

Все области коры мозга выглядят похожими друг на друга по своему внутреннему строению. Они весьма сильно варьируются по своему размеру и по своему месту в иерархии, но во всем остальном они похожи. Если вы возьмете их срез, высотой в два миллиметра, вы увидите шесть слоев - пять слоев клеток и один без них (из этого правила есть несколько исключений, но в общем оно соблюдается). Каждый из нейронных уровней областей коры мозга имеет множество взаимосвязанных клеток, организованных в колонки. Также и регионы НТМ представляют из себя слои сильно взаимосвязанных клеток, также организованных в колонки. «Уровень 3» нейронов в коре мозга является основным приемником входящих внешних сигналов. Клетки НТМ региона приблизительно похожи на нейроны уровня 3 в регионе коры мозга.

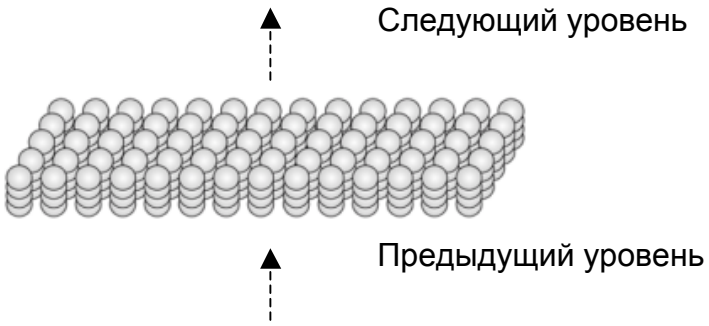


Рисунок 1.3. Часть НТМ региона. Многочисленные клетки НТМ региона организованы в двухмерный массив колонок. На рисунке показан регион с 4 клетками в каждой колонке. Каждая колонка связана с некоторым подмножеством входных данных региона и каждая клетка в колонке соединена с другими клетками в регионе (эти связи не показаны). Заметьте, что этот НТМ регион, включая его колончатую структуру, является эквивалентом только одного слоя нейронов в области коры головного мозга.

Хотя регион НТМ является эквивалентом только части области коры мозга, он вполне способен делать распознавания и предсказания в сложных потоках входных данных, что позволяет его использовать при решении многих задач.

Пространственные разреженные представления

Хотя нейроны в коре мозга очень плотно связаны между собой, многочисленные подавляющие (ингибиторные) нейроны гарантируют, что одновременно будут активен только небольшой процент всех нейронов. То есть, получается, что информация представляется в мозге всегда только небольшим количеством активных нейронов из всех имеющихся там. Такой тип кодирования информации называется «разреженное распределенное представление». «Разреженное» означает, что только небольшой процент нейронов коры может быть одновременно активным. «Распределенное» означает, что для представления

чего-либо в коре необходима активация многих нейронов. Активность одиночного нейрона, конечно, тоже что-то означает, но она должна интерпретироваться только в контексте множества других нейронов, чтобы можно было понять ее общее значение.

HTM регионы также как и области коры мозга используют в своей работе разреженные распределенные репрезентации. На самом деле, механизмы памяти HTM настолько от них зависят, что по другому просто не будут работать. Входной сигнал для HTM это тоже всегда распределенное его представление, но оно может не быть разреженным. Поэтому, первое, что делает HTM регион, это конвертация такого входа в разреженное распределенное представление. Например, пусть на входе в регион имеется 20 000 бит. И процент нулей и единиц среди них может существенно меняться во времени. В один момент тут может быть 5 000 единичек, а в другой момент – 9 000 единичных битов. HTM регион может конвертировать такой вход во внутреннее представление из 10 000 бит, из которых только около 2% (200 бит) будут активными не зависимо от количества единичек на входе. Поскольку входные данные HTM региона будут меняться со временем, это внутренние их представление (репрезентация) будет также меняться, но в ней всегда будет оставаться примерно 200 активных битов из 10 тысяч.

Может показаться, что при таком подходе происходит большая потеря информации, поскольку общее возможное число входных паттернов получается гораздо большим, чем число возможных репрезентаций в HTM регионе. Но, на самом деле, оба этих числа просто непомерно огромные. И реальное число всех видов входов, увиденных когда либо регионом HTM, будет просто букашкой по сравнению с общим возможным их числом. Позднее мы подробно опишем, как регион создает разреженные представления входных данных. Упомянутая нами теоретическая возможность потери информации, на практике не имеет существенного эффекта.

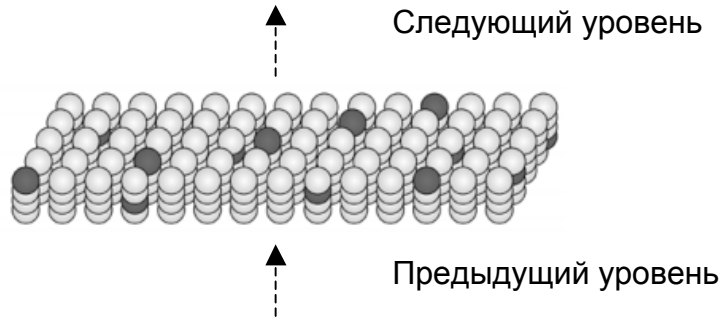


Рисунок 1.4. Часть HTM региона с показанным примером разреженного представления с помощью активных клеток.

Разреженные распределенные репрезентации имеют несколько очень полезных свойств и являются неотъемлемой частью работы HTM. Позднее мы еще к ним вернемся снова.

Роль времени

Время играет критически важную роль в обучении, распознавании и предсказании.

Давайте начнем с распознавания объектов. Без использования фактора времени, мы бы практически ничего не смогли понять из потока данных, приходящих от наших тактильных или аудиальных сенсоров. Например, если вам завязать глаза и кто-то положит вам в руку яблоко, то вы сможете понять, что это такое всего лишь через секунду или около того. И по мере движения ваших пальцев вокруг яблока (что меняет поступающую к вам тактильную информацию) сам объект (яблоко), равно как и ваше высокоуровневое представление о яблоке, остаются неизменными. Однако, если это яблоко положить вам на раскрытую ладонь, и полностью запретить двигать рукой и пальцами, то вам будет гораздо сложнее отличить это яблоко от, скажем, лимона.

То же самое верно и для слуха. Статичный звук имеет для нас мало смысла. Слово вроде «яблоко» или хрустящий звук, когда кто-то откусывает кусок яблока, могут быть распознаны только после восприятия сотен кратких последовательно меняющихся во времени звуков различных спектров.

Напротив, зрение является более сложным случаем. В отличие от осязания и слуха, люди способны распознать изображение, которое предъявляется им так быстро, что у них нет никакой возможности совершить какое-либо движение глазами (например, саккаду). То есть наше визуальное распознавание образов не всегда требует изменения входных данных во времени. Тем не менее, в процессе обычного зрительного восприятия мы постоянно двигаем нашими глазами, головой, телом и, соответственно, объекты окружающего мира двигаются вместе с ними тоже. Наша описанная выше возможность распознавания столь быстрых образов является особым случаем, который становится возможным благодаря статистическим свойствам зрения и годам тренировок. В общем случае, для нормальной работы зрения, слуха и осязания требуются изменяющиеся во времени входные данные.

Рассмотрев все эти случаи распознавания, давайте теперь посмотрим на обучение. Чтобы обучиться, на НТМ нужно подать переменный во времени поток данных. Даже для зрения, где иногда возможно статическое распознавание, мы должны увидеть многие изменяющиеся изображения объекта, чтобы понять и запомнить, как он выглядит. Например, представьте себе, что на вас бежит собака. В каждый момент времени изображение собаки проецируется на сетчатку нашего глаза, создавая некоторый паттерн. Вы воспринимаете эти паттерны как различные виды одной и той же собаки, хотя строго математически все эти паттерны совершенно разные. Однако наш мозг понимает, что все эти различные паттерны означают один объект, наблюдая их в одной временной последовательности. Таким образом, время здесь является «инструктором» («супервизором»), который говорит вам, какие пространственные паттерны идут вместе.

Обратите внимание, что не просто изменение входа важно для нашего сенсорных чувств. Последовательное восприятие не связанных между собой входных паттернов приведет нас только к замешательству. Изменяющиеся во времени входные данные должны приходить к нам от одного источника из реального мира. И это касается не только наших человеческих чувств, хотя мы их используем тут в качестве примеров. Если мы захотим обучить НТМ распознавать данные с датчиков температуры, вибрации и шума на теплоэлектростанции, то нам нужно будет ее обучать на наборах данных именно от них, меняющихся во времени.

Типично, когда для обучения сети НТМ требуется достаточно много входных данных. Вы сами научились распознавать собак, только увидев многих их представителей, а не одно изображение только одной собаки. Задачей алгоритмов обучения НТМ является извлечение временных последовательностей из потока входных данных с целью построения модели, какой из паттернов следует в ней за каким другим. Это не просто, поскольку может быть не известно, когда последовательность паттернов начинается и когда заканчивается, на входе могут одновременно присутствовать перекрывающиеся последовательности, обучение должно происходить постоянно и возможно присутствие всевозможного шума во входных данных.

Запоминание и распознание последовательностей является основой для выработки предсказаний. После того как НТМ выучила какие паттерны скорее всего следуют за какими, она может предсказать появление следующих паттернов на основе текущих и последних входных данных. Подробнее мы рассмотри работу предсказаний в НТМ позднее.

Теперь мы вернемся к четырем основным функциям НТМ: обучение, распознавание, предсказание и поведение. Каждый НТМ регион выполняет первые три функции. И лишь поведение исключено из этого ряда. И хотя мы знаем из нейрофизиологии, что большинство областей коры мозга как-то участвуют в поведении, мы полагаем, что, на данный момент, для большинства практических приложений это не является необходимым. Поэтому, мы не включили какую либо форму реализации поведения в наше текущее воплощение алгоритмов НТМ, хотя и упоминаем о нем здесь для общей полноты картины.

Обучение

Регион НТМ изучает свой мир путем нахождения паттернов и их последовательностей в своем входящем потоке данных. Регион, сам по себе, «не знает», что собой представляют эти данные, он работает чисто на статистических принципах, высматривая комбинации входных битов, которые часто появляются вместе, и которые мы называем пространственными паттернами. Потом регион ищет, как эти паттерны образуют последовательности во времени, что мы назовем временными паттернами или просто последовательностями.

Например, если регион НТМ получает данные с метеорологических датчиков на здании, он может обнаружить, что определенная комбинация температуры и влажности часто появляется на южной стороне здания, а другая комбинация – на северной его стороне. И потом он может обнаружить, что эти комбинации следуют одна за другой каждые сутки.

А если регион НТМ получает данные о текущих покупках в магазине, он может обнаружить, что определенные товары чаще покупаются в выходные, или, что по вечерам более предпочтителен определенный ценовой диапазон, при условии, что на улице холодная погода. Кроме этого, он может открыть, что различные классы людей следуют похожим последовательным паттернам в своих покупках.

При этом, один регион НТМ ограничен в своих возможностях по самообучению. Он автоматически размещает у себя то, чему он обучился, в зависимости от фактически имеющегося у него количества памяти и от наблюдаемой сложности своих входных данных. Если доступной региону памяти становится меньше, то и пространственные паттерны, которые он запоминает, соответственно становятся проще. Либо наоборот, они могут стать более сложными при увеличении доступной памяти. Если выученные регионами паттерны относительно просты, то для распознавания достаточно сложных образов может потребоваться большая иерархия. Подобные же характеристики мы можем наблюдать и в системе человеческого зрения, где зоны коры, получающие данные от сетчатки, запоминают пространственные паттерны маленьких участочков визуального пространства. И только после нескольких уровней иерархии визуальных зон эти пространственные паттерны комбинируются и представляют уже практически все визуальное пространство.

Подобно биологическим системам, алгоритмы обучения НТМ региона способы к обучению «на лету», то есть к непрерывному обучению во время работы, от каждого входного паттерна. Поэтому нет необходимости в отделении фазы распознавания от фазы обучения, хотя распознавание заметно улучшается при дополнительном обучении. Таким образом, при изменении паттернов на входе, НТМ регион постепенно также изменится вместе с ними.

После начального обучения НТМ может продолжить самообучаться, либо ее самообучение может быть отключено после фазы начального тренинга. Еще одним вариантом будет отключить самообучение только у низших слоев иерархии регионов, но продолжить самообучение у высших слоев. Если НТМ уже выучила самые базовые статистические структуры из окружающего ее мира, основное обучение будет происходить только на верхних уровнях иерархии. А если НТМ получит новые паттерны, в которых будут не наблюдавшиеся ранее структуры низшего уровня, то потребуется больше времени на изучение таких паттернов. То же самое мы можем наблюдать и у людей. Изучение еще одного слова в языке, который для вас знаком, проходит относительно просто. Однако, если вы пытаетесь выучить незнакомое слово иностранного языка с незнакомыми вам звуками, это будет для вас гораздо сложнее, поскольку вы не освоили его низкоуровневые фонемы.

Даже такое простое обнаружение паттернов - это потенциально очень полезное практическое свойство. Понимание высокоуровневых паттернов во флюктуациях финансового рынка, течении болезни, погоде, доходе от производства или в отказах сложных систем (таких как региональные электросети), весьма ценно само по себе. И при этом, изучение пространственных и временных паттернов является предшественником для распознавания и предсказания в НТМ.

Распознавание

После того как НТМ выучила паттерны из своего мира она может проводить распознавание новых входных данных. При получении очередного входа НТМ будет сопоставлять его выученным ранее пространственным и временным паттернам. Успех такого сопоставления составляет основу распознавания паттернов.

Подумайте о том, как вы распознаете музыкальную мелодию. Первая ее нота очень мало о чем скажет вам. Вторая нота, возможно, существенно сузит круг потенциальных вариантов, но все равно их может оставаться еще слишком много. Обычно требуется три – четыре, или более, нот для того, чтобы вы узнали знакомую мелодию. Распознавание в НТМ регионе очень похоже на этот процесс. Регион постоянно просматривает поток входных данных и сопоставляет его с заранее выученными последовательностями. Он может найти у себя соответствие и с самого начала последовательности, но, обычно, его работа скорее подобна более гибкой аналогии того, как вы распознаете мелодию с произвольного места. Однако, поскольку регионы НТМ используют распределенные представления паттернов, реализация в них памяти последовательностей и их последующее распознавание существенно сложнее, чем этот простой пример с мелодией, тем не менее, он и дает отличное общее представление о том, как это все работает.

Хотя это может быть и не совсем очевидным, но если у вас уже был некоторый сенсорный опыт восприятия чего-либо, то вы легко найдете похожие на него паттерны во входном потоке. Например, вы вполне сможете узнать слово «завтрак», произнесенное кем угодно, независимо старый это человек или совсем юный, мужчина или женщина, говорит он быстро или медленно или же с жутким акцентом. При этом, даже если один и тот же человек скажет слово «завтрак» сотню раз, звуки этого слова никогда не повторят стимуляцию вашей улитки (аудиального рецептора) в абсолютной точности.

НТМ регион сталкивается с точно такими же проблемами что и ваш мозг: исходные данные никогда точно не повторяются. Следовательно, также как и ваш мозг, регион НТМ должен уметь обрабатывать новые входные данные во время распознавания и обучения. И он может это сделать с помощью использования разреженных пространственных представлений (репрезентаций). Так как, их ключевым свойством является то, что вам нужно сопоставить только некоторую часть паттерна, чтобы быть уверенным в существенности всего совпадения.

Предсказание

Каждый регион НТМ хранит в себе последовательности паттернов. И путем их сопоставления с текущими входными данными, регион формирует предсказание насчет своего вероятного следующего входа. На самом деле регионы НТМ хранят в себе переходы между распределенными пространственными представлениями. В некоторых случаях эти переходы могут выглядеть как линейные последовательности, подобно нотам в мелодии, но в общем случае, в один момент времени, может быть предсказано много вероятных последующих входов. При этом НТМ регион делает различные предсказания, основываясь на текущем контексте событий, который может простираться далеко в прошлое. Основная часть памяти в НТМ отводится именно хранению последовательностей или переходов между пространственными паттернами.

Рассмотрим несколько ключевых качеств НТМ предсказаний.

1) Предсказание непрерывно

Даже не осознавая этого сами, вы постоянно делаете предсказания. И НТМ тоже их делает. Когда вы слушаете мелодию, вы предсказываете каждую следующую ноту. Когда вы идете вниз по лестнице, вы предвидите когда ваша нога коснется следующей ступеньки. Когда вы наблюдаете бросок питчера в бейсболе, вы предвидите, что мяч пройдет близко от биты. Для НТМ региона предсказание и распознавание практически одно и то же. Предсказание это не отдельный шаг в работе, а неотъемлемая ее часть для НТМ региона.

2) Предсказание происходит в каждом регионе на любом уровне иерархии

Если у вас есть иерархия регионов НТМ, предсказания там будут появляться на каждом уровне. Регионы будут делать предсказания насчет изученных ими паттернов. Используя пример с разговорным языком, регионы нижних уровней будут предсказывать возможные следующие фонемы, а регионы высшего уровня могут предсказывать слова или фразы.

3) Предсказания чувствительны к контексту

Предсказания основываются на том, что было в прошлом, как и на том, что происходит сейчас. То есть различный вход будет давать различные предсказания, с учетом предыдущего контекста. Регион НТМ обучается использовать настолько ранний контекст, насколько это нужно для распознавания и может хранить в себе контекст и за короткое и за продолжительное время. Подобная способность известна как «память переменного порядка». Например, вспомним о такой всем известной [в США] речи как Геттисбергское послание.

[Короткая (всего 268 слов в 10 предложениях), но самая знаменитая речь президента Линкольна, которую он произнес 19 ноября 1863 на открытии национального кладбища в Геттисберге. Речь начиналась словами "Восемь десятков и семь лет минуло с того дня, как отцы наши создали на этой земле новую нацию, основанную на идеалах Свободы и свято верящую, что все люди созданы равными ...", а заканчивалась знаменитыми словами о том, что демократия является "властью народа, волей народа, и для народа". Эта речь полностью высечена на пьедестале памятника Линкольну в г. Вашингтоне.]

Чтобы предсказать в этой речи следующее слово, знания только текущего слова явно не достаточно. Например, за словом «и» следует и слово «семь» и слово «свято» уже в самом первом предложении. Но иногда, даже небольшой дополнительный кусочек контекста помогает сделать правильное предсказание на будущее. Услышав «восемь десятков и» уже можно предсказать слово «семь» из этой речи. Кроме того, бывают еще повторяющиеся фразы, и для них требуется гораздо более продолжительный во времени контекст, чтобы понять в каком месте речи вы находитесь и что будет дальше.

4) Предсказания способствуют стабильности

Выходом НТМ региона является его предсказание. И одним из свойств НТМ является то, что этот выход становится более стабильным, т.е. он тем медленнее изменяется, чем выше мы поднимаемся по НТМ иерархии. Это свойство основано на способе предсказания региона. Ведь он не предсказывает только то, что случится непосредственно на следующем шаге. Если он может, он предскажет на несколько шагов вперед. Давайте предположим, что наш регион может предсказывать на пять шагов вперед. При появлении очередного входа, новый предсказываемый шаг изменяется вместе с ним, но четыре шага, уже предсказанные ранее, остаются неизменными. Следовательно, даже если каждый новый вход является новым, на выходе меняется только часть данных, что делает их гораздо более стабильными, чем вход. Эта характеристика отражает уровень наших знаний об окружающем нас сейчас мире, когда концепции верхних уровней, такие как название мелодии, меняются гораздо медленнее, чем концепции нижних уровней, такие как ноты.

5) Предсказание говорит нам, является ли данный вход ожидаемым или нет

Каждый регион НТМ является детектором нового. Поскольку каждый регион предвидит, что должно случиться дальше, он «понимает», когда происходит что-то неожиданное. И регион предсказывает несколько возможных дальнейших вариантов входа, а не только один. Поэтому, хотя он и не может точно предсказать, что случится дальше, но, если следующий вход не соответствует ни одному из предсказаний НТМ региона, он понимает, что произошла аномалия.

6) Предсказания помогают системе быть более устойчивой к шумам

Когда HTM предсказывает, что будет дальше, это предсказание может повлиять на распознавание всей системой того, что было предсказано. Например, если HTM обрабатывает разговорную речь, она будет предсказывать какие звуки, слова и предложения будут произнесены далее. Это предсказание поможет системе заполнить пропущенные данные. Или если поступит непонятный звук, сеть HTM может его интерпретировать согласно тому, что она ожидала, что помогает распознаванию даже при наличии шумов.

Подводя промежуточный итог, можно сказать, что в HTM регионе запоминание последовательностей, распознавание и предсказание являются неразрывно связанными между собой. Все они представляют собой центральные, основные функции HTM региона.

Поведение

Наше мотороное поведение оказывает существенное влияние на то, что мы воспринимаем. Когда мы двигаем глазами, их сетчатка воспринимает совсем другие сенсорные данные. Движения наших конечностей и пальцев изменяют осязательные ощущения, поступающие в наш мозг. Практически все наши действия изменяют то, что мы чувствуем. Сенсорные потоки и наше моторное поведение связаны между собой неразрывным образом.

В течение нескольких десятилетий превалировала точка зрения, что моторные команды образуются в специфической зоне коры – основной моторной области мозга. Но впоследствии было обнаружено, что практически все регионы коры имеют свои моторные исходящие связи, включая даже сенсорные области нижнего уровня иерархии. То есть, похоже, что все они как-то интегрируют в себе и сенсорные, и моторные функции.

Мы полагаем, что подобные исходящие моторные связи могут быть, в принципе, добавлены каждому региону HTM в рамках существующей ныне общей системы, поскольку генерация моторных команд, по сути, во многом подобна выработке предсказаний. Тем не менее, все имплементации HTM, сделанные нами на сегодняшний день, были чисто сенсорными, без каких-либо моторных компонент.

Реализованные имплементации HTM

Мы уже проделали значительный путь по реализации наших теоретических наработок (theoretical framework) по HTM в практическую технологию. Нам удалось реализовать и протестировать несколько версий кортикальных алгоритмов обучения HTM, и мы убедились в работоспособности их базовой архитектуры. По мере продолжения их тестирования на новых наборах данных мы будем совершенствовать эти алгоритмы, и добавлять в них недостающие части. При этом, мы будем регулярно обновлять и дополнять данный документ, по мере нашего продвижения. В следующих трех его главах описывается текущее состояние алгоритмов кортикального обучения HTM.

Тем не менее, существует еще много элементов теории HTM, которые пока не были нами реализованы, включая такие ее области как внимание, обратную связь между регионами, точный тайминг (действия связанные с точными интервалами времени), а также интеграция сенсорно-моторных данных и поведения. Все эти пока пропущенные нами части должны, потом, быть реализованными в рамках уже созданной среды.

Глава 2: Алгоритмы обучения HTM

В этой главе описываются алгоритмы обучения, работающие внутри региона HTM. В главах 3 и 4 описываются имплементации этих алгоритмов обучения с использованием псевдокода, в то время как данная глава в большей степени их основные концепции.

Терминология

До того как мы начнем наше описание, нам будет полезно немного определиться с терминологическим аппаратом. Для описания алгоритмов обучения HTM мы будем использовать термины нейрофизиологии, такие как клетки, синапсы потенциальные синапсы дендритные сегменты и колонки. Это будет логично, поскольку все наши алгоритмы обучения были, в основе своей, выработаны на основе знаний нейрофизиологии с некоторыми теоретическими допущениями. Однако, в процессе реализации наших алгоритмов в виде реальных программ, мы столкнулись с рядом трудностей в плане производительности, и поэтому, как только мы чувствовали, что понимаем, как что-то работает, мы искали способы ускорения обработки данных. Часто это приводило к отходу от строго следования знаниям из биологии, с сохранением требуемого для нас конечного результата. Если вы не очень сильны в нейрофизиологии, то это не составит для вас проблему. Однако, если вы хорошо знакомы с ее терминологией, то вас может несколько смутить то, как мы иногда употребляем здесь ее термины с их новыми значениями. Мы планируем написать отдельную главу, посвященную биологическим основам алгоритмов обучения HTM, но пока что мы только упомянем несколько самых ярких отклонений в терминологии, которые могут вызвать основные проблемы.

Состояние клеток

Клетки HTM (аналоги биологических нейронов) имеют три демонстрируемых состояния: активны от прямого (feed-forward) воздействия, активны от латерального (бокового) воздействия и не активны (пассивны). Первое из этих состояний соответствует короткой пачке импульсов потенциалов действия. Второе – более медленной и размеренной частоте импульсов нейронов. Мы не отметили никакой практической необходимости в более подробном моделировании индивидуальных потенциалов действия и даже различных частот их следования между двумя указанными активными состояниями клеток. Похоже, что использование наших распределенных представлений превосходит все потребности в моделировании наблюдаемых в коре произвольных частот активности клеток.

Дендритные сегменты

В технологии HTM клетки имеют относительно более реалистичную (и следовательно, более сложную) модель дендритов. В теории, каждая клетка HTM имеет один близкий (**проксимальный**) дендритный сегмент и дюжину – другую удаленных (**дистальных**) дендритных сегментов. Близкий дендритный сегмент клетки воспринимает прямое воздействие на нее, а дистальные дендритные сегменты получают латеральные (т.е. «по горизонтали») воздействия от близлежащих клеток. Класс ингибиторных (т.е. подавляющих, тормозящих) нейронов мозга заставляет все нейроны в одной его колонке сходно реагировать на одно и то же прямое воздействие. Чтобы просто сократить вычисления, мы убрали индивидуальные близкие дендритные сегменты от каждой клетки и заменили их на один общий близкий дендритный сегмент для целой колонки нейронов. Функция построения пространственного представления входного патерна (описанная далее) оперирует именно с общими дендритными сегментами на уровне колонок. Но функция темпорального представления работает с дистальными дендритными сегментами, на уровне индивидуальных клеток внутри колонок. Такое упрощение дает нам ту же самую требуемую функциональность, хотя в нейрофизиологии не существует никакого биологического эквивалента

дендритного сегмента общего для целой колонки.

Синапсы

В технологии HTM синапсы клеток имеют только бинарный вес (0 или 1). Реальные, биологические синапсы имеют различный вес, хотя они, при этом, работают стохастично и биологический нейрон никак не может полагаться на точный вес своих синапсов. Использование распределенных представлений в HTM, плюс наша модель дендритных операций позволяет нам обойтись только бинарными весами для синапсов HTM без нежелательных побочных эффектов от такого упрощения. Чтобы смоделировать процесс формирования и исчезновения биологических синапсов нейронов мы использовали две дополнительные концепции из нейрофизиологии, которые могут быть вам не знакомы. Одна из них, это концепция потенциальных синапсов. Согласно ее представлениям, все аксоны, которые проходят достаточно близко к дендритному сегменту, потенциально могут сформировать с ним действующие синапсы. Вторая концепция называется «перманентность» (~ стабильность, устойчивость) синапсов. Это некоторое скалярное (вещественное) значение, изначально присвоенное каждому потенциальному синапсу. Перманентность синапса представляет собой числовое выражение степени связанности между аксоном и дендритом. В биологии она варьируется от полной несвязанности аксона и дендрита, до формирующегося синапса, минимального синапса, и до крупного, полностью связанного синапса. В нашей модели перманентность синапса будет изменяться от 0 до 1. Обучение сети клеток HTM подразумевает под собой увеличение и уменьшение перманентности синапсов. Когда перманентность синапса превышает некоторый порог, то его вес у нас становится равным 1 [синапс становится действующим]. Когда она меньше этого порога, то вес синапса считается равным 0 [синапс просто отключен].

Вводный обзор

Представьте себя на месте региона HTM. На вход к вам поступают тысячи и десятки тысяч бит сигналов. Они могут представлять собой какие-то сенсорные данные, либо же могут приходить от других регионов, стоящих ниже в иерархии обработки сигналов. Входные сигналы содержат в себе нули и единицы в самых замысловатых комбинациях. И что же вам следует сделать со всем этим?

Мы ранее уже обсудили ответ на этот вопрос в самой простой форме. Каждый регион HTM ищет сходные паттерны в своих входных данных и потом запоминает последовательности этих паттернов. Используя свою накопленную память этих последовательностей, каждый регион выдает свои предсказания. Такое высокоуровневое описание работы HTM может показаться вам достаточно простым, но в реальности все довольно сложно. Давайте-ка разделим его на следующие три шага:

- Формирование в регионе HTM пространственного распределенного представления входных данных.
- Формирование представления входных данных в контексте предыдущих входов.
- Формирование предсказания региона HTM на основе текущего входа в контексте предыдущих входов

Далее мы рассмотрим каждый из этих шагов более подробно.

1) Формирование пространственно распределенного представления входа

Входная информация для нашего региона представляет собой огромное количество бит. В мозге это были бы входные аксоны от других нейронов. В любой момент времени некоторые из них были бы активны (равны 1) или не активны (равны 0). И пусть процент активных входных битов варьируется, скажем, от 0 до 60%. Первым делом HTM регион конвертирует свой вход в пространственное разреженное представление. Например, при 40% активных битах на входе региона, в новом его внутреннем представлении может быть всего только около 2% активных битов.

Регион НТМ логически состоит из множества *колонок*. Каждая колонка состоит из одной или более клеток [аналогов биологических нейронов]. Совокупность колонок региона можно представлять себе в виде некоторого двухмерного массива, хотя это и не обязательно. Каждая колонка в нашем регионе воспринимает свою уникальную область входных битов (обычно они перекрываются между собой, но никогда полностью не совпадают для двух разных колонок). В результате, различные паттерны на входе региона приводят к различным уровням активации колонок региона. При этом колонки с более высоким уровнем активации подавляют (деактивируют) [или ингибируют, тормозят] близлежащие колонки с меньшим уровнем активации. Такое подавление происходит в окрестности, радиус которой может изменяться от очень небольшого, до размеров всего региона. Распределенное представление текущих входных сигналов кодируется тем, какие колонки являются активными, а какие нет, после описанной деактивации. Функция такого подавления колонок определяется в соответствующем алгоритме НТМ так, чтобы всегда получать относительно постоянный процент активных колонок в регионе, даже когда процент его активных входных битов изменяется в существенных пределах.

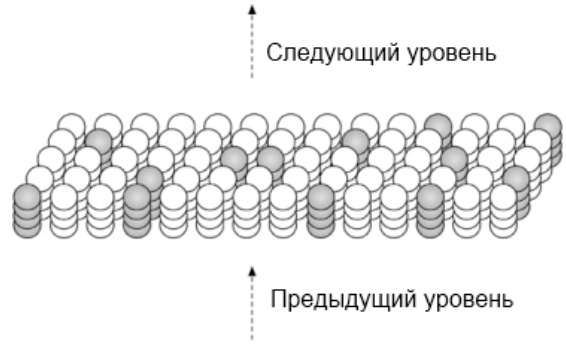


Рисунок 2.1: Регион НТМ состоит из колонок клеток. Показана только небольшая часть региона. Каждая колонка клеток получает на вход активацию от уникальной области входных битов. Колонки с более сильной активацией подавляют колонки с более слабой активацией. В результате, получается разраженное пространственное представление входных данных. (Активные колонки показаны светло-серым цветом).

Представьте себе теперь, что входной паттерн изменился. Если при этом поменялось небольшое число входных битов, то некоторые из колонок будут получать себе на вход лишь немного больше, или немного меньше, активных битов, но итоговое множество активных бит, вероятно, не должно сильно измениться. Таким образом, похожие входные паттерны (те которые имеют существенное множество общих активных битов) будут отображаться в относительно стабильное множество активных колонок. Насколько стабильным будет такое кодирование, сильно зависит от того, к каким входным битам подключена каждая из колонок [синапсами своего близкого (проксимального) дендритного сегмента]. Эти подключения определяются алгоритмом обучения НТМ, который описывается далее.

Все вышеперечисленные шаги (обучение каждой колонки путем подключения к конкретным битам из своего входа, определение уровня активации входом для каждой из колонок, и использование подавления для создания разреженного множества активных колонок) будут нами называться работой «*пространственного группировщика*» (Spatial Pooler) [Один из алгоритмов самообучения НТМ]. Этот термин означает, что результирующие паттерны будут «пространственно похожи» (то есть у них будет много общих активных битов) и таким образом «сгруппированы» (то есть, собраны этим алгоритмом вместе, в общие представления)

2) Формирование представления входа в контексте предыдущих входов

Следующая функция, которую выполняет регион НТМ, конвертирует описанное

колончатое представление текущего состояния входа региона в новое представление, которое включает в себя состояния региона из прошлого (или контекст). Это новое представление формируется путем активации только некоторого подмножества клеток из каждой колонки, при этом, обычно, это будет только одна клетка из всей колонки.

Рассмотрим простой пример. Представьте себе, что вы услышали два предложения: «Я принес косу» и «Я заплел косу». В них одно и то же слово «коса» имеет разное смысловое значение, это омонимы. Можно быть уверенным, что в какой-то момент времени в мозгу есть нейроны, которые реагируют одинаково на приходящие звуки слова «коса». В конце концов, в наше ухо при этом приходят одинаковые звуковые волны. Тем не менее, при этом, несомненно и то, что в мозгу будут и другие нейроны, которые по разному отреагируют на это слово в столь разных контекстах. То есть полное нейронное представление звуков слова «косу» будет разным, когда вы слышите «я принес косу» и «я заплел косу». Представьте себе теперь, что вы выучили и запомнили эти два предложения. Тогда начало «я принес…» приведет вас к другому предсказанию продолжения этого предложения, чем начало «я заплел…». Таким образом, в мозгу олжны существовать разные внутренние представления после того, как вы услышали «я принес…» и «я заплел…».

Этот принцип различного кодирования поступающих данных в различных контекстах является универсальным свойством восприятия и действия [мозга], а также одной из самых важных функций НТМ региона. Ее значение трудно переоценить.

Каждая колонка в регионе НТМ состоит из нескольких клеток. И каждая из клеток может быть активной или пассивной. Выбирая различные активные клетки в каждой активной колонке, мы можем по-разному представить один и тот же вход региона в различных контекстах. Рассмотрим небольшой пример. Пусть в каждой колонке имеется по 4 клетки и представление каждого из входов региона НТМ содержит 100 активных колонок. Даже если в каждый момент времени только одна клетка в колонке может быть активна, у нас имеется 4 в степени 100 способов представить один и тот же вход. То есть, даже если один вход всегда дает одни и те же 100 активных колонок, тем не менее, в различных контекстах, в этих колонках будут активны различные клетки. Таким образом, мы можем представить один вход в очень многих различных контекстах, но насколько уникальными будут эти представления? Легко посчитать, что практически все случайно выбранные пары среди этих 4^100 всевозможных представлений будут перекрываться между собой примерно в 25 клетках. То есть в двух представлениях одного входа в различных контекстах у нас будет порядка 25 общих клеток и около 75 отличающихся клеток, что позволяет легко различить такие представления.

Общее правило для НТМ региона состоит в следующем. Когда колонка становится активной, она активирует в себе все клетки, если данный вход был для нее неожиданным (не предсказанным в ней самой заранее). Но если некоторые из клеток колонки уже будут находиться в состоянии предсказания (которое будет объяснено позже), то активными будут только они, а остальные – нет.

Если у колонки не было никакого предыдущего состояния, и, следовательно, никакого контекста текущей ситуации и соответствующего из него предсказания, то тогда в колонке все клетки становятся активными (если активизируется сама колонка). Такой сценарий напоминает ситуацию, когда вы слышите самую первую ноту мелодии. Без контекста вы, обычно, не сможете предсказать, что прозвучит потом, поскольку возможны практически любые варианты. Аналогично, если у колонки было некоторе предыдущее состояние [соответственно, некоторые контекст и предсказание], но текущий вход не соответствует тому, что ей ожидалось, тогда также все клетки в колонке становятся активными. Все это делается для каждой колонки отдельно, поэтому соответствие в них предсказанию или нет, никогда не бывает событием «все или ни одной».

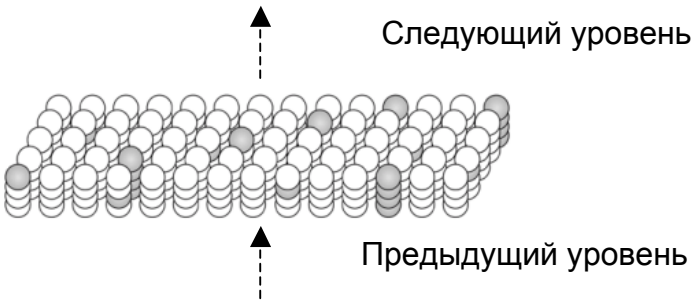


Рисунок 2.2: Путем активации некоторого подмножества клеток в каждой активной колонке, HTM регион может отобразить (представить) один и тот же вход в различных контекстах. Все колонки активируют в себе только предсказывающие клетки. Колонки без таких клеток активируют все свои клетки. На рисунке показано несколько колонок с одной активной клеткой и несколько колонок со всеми активными клетками.

Как уже упоминалось ранее, в терминологическом разделе, клетки HTM могут быть только в одном из трех состояний. Введем следующую терминологию. Если клетка активна благодаря прямому воздействию на нее входных сигналов, то мы ее будем называть просто «активной». Если же клетка стала активна благодаря своим латеральным связям с соседними клетками, мы будем говорить, что она находится в состоянии «предсказания» (или «предчувствия»).

3) Формирование предсказания на основе входа и контекста предыдущих входов

Финальным шагом алгоритма для нашего региона является выработка предсказания, что же должно произойти далее. Это предсказание вырабатывается на основе представления, сформированного на шаге 2), которое включает в себя и контекст от предыдущих входов.

Когда регион создает такое предсказание, он активирует (в состояние предсказания) все клетки, которые вероятно станут активными благодаря последующему прямому воздействию. Поскольку представления в регионе являются разреженными, в одно и то же время может быть сделано несколько предсказаний. Например, если входные данные активируют только около 2% клеток региона, то можно ожидать, что 10 различных предсказаний активируют около 20% клеток в состоянии предсказания. Или 20 различных предсказаний могут дать около 40% колонок с клетками в состоянии предсказания. Если каждая из наших колонок имеет четыре клетки, из которых только одна активна в каждый момент времени, то тогда около 10% всех клеток будут в состоянии предсказания.

В будущей (пока не написанной) главе о разреженных представлениях мы покажем, что даже когда различные предсказания накладываются друг на друга, регион достаточно определенно может узнать, был ли его вход предсказанным или нет.

Так как же регион создает предсказание? Когда входной паттерн изменяется с течением времени, различные наборы колонок и клеток становятся последовательно активными. А когда любая клетка становится активной, то она формирует соединения с подмножеством близких к ней клеток, которые были активны непосредственно до этого. Такие связи могут формироваться быстро или медленно, в зависимости от установленной скорости обучения, требующейся конкретному приложению на основе HTM технологии. После этого, все, что нужно сделать клетке, это отследить по этим сформированным связям совпадающую во времени активность. Если эти соединения вдруг стали активны, то клетка может предположить, что она тоже вскоре может стать активной [как это уже было ранее], и она переходит в состояние предчувствия своей активации. То есть прямое воздействие, ведущее к активации некоторого множества клеток региона, ведет также к предсказательной активации других клеток, которые, вероятно, будут активними далее. Представьте себе это как момент, когда вы распознали слышимую вами мелодию и стали предсказывать, какие ноты будут следующими.

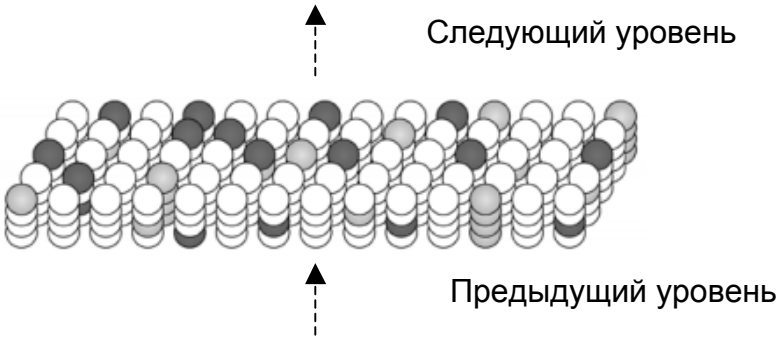


Рисунок 2.3: В любой момент времени некоторые клетки региона НТМ будут активны благодаря прямому воздействию входных сигналов (они показаны светло-серым цветом). При этом другие клетки, получающие латеральные воздействия от активных клеток будут переходить в состоянии предсказания (они показаны темно-серым цветом).

Подводя итог всему вышесказанному, повторим еще раз: когда регион НТМ получает новые входные данные, он формирует по ним разреженное множество активных колонок. Одна или несколько клеток в этих колонках становятся активными, что в свою очередь приводит другие клетки региона в состояние предчувствия активации благодаря выученным ранее связям между клетками в данном регионе. Эти клетки, активированные благодаря своим латеральным связям, представляют собой некоторое предсказание того, что вероятно должно случиться с этим регионом НТМ далее. Когда для него появляется следующий прямой вход, он формирует другое разреженное множество активных колонок. Если такая новая активная колонка не была ранее предсказанна ни одной из ее собственных клеток, то она активирует все свои клетки. А если в новой активной колонке уже находилось несколько клеток в состоянии предсказания, только эти клетки и становятся активными. Итоговым выходом всего региона является активность всех его клеток, включая клетки активные благодаря прямому воздействию входа и клетки активные в состоянии предсказания.

Как нами уже упоминалось ранее, предсказания создаются не только для *следующего* временного шага. Предсказания в НТМ регионе могут быть на несколько шагов в будущем. Используя аналогию с мелодиями, НТМ регион может предсказывать не только следующую ноту, но и четыре следующие ноты в мелодии. Это как раз то, что нам нужно. Ведь тогда суммарный выход региона (совокупность в нем всех активных и предсказывающих клеток) будет изменяться более медленно, чем его вход. Например, представьте себе регион, который предсказывает четыре следующие ноты в мелодии. Для простоты мы представим эту мелодию в виде последовательности букв А, Б, В, Г, Д, Е, Ж. После прослушивания первых двух нот регион узнает эту мелодию и начинает предсказывать ее продолжение в виде В, Г, Д, Е. Клетки для Б уже активны, поэтому клетки для Б, В, Г, Д, Е находятся в одном из двух активных состояний. Теперь регион слышит следующую ноту В. Теперь множество активных и предсказывающих клеток представляет собой множество В, Г, Д, Е, Ж. Обратите внимание, что входной паттерн полностью изменился с Б на В, а на выходе региона изменилось только около 20% клеток.

Поскольку выходом НТМ региона является совокупный вектор, представляющий собой активность всех клеток региона, он будет в пять раз более стабилен в приведенном нами примере, чем вход. При организации регионов в некоторую иерархию, мы будем наблюдать в ней все возрастающую стабильность при продвижении к ее вершине.

Мы будем использовать термин «темпоральный группировщик» для описания этих двух шагов по добавлению контекста в наше представление и предсказание. [Второй алгоритм самообучения НТМ.] Поскольку при этом получается медленно изменяющийся выход региона, мы действительно группируем вместе различные входные паттерны, которые следуют друг за другом с течением времени.

Теперь настала пора нам опуститься в нашем рассмотрении алгоритмов НТМ на

следующий уровень детализации. Мы начнем его с некоторых общих концепций для пространственного и темпорального группировщиков. Потом мы обсудим особенности уникальные для пространственного группировщика, а затем понятия уникальные для темпорального группировщика.

Общие концепции

Обучение и в пространственном и в темпоральном группировщике происходит сходным образом. И там и там при этом происходит установление связей, или синапсов между клетками. Однако, в темпоральном группировщике устанавливаются «горизонтальные» связи между клетками из одного региона. А в пространственном группировщике обучаются связи от прямого воздействия входных бит к колонкам региона.

Бинарные веса

В технологии HTM все синапсы имеют только коэффициенты 0 и 1, то есть их «веса» бинарны, что резко отличается от большинства других моделей нейронных сетей, где веса синапсов являются вещественным числом от 0 до 1.

Перманентность синапсов

Синапсы постоянно создаются и расформировываются во время обучения. Мы присвоили каждому синапсу некоторое вещественное число (от 0.0 до 1.0) для обозначения насколько перманентно (постоянно) его соединение. Когда любое такое соединение в очередной раз задействуется, его перманентность возрастает. Во всех прочих случаях перманентность уменьшается. Если перманентность синапса в итоге падает ниже некоторого предела, данный синапс полностью отключается [его вес становится 0].

Дендритные сегменты

Все синапсы подключаются только к дендритным сегментам клеток, которых существует всего два типа:

- Один тип дендритных сегментов клеток формирует синапсы с входными битами. Активность синапсов из такого сегмента линейно суммируется для определения прямой активации всей колонки.
- Другой тип дендритных сегментов формирует синапсы только с клетками внутри региона. Каждая клетка имеет несколько дендритных сегментов такого типа. Если сумма активных синапсов такого сегмента превышает некоторый предел, тогда соответствующая клетка становится активной в состоянии предсказания. И поскольку таких дендритных сегментов много у каждой клетки, то в ее функции выработки состояния предсказания применяется логическое ИЛИ для нескольких составляющих пороговых детекторов [т.е. дендритных сегментов данного типа].

Потенциальные синапсы

Концепция потенциальных синапсов клеток также позаимствована нами из нейрофизиологии. Она основана на том факте, что далеко не все соединения между клетками физически возможны. Следовательно, каждый дендритный сегмент имеет свой список потенциальных синапсов. У каждого потенциального синапса HTM есть свое значение величины перманентности, и он может, таким образом, превратиться в функционирующий синапс, если его величина перманентности превысит определенный порог.

Обучение

Обучение представляет собой увеличение или уменьшение величин перманентности потенциальных синапсов в дендритных сегментах клеток региона HTM. При этом правило, которое используется для превращения синапса в более

или менее перманентный, весьма похожи на «Хэбовские» правила обучения синапсов. Например, если постсинаптическая клетка активна благодаря какому-то дендритному сегменту, получившему суммарный вход выше своего порога, тогда значения перманентности всех синапсов этого сегмента изменяются. Синапсы, которые были активны, и, следовательно, внесли свой вклад в активность клетки, увеличивают свои перманентности. А синапсы, которые были не активны, соответственно, уменьшают значение своей перманентности. Однако, условия, при которых значения перманентности у синапсов изменяются, различны в пространственном и темпоральном группировщике. Подробнее они будут описаны далее.

Теперь мы обсудим отдельно концепции специфические для работы пространственного и темпорального группировщиков.

Концепции пространственного группировщика

Самой основной функцией пространственного группировщика является преобразование входа региона HTM в его разреженный паттерн. Это очень важно, поскольку механизмы, используемые в запоминании последовательностей и выработке предсказаний активно используют свойства пространственных разреженных паттернов HTM.

Существует несколько взаимоперекрывающихся задач и целей работы пространственного группировщика, которые, в конечном итоге, и определяют как он работает и (само)обучается.

1) Использовать все колонки

В HTM регионе существует только фиксированное число колонок, которые обучаются отображать повторяющиеся паттерны входных данных региона. И одной из наших целей будет, чтобы все из имеющихся колонок обучились представлять что-то полезное не зависимо от того, сколько их имеется в наличии. Нам не нужны колонки, которые никогда не бывают активными. И чтобы предотвратить это, мы будем отслеживать, как часто каждая колонка бывает активной относительно своих соседей. И если ее относительная активность будет падать слишком низко, то эта колонка будет сама увеличивать уровень своей входной чувствительности, пока не войдет в число колонок – «победителей» в конкуренции за активацию. Иначе говоря, все колонки участвуют в некотором «соревновании» со своими соседями, чтобы поучаствовать в представлении входных паттернов. И если какая-то колонка редко бывает активной, то она становится более агрессивной в этом плане. Как следствие, другие колонки будут вынуждены изменить свои входы и начать представлять немного другие входные паттерны.

2) Поддержание желательной плотности

Региону HTM нужно формировать именно *разреженные* пространственные представления своих входных данных. Поэтому, самые активные от прямых входных данных колонки региона HTM просто подавляют своих соседей. Существует некоторый радиус окрестности такого подавления, который пропорционален размерам рецепторного поля колонок (и, следовательно, может изменяться от малых областей до размера всего региона). Внутри этого радиуса подавления мы позволяем только малому проценту самых активных по входу колонок быть «победителями». Остальные колонки будут пассивными. (Такой «радиус» подавления подразумевает некоторую организацию колонок на плоскости, но данная концепция может быть адаптирована и для других пространственных топологий.)

3) Избегание тривиальных паттернов

Мы хотим, чтобы все наши колонки представляли не тривиальные входные паттерны. Эта цель может быть достигнута установкой некоторого минимального порога активации колонки от ее входных данных. Например, если мы установим такой порог равным 50, то тогда колонка должна иметь минимум 50 активных

синапсов в своем входном дендритном сегменте, чтобы стать активной. Это гарантирует определенный уровень сложности представляемого входного паттерна.

4) Подавление излишних связей

Если мы не будем осторожны, колонка сможет сформировать громадное число действующих синапсов. После этого она будет сильно реагировать на множество различных, не относящихся друг к другу, входных паттернов. Просто различные подмножества ее синапсов будут давать достаточную реакцию на самые разные паттерны. Чтобы избежать такой проблемы, мы будем уменьшать значение перманентности у каждого синапса, который не был вовлечен в текущую активность колонки-победительницы. Если такие молчащие синапсы будут подвергаться заметному подавлению, то мы можем гарантировать, что колонка будет участвовать в представлении ограниченного числа входных паттернов, вплоть до только одного.

5) Самонастраиваемые рецептивные поля

Реальный мозг крайне «пластичен», то есть области его коры могут перенастраиваться на представление совершенно других вещей, реагируя на произошедшие изменения. Если какая-то часть коры головного мозга была повреждена, другие ее части постараются приспособиться и обрабатывать в себе то, что до этого обрабатывала указанная поврежденная часть. И если некоторый сенсорный орган был поврежден либо сильно изменился, то соответствующая ему область коры перенастроится, чтобы обрабатывать в себе что-то другое. Такая система является самонастраиваемой и самоадаптируемой.

И мы хотим, чтобы наши HTM регионы демонстрировали подобную же гибкость. Если мы поместим 10 000 колонок в HTM регион, он должен обучиться, как наилучшим образом представлять свои входные данные с помощью этих 10 000 колонок. Если мы поместим туда 20 000 колонок, он должен обучиться, как лучше всего использовать и это их число. Если статистика входных данных изменится, колонки также должны измениться, чтобы наилучшим образом отражать в себе новую реальность. Корче говоря, дизайнер HTM должен иметь возможность поместить любое желаемое число ресурсов в регион HTM и последний должен наилучшим образом представлять в себе входные данные, основываясь на имеющихся колонках и статистике входа. Общее правило таково, что чем больше колонок есть в регионе, тем более крупные и более детальные входные паттерны будет представлять каждая из колонок. При этом, скорее всего, каждая из колонок будет активной несколько реже, хотя мы сохраним относительно постоянный уровень разреженности представлений.

Теперь нам нужны соответствующие правила обучения, которые помогут нам достичь обозначенные цели и задачи. Стимулируя неактивные колонки; подавляя соседние колонки, для поддержки постоянной разреженности представлений; устанавливая минимальный порог для входных данных; поддерживая большой набор потенциальных синапсов; и увеличивая или забывая синапсы, в зависимости от их вклада в стимуляцию, наши ансамбли колонок будут динамически самоконфигурироваться, чтобы достигать желаемого эффекта.

Детали реализации пространственного группировщика

Теперь мы можем пройтись по списку всех функций, которые выполняет пространственный группировщик.

Все начинается с фиксированного числа входных битов региона HTM. Они могут представлять собой некоторые сенсорные данные или могут приходить от другого региона, нижележащего в иерархии.

Фиксированное число колонок данного региона будет получать эти входные биты. У каждой колонки имеется свой дендритный сегмент. Каждый дендритный сегмент имеет набор потенциальных синапсов, связанных с некоторым подмножество входных битов. У каждого такого потенциального синапса есть его значение величины перманентности. На основе этих значений, некоторые из потенциальных синапсов будут действующими, а некоторые - нет.

Для любого набора входных данных подсчитывается сколько действующих синапсов у каждой колонки подключено к активным входным битам.

Число активных синапсов умножается на фактор «ускорения» («агрессивности») данной колонки, который динамически определяется из того, насколько часто данная колонка бывает активной относительно своих соседей.

Колонка с максимальным итоговым входом подавляет (ингибирует) все ближайшие к ней колонки (за исключением определенного небольшого процента) в своем радиусе подавления. Этот радиус, в свою очередь, динамически определяется числом входных битов (или «входных проводов») колонок. В результате получается разреженное множество активных колонок.

Для каждой активной колонки мы подстраиваем значения перманентности для всех ее потенциальных синапсов. Значения перманентности для синапсов подключенных к активным входным битам увеличивается, а подключенных к пассивным входным битам – уменьшается.

Концепции работы темпорального группировщика

Напомним, что темпоральный группировщик запоминает последовательности событий и вырабатывает предсказания. Основной принцип здесь состоит в том, что когда клетка становится активной, она создает связи с другими клетками региона, которые были активными только что. После этого, клетки могут предсказывать наступление момента своей активности, просто отслеживая эти свои связи. Если так будут поступать все клетки, то все вместе, коллективно, они могут запоминать и вспоминать последовательности, а также предсказывать, что, скорее всего, произойдет дальше. При этом не существует никакого центрального хранилища последовательностей паттернов. Вместо этого, память распределена среди многих отдельных клеток. И благодаря этому, все система получается устойчивой к шуму и к ошибкам. Отдельные клетки могут вообще иногда не срабатывать с очень небольшим, или вообще без всякого отрицательного эффекта для всей системы.

Это одни из самых важных свойств разреженных представлений, которые использует темпоральный группировщик.

Предположим, что у нас есть гипотетический регион НТМ, который всегда формирует представления своих входных паттернов используя только 200 своих клеток из 10 000 (2% активных клеток в любой момент времени). Как мы можем при этом запомнить и распознать какой-то его определенный паттерн из 200 клеток? Простейший способ, это составить список из этих 200 клеток, за которыми мы будем следить. И если мы увидим, что все 200 из них опять стали активными, мы распознаем наш паттерн. Но что будет, если мы составим список только из 20 клеток и проигнорируем остальные 180? Вам может показаться, что наблюдение только за 20 клетками из 200 приведет нас к множеству ошибок, поскольку они могут быть активными во множестве других паттернов из 200 клеток. Но, на самом деле, это не так. Поскольку наши паттерны очень разреженные и разбросанные по всему региону (в нашем случае 200 клеток из 10 000), отслеживание только 20 клеток дает примерно те же самые результаты, как и запоминание всех 200 клеток. Вероятность ошибки в подобных системах очень мала.

Клетки в НТМ регионе используют это его свойство. Дендритные сегменты каждой клетки имеет множество связей с другими клетками региона. И дендритные сегменты формируют эти связи с целью распознания определенного состояния сети клеток в определенный момент времени. Поблизости могут быть сотни или тысячи активных клеток, но дендритный сегмент должен сформировать связи только с 15 или 20 из них. И когда дендритный сегмент обнаружит, что эти 15 клеток стали активными, он может быть практически уверен, что в сети повторился большой паттерн. Такая техника называется «подвыборка» (sub-sampling) и она будет использоваться во многих алгоритмах НТМ.

Каждая наша клетка участвует во многих различных разреженных паттернах и во многих различных последовательностях. Конкретная клетка может участвовать в десятках и сотнях временных переходов. Поэтому, каждая клетка имеет не один, а несколько дендритных сегментов. В идеале у клетки было бы по одному дендритному сегменту для каждого паттерна активности, который она хотела бы

распознавать. Однако на практике, один дендритный сегмент может создать связи для запоминания нескольких различных паттернов, и все равно хорошо работать. Например, один сегмент может запомнить по 20 связей для каждого из 4 различных паттернов, что в итоге составит 80 связей. Пусть мы установим порог так, что дендритный сегмент становится активным, когда любые из его 15 связей становятся активными. В этом случае появляется возможность ошибки, если связи для различных паттернов не корректно скомбинируются между собой и совместно превысят пороговое значение. Тем не менее, такой тип ошибки довольно маловероятен, опять же, благодаря разреженности наших представлений.

Теперь мы можем себе представить, как одна клетка с одним или парой дюжин дендритных сегментов и несколькими тысячами синапсов может распознать сотни различных паттернов активности клеток.

Детали реализации темпорального группировщика

Здесь мы еще раз перечислим все шаги работы темпорального группировщика. Мы начнем с того места, где закончил свою работу пространственный группировщик, оставив множество активных колонок отображающих входные данные региона HTM.

Для каждой активной колонки проверяем в ней все клетки, находящиеся в состоянии предсказания и активируем их. Если таких клеток нет, то активизируем все клетки в этой колонке. Полученное множество активных клеток является представлением текущих входных данных в контексте предыдущих входов.

Для каждого дендритного сегмента каждой клетки региона, подсчитываем, сколько действующих синапсов подключены к другим активным клеткам. Если их число превышает установленное пороговое значение, то такой дендритный сегмент считается активным. Клетки с активными дендритными сегментами переводятся в состояние предсказания (или предчувствия), если только они уже не активны благодаря прямому воздействию входных данных. Клетки без активных дендритных сегментов и неактивные от входных данных снизу становятся (или остаются) неактивными. Теперь совокупность клеток в состоянии предсказания (предчувствия) представляет собой итоговое предсказание нашего региона.

Для всех активных дендритных сегментов модифицируются значения перманентности для всех синапсов, ассоциированных с этим сегментом. Для каждого потенциального синапса активного дендритного сегмента мы увеличиваем его значение перманентности, если он подключен к активной клетке, а если он подключен к пассивной клетке, то мы уменьшаем его перманентность. Все эти изменения в перманентности синапсов помечаются как временные.

Так модифицируются синапсы сегмента, который уже обучен активироваться и, соответственно, вырабатывать предсказание. Однако, нам хотелось бы расширить наши предсказания во времени, если это возможно. Поэтому, мы выбираем другой дендритный сегмент у той же самой клетки для обучения. Мы подбираем для этого тот сегмент, который более всего соответствовал состоянию системы на предыдущем временном шаге. Для этого сегмента, используя это состояние предыдущего шага, мы увеличиваем перманентность тех синапсов, которые подключены к активным клеткам, и уменьшаем у тех, которые подключены к пассивным клеткам. Такие изменения в перманентности синапсов также помечаются как временные.

Если клетка переключается из пассивного состояния благодаря прямому воздействию входных данных, то мы проходим по всем потенциальным синапсам, ассоциированным с этой клеткой, и удаляем у них все пометки временности изменений. Таким образом, мы обновляем значения перманентности синапсов, только если они корректно предсказали активацию клетки по входным данным.

Когда клетка переключается из любого активного состояния в пассивное, мы отменяем все изменения в перманентности синапсов, помеченные как временные, для всех потенциальных синапсов этой клетки. Поскольку мы не хотим усиливать перманентность синапсов, которые неправильно предсказали активацию клетки от входных данных.

Обратите внимание, что только те клетки, которые стали активными благодаря

прямому воздействию входа могут распространять дальнейшую активность *внутри* региона НТМ. Иначе его предсказания приводили бы нас к другим предсказаниям и т.д. Но при этом все активные клетки обоих типов (и от входных данных и в состоянии предсказания) создают совместно выход данного региона, который передается *следующему* региону в иерархии.

Последовательности, предсказания первого порядка и предсказания изменяемого порядка

Нам нужно обсудить еще одну тему, касающуюся пространственного и темпорального группировщиков. Хотя она может быть и не очень интересна для всех читателей, и не очень нужна для понимания последующих глав 3 и 4.

Что случится, если в колонках будет немного больше или немного меньше клеток? например, что будет, если в наших колонках будет только одна клетка? В примере, который мы использовали ранее, мы показали, что представление состоящее из 100 активных колонок с 4-я клетками на колонку может быть закодировано 4 в степени 100 различными способами, что дает нам очень большое число. Таким образом, один и тот же вход может быть использован во множестве различных контекстов. Если, например, каждый входной паттерн будет представлять у нас некоторое слово, то весь наш регион НТМ легко сможет запомнить большое множество предложений, которые используют одни и те же слова. Или же мы можем использовать то же самое количество слов и выучить меньшее число, но зато более длинные, предложения. Такой тип памяти называется памятью «переменного порядка», что означает, что ей нужно переменное количество предыдущего контекста, чтобы предсказать следующий вход. НТМ регион является именно такой памятью переменного порядка.

Если мы увеличим количество клеток до пяти на одну колонку, доступное нам число представлений для каждого входа в нашем примере возрастет до 5^100, что гораздо больше чем 4^100. Однако оба эти числа настолько велики, что для большинства практических приложений такое увеличение емкости памяти просто не будет использовано.

При этом, существенное уменьшение числа клеток в колонках дает большую разницу.

Если мы, например, оставим только одну клетку в колонках региона НТМ, то мы потеряем возможность включать прошлый контекст в наши представления. Любые входные данные для такого региона будут приводить к одному и тому же предсказанию, независимо от предыдущих событий. С одной клеткой на колонку, память НТМ региона становится памятью «первого порядка»; т.е. ее предсказания зависят только от текущего входа.

Тем не менее, предсказания первого порядка идеально подходят для решения одного из типов проблем, которые постоянно решает наш мозг: статического пространственного распознавания (inference). Как нами уже упоминалось ранее, быстрый показ человеку визуального образа позволяет ему успешно распознать его, даже если время показа настолько мало, что человек не успевает сделать движение глазами (саккаду). Когда вы слушаете, то вам всегда нужно слышать некоторую последовательность паттернов звуков, чтобы распознать, что же это такое. Со зрением это обычно тоже так: вы воспринимаете именно поток визуальных образов. Но в особых случаях вы можете распознать и единичный образ.

Вам может показаться, что темпоральное и статическое распознавание требуют различных механизмов для своего осуществления. Одно требует распознавания последовательностей паттернов и делает предсказания, основываясь на переменной глубине временного контекста. Другое требует распознавания статических пространственных паттернов без использования временного контекста. Тем не менее, регион НТМ со многими клетками в каждой колонке идеально приспособлен для распознавания временных последовательностей, а регион НТМ только с одной клеткой в колоноках идеально подходит для распознавания пространственных паттернов. В компании Numenta мы провели множество экспериментов, используя регионы с одноклеточными колонками, применительно к решению различных визуальных задач. И хотя полное описание всех этих экспериментов выходит за рамки данной главы, мы остановимся здесь

на их самых важных концепциях.

Если мы продемонстрируем HTM региону ряд картинок, то колонки в регионе обучатся представлять общие пространственные взаиморасположения пикселей. Типы этих выученных паттернов будут напоминать те, что мы наблюдаем в первичной визуальной зоне V1 коры головного мозга (которая достаточно подробно изучена в нейрофизиологии). Типичными примерами тут будут линии и углы в различных ориентациях. При обучении на движущихся образах, HTM регион запоминает перемещения этих базовых фигур. Например, за вертикальной линией в одной позиции часто следует также вертикальная линия, только сдвинутая влево или вправо. Все такие часто наблюдаемые перемещения паттернов запоминаются регионом HTM.

Что же произойдет, если мы покажем такому региону вертикальную линию, движущуюся вправо? Если в нашем регионе есть только колонки с одной клеткой, он будет предсказывать, что линия далее может сдвинуться влево или вправо. Он не сможет использовать контекст знания, где была линия в прошлом и, следовательно, понимать движется ли она влево или вправо. Получается, что такие одноклеточные колонки ведут себя подобно «простым клеткам» коры мозга. Прогноз от таких клеток будет активен для линии в различных позициях не зависимо от того, движется ли она вправо или влево или вообще не движется. Кроме того, мы обнаружили, что подобные регионы демонстрируют свойства стабильности к перемещениям, изменениям масштаба и т.д., сохраняя при этом возможность различать разные образы. Такое поведение бывает просто необходимо для пространственной инвариантности (распознавание одного паттерна в различных местах одного образа).

Если мы теперь проделаем те же самые эксперименты с регионом HTM, в колонках которого имеется множество клеток, мы можем обнаружить, что последние ведут себя подобно «комплексным клеткам» коры мозга. Их предсказания будут активны только для случаев линии, движущейся либо влево, либо вправо, но не оба сразу.

Собрав вместе все эти наблюдения, мы выдвинули следующую гипотезу. В коре головного мозга должны выполняться предсказание с распознаванием как первого так и переменного порядка. В каждом регионе коры мозга имеется четыре – пять клеточных слоев. Они различаются по некоторым параметрам, но все они сохраняют свойство общеколоночной реакции и имеют множество горизонтальных связей внутри своего слоя. Мы предполагаем, что каждый уровень клеток в регионе коры мозга выполняет аналог правил распознавания и обучения HTM, описанных в этой главе. Различные уровни клеток играют в этом различные роли. Например, известно из анатомических исследований, что слой 6 создает обратную связь в иерархии зон мозга, а уровень 5 вовлечен в моторное поведение. Два основных клеточных слоя восприятия входных данных это слои 4 и 3. Мы предполагаем, что одно из отличий между слоями 4 и 3 состоит в том, что клетки в слое 4 работают независимо, как одна клетка в колонке, а клетки в слое 3 работают подобно многим клеткам в колонке. Так зоны коры мозга вблизи областей ввода сенсорных данных имеют оба вида памяти и первого порядка и переменного порядка. Память последовательностей первого порядка (ориентировочно соответствующая 4 слою нейронов) весьма полезна для формирования репрезентаций, которые инвариантны к пространственным изменениям. А память последовательностей переменного порядка (ориентировочно соответствующая нейронам 3-го уровня) полезна для выводов и предсказаний о движущихся образах.

Суммируя вышесказанное, мы выдвигаем гипотезу, что на всех нейронных уровнях коры головного мозга работают алгоритмы, похожие на те, что описаны в данной главе. Эти уровни различаются между собой во многих аспектах, что заставляет предположить их различную роль в обеспечении восприятия входных данных, создания обратной связи, внимания и моторного поведения. А в зонах коры, близких к областям ввода сенсорных данных, очень полезно иметь уровень нейронов выполняющих функции памяти первого порядка, поскольку это ведет к пространственной инвариантности получаемых представлений.

В компании Numenta мы проделали много экспериментов с HTM регионами первого порядка (одна клетка на колонку) в задачах на распознавание образов. Мы также экспериментировали с HTM регионами переменного порядка

(множество клеток на колонку) в задачах распознавания и предсказания последовательностей различного порядка. На будущее будет логично попытаться их скомбинировать в одном регионе и расширить его алгоритмы на достижение и других целей. Тем не менее, уже сейчас мы убеждены, что множество интересных проблем могут быть разрешены с помощью эквивалентов одноуровневых или многоуровневых регионов, либо отдельных, либо собранных в иерархию.

Глава 3. Имплементация пространственного группировщика и ее псевдокод

Данная глава содержит детализированный псевдокод базовой реализации алгоритма пространственного группировщика. Входными данными для данного кода является массив входных битов, приходящих от сенсоров или от предыдущего уровня. В результате своей работы данный код вычисляет **activeColumns(t)** – список колонок, которые «побеждают» в конкуренции за активацию от прямых входных данных снизу, в момент времени **t**. Потом этот список посылается на вход темпоральному группировщику, который описан в следующей главе (т.е. **activeColumns(t)** является выходом для пространственного группировщика).

Нижеследующий псевдокод разбит на три разные фазы, которые выполняются одна за другой:

Фаза 1: вычисление значения перекрытия с текущим входом для каждой колонки.

Фаза 2: вычисление побеждающих колонок после подавления.

Фаза 3: обновление перманентности синапсов и внутренних переменных.

Хотя пространственный группировщик без проблем самообучается прямо во время своей работы, вы всегда можете выключить это самообучении просто исключив Фазу 3.

Далее в этой главе приводится псевдокод для каждой из этих трех фаз. Различные структуры данных и вспомогательные процедуры, используемые в псевдокоде, описаны в конце этой главы.

Инициализация

Еще до того как получить любые входные данные, регион должен быть проинициализирован, а для этого надо создать начальный список потенциальных синапсов для каждой колонки. Он будет состоять из случайного множества входных битов, выбранных из пространства входных данных. Каждый входной бит будет представлен синапсом с некоторым случайным значением перманентности. Эти значения выбираются по двум критериям. Во-первых, эти случайные значения должны быть из малого диапазона около **connectedPerm** (пороговое значение – минимальное значение перманентности при котором синапс считается «действующим» («подключенным»)). Это позволит потенциальным синапсам стать подключенными (или отключенными) после небольшого числа обучающих итераций. Во-вторых, у каждой колонки есть геометрический центр ее входного региона и значения перманентности должны увеличиваться по направлению к этому центру (т.е. у центра колонки значения перманентности ее синапсов должны быть выше).

Фаза 1: Перекрытие (Overlap)

Первая фаза вычисляет значение перекрытия каждой колонки с заданным входным вектором (данными). Перекрытие для каждой колонки это просто число действующих синапсов подключенных к активным входным битам, умноженное на фактор ускорения («агрессивности») колонки. Если полученное число будет меньше **minOverlap**, то мы устанавливаем значение перекрытия в ноль.

```
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)
```

Фаза 2: Ингибирование (подавление)

На второй фазе вычисляется какие из колонок остаются победителями после применения взаимного подавления. Параметр **desiredLocalActivity** контролирует число колонок, которые останутся победителями. Например, если **desiredLocalActivity** равен 10, то колонка останется победителем если ее значение перекрытия выше чем значения перекрытия 10 самых лучших колонок в ее радиусе подавления (ингибирования).

```
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.
```

Фаза 3: Обучение

На третьей фазе происходит обучение. Здесь обновляются значения перманентности всех синапсов, если это необходимо, равно как и фактор ускорения («агрессивности») колонки вместе с ее радиусом подавления.

Основное правило обучения имплементировано в строках 20-26. Для победивших колонок, если их синапс был активен, его значение перманентности увеличивается, а иначе – уменьшается. Значения перманентности ограничены промежутком от 0.0 до 1.0 .

В строках 28- 36 имплементирован механизм ускорения. Имеется два различных механизма ускорения помогающих колонке обучать свои соединения (связи). Если колонка не побеждает достаточно долго (что измеряется в **activeDutyCycle**), то увеличивается ее общий фактор ускорения (строки 30-32). Альтернативно, если подключенные синапсы колонки плохо перекрываются с любыми входными данными достаточно долго (что измеряется в **overlapDutyCycle**), увеличиваются их значения перманентности (строки 34-36). Обратите внимание: если обучение выключено, то **boost(c)** замораживается.

И наконец, в конце фазы 3 обновляется радиус подавления колонки (строка 38).

```
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.
```

Внутренние структуры данных и процедуры

Следующие переменные и структуры данных используются в нашем псевдокоде:

columns	Список всех колонок.
input(t,j)	Вход для данного уровня в момент времени **t**. input(t, j) = 1 если j-ый бит входа = 1.
overlap(c)	Значение перекрытия для колонки **c** в пространственном группировщике для данного конкретного входа.
activeColumns(t)	Список индексов колонок – победителей благодаря прямым входным данным.
desiredLocalActivity	Параметр контролирующий число колонок победителей после шага подавления.
inhibitionRadius	Средний размер входного (рецепторного) поля колонок.
neighbors(c)	Список колонок находящихся в радиусе подавления **inhibitionRadius** колонки **c**.
minOverlap	Минимальное число активных входов колонки для ее участия в шаге подавления.
boost(c)	Значение ускорения («агрессивности») для колонки **c** вычисленное во время обучения – используется для увеличения значения перекрытия для малоактивных колонок.
synapse	Структура данных представляющая собой синапс. Содержит в себе значение перманентности синапса и индекс его входного бита.
connectedPerm	Если значение перманентности синапса больше данного

параметра, то он считается подключенным (действующим).

potentialSynapses(c) Список потенциальных синапсов и их значений перманентности.

connectedSynapses(c) Подмножество потенциальных синапсов **potentialSynapses(c)** у которых значение перманентности больше чем **connectedPerm**. То есть это прямые входные биты, которые подключены к колонке **c**.

permanenceInc Количество значений перманентности синапсов, которые были увеличены при обучении.

permanenceDec Количество значений перманентности синапсов, которые были уменьшены при обучении.

activeDutyCycle(c) Интервальное среднее показывающее как часто колонка с была активна после подавления (то есть за последние 1000 итераций).

overlapDutyCycle(c) Интервальное среднее показывающее как часто колонка с имела существенное значение перекрытия (т.е. большее чем **minOverlap**) со своим входом (то есть за последние 1000 итераций).

minDutyCycle(c) Переменная представляющая минимальную желательную частоту активации (firing) для клетки. Если эта частота клетки упадет ниже данного значения, то она будет ускорена (boosted). Это значение определяется как 1% от максимальной частоты активации соседей клетки.

Следующие внутренние процедуры используются в вышеприведенном коде.

kthScore(cols, k)

Для заданного списка колонок возвращает их **k**-ое максимальное значение их перекрытий со входом.

updateActiveDutyCycle(c)

Вычисляет интервальное среднее того, как часто колонка **c** была активной после подавления.

updateOverlapDutyCycle(c)

Вычисляет интервальное среднее того, как часто колонка **c** имела значение перекрытия со входом большее, чем **minOverlap**.

averageReceptiveFieldSize()

Средний радиус подключенных рецептивных полей всех колонок. Размер подключенного рецептивного поля колонки определяется только по подключенным синапсам (у которых значение перманентности >= **connectedPerm**). Используется для определения протяженности латерального подавления между колонками.

maxDutyCycle(cols)

Возвращает максимальное число циклов активности для всех заданных колонок.

increasePermanences(c, s)

Увеличивает значение перманентности всех синапсов колонки **c** на коэффициент умножения **s**.

boostFunction(c)

Возвращает значение ускорения колонки **c**. Это вещественное значение >= 1. Если **activeDutyCyle(c)** больше **minDutyCycle(c)**, то значение ускорения = 1. Ускорение начинает линейно увеличиваться как только **activeDutyCyle** колонки падает ниже **minDutyCycle**.

Глава 4. Имплементация темпорального группировщика и ее псевдокод

В данной главе содержится детальный псевдокод базовой реализации алгоритма темпорального группировщика. Входом для него является результат вычислений **activeColumns(t)**, полученный в результате работы пространственного группировщика. Данный код вычисляет состояния активности и предсказания для всех клеток региона HTM в данный момент времени **t**. Логическая операция ИЛИ для указанных состояний по всем клеткам формирует вектор результата работы темпорального группировщика для передачи его на вход следующему уровню HTM.

Данный псевдокод разбивается на три отдельные фазы, которые выполняются последовательно:

Фаза 1: вычисление активных состояний **activeState(t)** для каждой клетки.

Фаза 2: вычисление состояний предчувствия (предсказания) **predictiveState(t)** для каждой клетки.

Фаза 3: обновление синапсов.

Фаза 3 необходима только для обучения. Тем не менее, в отличие от работы пространственного группировщика, здесь фазы 1 и 2 включают в себя несколько операций, специфических именно для обучения, если оно задействовано. Поскольку темпоральный группировщик гораздо более сложен в своей работе, чем пространственный, мы рассмотрим сначала его версию, предназначенную только для распознавания, а потом версию, где будут и распознавание и обучение. Описание некоторых деталей данной реализации, терминологии и вспомогательных процедур приводится в конце этой главы после обсуждения псевдокода.

Псевдокод темпорального группировщика: только распознавание

Фаза 1

На первой фазе вычисляются активные состояния для каждой клетки. То есть для каждой победившей активной колонки мы должны определить, какие из ее клеток должны стать активными. Если прямой вход был предсказан какой-либо из ее клеток (т.е. ее параметр **predictiveState** был равен 1 благодаря ее сегменту последовательностей (латеральному), на предыдущем временном шаге), тогда такие клетки становятся активными (строки 4-9). А если данный прямой вход был неожиданным для колонки (т.е. в ней не было клеток с **predictiveState** равным 1), тогда каждая клетка в этой колонке становится активной (строки 11-13).

```
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
```

Фаза 2

Вторая фаза вычисляет состояния предсказания (предчувствия активации) для каждой клетки. Каждая клетка включает свое состояние предчувствия (параметр **predictiveState** становится 1) если хоть один из ее латеральных сегментов становится активным, т.е. достаточное число его горизонтальных (боковых, латеральных) соединений сейчас активно благодаря прямому входу.

```
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
```

Псевдокод темпорального группировщика: комбинация распознавания и обучения

Фаза 1

На первой фазе вычисляются активные состояния (значения **activeState**) для каждой клетки из победивших колонок. Из этих колонок далее выбирается одна клетка на колонку для обучения (**learnState**). Логика здесь следующая: если текущий прямой вход снизу был предсказан какой-либо из клеток (т.е. ее параметр **predictiveState** был равен 1 благодаря какому-то ее латеральному сегменту), тогда эти клетки становятся активными (строки 23-27). Если этот сегмент стал активным из-за клеток выбранных для обучения (**learnState** ==1), тогда такая клетка также выбирается для обучения (строки 28-30). Если же текущий прямой вход снизу не был предсказан, тогда все клетки становятся активными (строки 32-34) и кроме того, клетка, лучше всего соответствующая входным данным, выбирается для обучения (строки 36-41), причем ей добавляется новый латеральный дендритный сегмент.

```
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.                 i = getBestMatchingCell(c, t-1)
38.                 learnState(c, i, t) = 1
39.                 sUpdate = getSegmentActiveSynapses (c, i, -1, t-1, true)
40.                 sUpdate.sequenceSegment = true
```

```
41.             segmentUpdateList.add(sUpdate)
```

Фаза 2

Вторая фаза вычисляет состояния предсказания (предчувствия активации) для каждой клетки. Каждая клетка включает свое состояние предчувствия (параметр **predictiveState**), если любой из ее латеральных дендритных сегментов становится активным, т.е. достаточное число его горизонтальных (боковых, латеральных) соединений становятся активными благодаря прямому входу. В этом случае клетка ставит в очередь на отложенное исполнение следующий ряд своих изменений: а) усиление активных сейчас латеральных сегментов (строки 47-48)) и б) усиление сегментов которые могли бы предсказать данную активацию, т.е. сегментов которые соответствуют (возможно, пока слабо) активности на предыдущем временном шаге (строки 50-53)).

```
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)
```

Фаза 3

Третья и последняя фаза занимается обучением. В этой фазе происходит реальное обновление сегментов (которое было поставлено в очередь на исполнение) в том случае если колонка клетки активирована прямым входом и эта клетка выбрана в качестве кандидатки для обучения (строки 55-57). В противном случае, если клетка по каким-либо причинам перестала предсказывать, мы ослабляем ее латеральные сегменты (строки 58-60)

```
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.
```

Детали имплементации и терминология

В данном разделе мы остановимся на некоторых деталях нашей реализации темпорального группировщика и используемой при этом терминологии. Каждая наша клетка индексируется с помощью двух чисел: индекса колонки **c**, и индекса клетки в ней **i**. Клетки содержат в себе списки дендритных сегментов, а каждый из этих сегментов содержит список синапсов плюс значение перманентности для каждого из них. Изменения в синапсах клетки маркируются как отложенные до тех пор, пока она опять не станет активной от прямого входа. Эти отложенные

изменения в синапсах содержатся в списке **segmentUpdateList**. Каждый сегмент кроме того содержит в себе логическую переменную (флаг) **sequenceSegment**, которая показывает, предсказывает ли данный сегмент активацию своей клетки от прямого входа в следующий момент времени.

Имплементация работы потенциальных синапсов здесь несколько отличается от их имплементации в пространственном группировщике. Там полный список потенциальных синапсов представляется явным списком. В темпоральном же группировщике каждый сегмент может иметь свой собственный (возможно большой) список потенциальных синапсов. Однако, на практике поддержание большого списка для каждого сегмента весьма накладно в плане производительности и требует много памяти. Поэтому, в темпоральном группировщике мы просто случайно добавляем активные синапсы к каждому сегменту во время обучения (что контролируется параметром **newSynapseCount**). Такая оптимизация этого процесса дает эффект, сходный с поддержанием полного списка потенциальных синапсов, но списки для каждого сегмента становятся гораздо короче при сохранении возможности обучению новым временным паттернам.

В нашем псевдокоде также используется небольшой математический автомат, чтобы следить за состояниями клеток в различные моменты времени. Для каждой клетки мы вводим три различных состояния. Массивы **activeState** и **predictiveState** хранят в себе записи об активных состояниях и состояниях предсказания (предчувствия) для каждой из клеток в каждый из моментов времени. Массив **learnState** определяет какие клетки используются во время обучения. Если данный прямой вход оказался неожиданным, все клетки в соответствующей колонке становятся активными на этот момент времени. Но только одна из этих клеток (которая лучше всего соответствует данному входу) включает свое состояние обучения **learnState**. И мы добавляем синапсы только клеткам с таким включенным параметром (что исключает излишнее использование дендритных сегментов для полностью активной колонки).

Следующие структуры данных используются в нашем псевдокоде темпорального группировщика.

cell(c,i)	Список всех клеток с индексами по **i** и **c**.
cellsPerColumn	Число клеток в каждой из колонок.
activeColumns(t)	Список индексов колонок – победителей в активации благодаря прямому входу (выход пространственного группировщика).
activeState(c, i, t)	Логический вектор с одним значением на каждую клетку. Он представляет активные состояния клеток колонки **c** с номером **i** в момент времени **t** при текущем прямом входе и временном контексте. **activeState(c, i, t)** это вклад от клетки колонки **c** с номером **i** во время **t**. Если это 1, то клетка активна при данном прямом входе и временном контексте.
predictiveState(c, i, t)	Логический вектор с одним значением на каждую клетку. Он представляет активные состояния клеток колонки **c** с номером **i** в момент времени **t** при текущей активности других колонок и временном контексте. **predictiveState(c, i, t)** это вклад от клетки колонки **c** с номером **i** во время **t**. Если это 1, то клетка предсказывает активацию от прямого входа при данном временном контексте.

learnState(c, i, t)	Логическое значение, показывающее, была ли клетка колонки **c** с номером **i** выбрана для обучения.
activationThreshold	Порог активации для сегмента. Если число активных подключенных синапсов в сегменте больше чем **activationThreshold**, данный сегмент считается активным.
learningRadius	Область вокруг клетки темпорального группировщика, откуда она может получить латеральные связи.
initialPerm	Начальное значение перманентности для синапсов.
connectedPerm	Если значение перманентности синапса больше данного значения, то он считается подключенным.
minThreshold	Минимальная активность в сегменте для обучения.
newSynapseCount	Максимальное число синапсов добавляемых сегменту при обучении.
permanenceInc	Количество значений перманентности синапсов, которое увеличивается при обучении.
permanenceDec	Количество значений перманентности синапсов, которое уменьшается при обучении.
segmentUpdate	Структура данных с тремя видами следующей информации, нужной для обновления данного сегмента: а) индекс сегмента (-1 если это новый сегмент), б) список существующих активных синапсов, и в) логический флаг, показывающий должен ли этот сегмент быть помеченным как сегмент последовательности (по умолчанию **false**).
segmentUpdateList	Список структур **segmentUpdate**. **segmentUpdateList(c,i**) это список изменений для клетки **i** в колонке **c**.

Следующие внутренние процедуры используются в нашем псевдокоде:

segmentActive(s, t, state)

Эта процедура возвращает **true,** если число подключенных синапсов сегмента **s**, которые активны благодаря заданным состояниям в момент **t,** больше чем **activationThreshold**. Вид состояний **state** может быть **activeState**, или **learnState**.

getActiveSegment(c, i, t, state)

Для данной клетки **i** колонки **c**, возвращает индекс сегмента такого, что **segmentActive(s,t, state)** равно **true**. Если активны несколько сегментов, то сегментам последовательностей отдается предпочтение. В противном случае предпочтение отдается сегментам с наибольшей активностью.

getBestMatchingSegment(c, i, t)

Для данной клетки **i** колонки **c** в момент **t**, находит сегмент с самым большим числом активных синапсов. Т.е. она ищет наилучшее соответствие. При этом значения перманентности синапсов допускаются и ниже порога **connectedPerm**. Число активных синапсов допускается ниже порога **activationThreshold**, но должно быть выше **minThreshold**. Данная процедура возвращает индекс сегмента. А если такого не обнаружено, то возвращается -1.

getBestMatchingCell(c)

Для данной колонки возвращает клетку с самым соответствующим входу сегментом (как это определено выше). Если такой клетки нет, то возвращается клетка с минимальным числом сегментов.

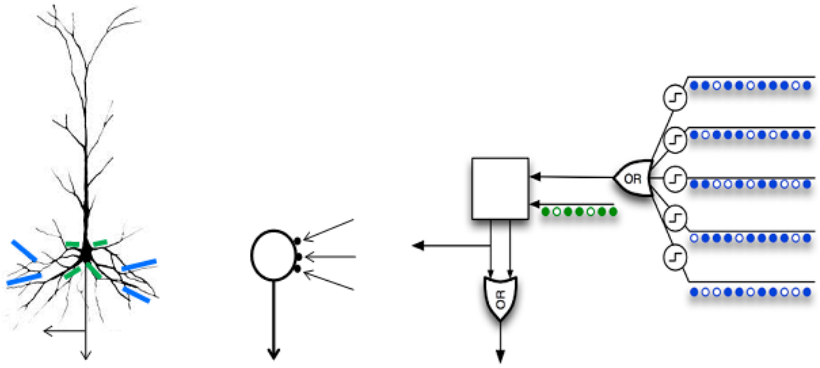
getSegmentActiveSynapses(c, i, t, s, newSynapses= false)

Возвращает структуру данных **segmentUpdate**, содержащую список предлагаемых изменений для сегмента **s**. Пусть **activeSynapses** список активных синапсов у исходных клеток которых **activeState** равно 1 в момент времени **t**. (Этот список будет пустым если **s** равно -1 при не существующем сегменте.) **newSynapses** это опциональный параметр, по умолчанию равный **false**. А если **newSynapses** равно **true**, тогда число синапсов, равное **newSynapseCount - count(activeSynapses),** добавляется к активным синапсам **activeSynapses**. Такие синапсы случайно выбираются из числа клеток, у которых **learnState** равно 1 в момент времени **t**.

adaptSegments(segmentList, positiveReinforcement)

Эта функция проходит по всему списку **segmentUpdate** и усиливает каждый сегмент. Для каждого элемента **segmentUpdate** делаются следующие изменения. Если **positiveReinforcement** равно **true,** тогда синапсы из списка **activelist** увеличивают значения своих перманентностей на величину **permanenceInc**. Все остальные синапсы уменьшают свои перманентности на величину **permanenceDec**. Если же **positiveReinforcement** равно **false**, тогда синапсы из списка активных уменьшают свою перманентность на величину **permanenceDec**. После этого шага любым синапсам из **segmentUpdate,** которые только что появились, добавляется значение **initialPerm**.

Приложение A. Сравнение между биологическими нейронами и HTM клетками



На этой картинке показано: слева схематическое изображение биологического нейрона (пирамидального), посередине – изображение простого искусственного нейрона и справа - нейрона HTM. В данном приложении нам хотелось бы еще раз, более подробно, объяснить работу клеток HTM путем их сравнения с биологическими нейронами и простыми искусственными нейронами.

Реальные нейроны мозга ужасно разнообразны и сложны. Поэтому, мы сфокусируемся только на самых основных принципах их работы, которые приложимы к нашей модели. И хотя мы тем самым опустим множество деталей работы биологических нейронов, все же клетки, используемые в алгоритмах обучения HTM, гораздо более реалистичны, чем искусственные нейроны, используемые ныне в большинстве т.н. нейронных сетей. Все рабочие элементы, включенные нами в состав HTM клеток, необходимы нам для выполнения тех или иных операций в HTM.

Биологические нейроны

Нейроны являются клетками – переносчиками информации в мозге. Картинка вверху слева изображает типичный возбуждающий нейрон. В его визуальном изображении в основном доминируют ветвящиеся дендриты. Все возбуждающие входы такого нейрона происходят через синапсы на этих дендритах. В последние годы наши знания о нейронах значительно расширились. И одним из самых больших изменений в наших представлениях о них явилось то, что дендриты нейронов не являются простыми проводниками, передающими приходящие импульсы к телу клетки. Теперь мы знаем, что дендриты фактически являются сложными, нелинейными элементами обработки сигналов. И в алгоритмах обучения HTM учитываются эти нелинейные свойства.

Биологические нейроны состоят из нескольких основных частей:

Тело клетки

Тело клетки, это небольшой объем в центре нейрона. Выходной сигнал нейрона передается от него с помощью его аксона, который начинается у тела клетки. А входами нейрона являются синапсы, прикрепленные к дендритам, которые передают сигналы к телу клетки.

Проксимальные (близкие) дендриты

Ветки дендритов (и их части) расположенные ближе всего к телу клетки называют проксимальными дендритами. На диаграмме вверху некоторые из проксимальных дендритов помечены зелеными линиями.

Многочисленные активные синапсы на проксимальных дендритах оказывают

практически линейный аддитивный эффект своими принимаемыми сигналами на тело клетки. Пять активных синапсов дают приблизительно в пять раз большую деполяризацию тела клетки, по сравнению с одним таким активным синапсом. Напротив, если один из таких синапсов будет активирован серией последовательных, быстро повторяющихся потенциалов действия, второй, третий и последующие импульсы (спайки) окажут гораздо меньший эффект на тело клетки, чем самый первый.

Следовательно, мы можем сказать, что воздействия на проксимальные дендриты линейно суммируются на теле клетки, и что быстрые серии спайков приходящие на отдельный синапс будут иметь лишь чуть-чуть больший эффект, чем одиночный спайк.

Поступающие извне связи в зону коры головного мозга преимущественно подключаются к проксимальным дендритам. Это было отмечено, по крайней мере, для нейронов 4-го слоя неокортекса, основного входного слоя нейронов для каждой зоны мозга.

Дистальные (удаленные) дендриты

Более удаленные от тела клетки ветви дендритов, называют дистальными дендритами. На диаграмме вверху некоторые из дистальных дендритов помечены синими линиями.

Дистальные дендриты гораздо тоньше, чем проксимальные дендриты. Они подключаются к другим дендритам в точках ветвления дендритного дерева и никогда не подключаются непосредственно к телу клетки. Такие отличия дают дистальным дендритам уникальные электрические и химические свойства. Когда только один синапс активируется на дистальном дендрите, он дает самый минимальный эффект на тело клетки. Деполяризация, которая образуется локально у синапса, сильно ослабевает к тому времени, когда она достигает тела клетки. Многие годы это представлялось большой загадкой для исследователей. Казалось, что дистальные синапсы, которые составляют абсолютное большинство среди всех синапсов нейрона, практически не могут никак на него повлиять.

Теперь мы знаем, что секции дистальных дендритов работают как полуавтономные обрабатывающие элементы. Если достаточное число их синапсов становятся активными в одно и то же время, на небольшом участке дендрита, то они генерируют дендритный спайк, который может дойти до тела клетки, вызвав значительный эффект. Например, 20 активных синапсов на расстоянии не более 40 микрон друг от друга сгенерируют общий дендритный спайк.

Следовательно, можно сказать, что дистальные дендриты работают как множество пороговых детекторов совпадающих импульсов.

Синапсы, формируемые на дистальных дендритах, в основном связаны с другими ближайшими клетками данной области коры мозга.

На рисунке биологического нейрона вверху показана также одна большая дендритная ветвь, направляющаяся вверх, которую называют апикальным дендритом. Одна из теорий гласит, что эта структура нейрона позволяет ему разместить несколько дистальных дендритов в той области, где им проще всего будет установить связи с проходящими там аксонами. В рамках такой интерпретации, апикальный дендрит пирамидального нейрона работает просто как продолжение тела клетки.

Синапсы

Типичный нейрон мозга может иметь несколько тысяч синапсов. И в подавляющем большинстве случаев (возможно, до 90%) они будут располагаться на его дистальных дендритах, а остальные синапсы – на проксимальных.

В течении многих лет предполагалось, что обучение мозга подразумевает усиление и/или ослабление эффекта воздействия, или «веса», синапсов нейронов. И хотя все это многократно наблюдалось в экспериментах, тем не менее, реальное воздействие каждого синапса на нейрон является сугубо стохастическим. При активации синапса, он, например, не всегда выбрасывает

нейротрансмитер. И, следовательно, общий алгоритм работы мозга никак не может зависеть от точности или исполнительности срабатывания веса отдельного синапса.

Более того, теперь мы знаем, что сами синапсы между нейронами очень быстро создаются и быстро расформировываются. Такая их гибкость представляет собой очень мощную форму обучения и лучше объясняет быстрое усвоение новых знаний. При этом, синапсы могут сформироваться только тогда, когда аксон и дендрит находятся друг от друга не далее определенного расстояния, что приводит нас к концепции «потенциальных» синапсов. В рамках такого предположения, мы можем сказать, что обучение происходит путем формирования действующих синапсов из потенциальных синапсов.

Выход нейрона

Выходным сигналом нейрона является спайк, или «потенциал действия», который распространяется вдоль его аксона. Аксон начинается из тела клетки и почти всегда разделяется на две ветви. Одна из них направляется горизонтально и создает множество связей с другими близлежащими клетками. Другая ветвь проектируется в другие уровни клеток или даже в другие области и структуры мозга. На рисунке нейрона вверху его аксон изначально не был показан. Поэтому мы добавили линию и две стрелки для его схематического обозначения.

Хотя реальным выходным сигналом нейрона всегда являются его спайки, существует несколько различных точек зрения, как их можно интерпретировать. Доминирующей точкой зрения (особенно, что касается коры головного мозга) является то, что основное значение имеет частота спайков. Таким образом, выход нейрона можно представить с помощью некоторой скалярной величины (равной этой частоте).

Кроме того, некоторые из нейронов демонстрируют «взрывное» поведение, проявляющиеся в быстрой серии из нескольких спайков, которая существенно отличается от регулярной импульсной активности.

Данное нами здесь описание биологического нейрона является лишь чрезвычайно кратким, начальным введением в некоторые выделенные его свойства. Оно фокусируется только на тех аспектах деятельности нейронов, которые имеют отношение к свойствам клеток НТМ, и опускает великое множество прочих деталей. При этом, далеко не все из описанных здесь свойств широко воспринимаются сейчас научной общественностью. Тем не менее, мы включили их сюда, поскольку они необходимы для обоснования наших моделей. Все те знания, которые сейчас вообще известны о нейронах мозга, могут легко стать содержанием нескольких весьма толстых книжек. Кроме того, дальнейшие активные исследования свойств нейронов продолжаются и в настоящее время.

Простые искусственные нейроны

На среднем рисунке, в начале этого приложения, показан простой нейроно-подобный элемент, который используется во многих моделях классических искусственных нейронных сетей. У такого искусственного нейрона есть набор синапсов, каждый со своим весом. Каждый синапс получает скалярное возбуждение, которое умножается на его вес. Результаты синапсов суммируются нелинейным образом, чтобы получить выход такого искусственного нейрона. Его обучение осуществляется подбором весов синапсов и, возможно, упомянутой нелинейной функции.

Такой тип искусственных нейронов, как и его вариации, доказал, как свою применимость во множестве практических приложений, так и проявил себя как полезное вычислительное средство. Тем не менее, он не включает в себя всей сложности внутреннего устройства и возможности обработки информации присущие биологическим нейронам. Если нашей целью является понимание и моделирование того, как работают ансамбли реальных нейронов в мозге, то нам нужны их более сложные и точные модели.

Клетки HTM

На нашей иллюстрации, в начале этого приложения, рисунок справа представляет собой схему клетки HTM, используемую в наших кортикальных («мозгоподобных») алгоритмах обучения. Клетки HTM включают в себя, как множество важных особенностей реальных нейронов мозга, так и несколько явных упрощений механизмов их работы.

Проксимальные дендриты

Каждая клетка HTM имеет один проксимальный дендрит. Прямой ввод исходных данных в нее происходит с помощью синапсов (показаны зелеными кружочками). Активность таких синапсов линейно суммируется для получения прямой активации данной клетки.

Мы потребовали, что бы все клетки в одной колонке имели одинаковое восприятие прямых входных данных. Для реальных нейронов это скорее всего происходит с помощью определенного типа ингибиторных (подавляющих) клеток. А в HTM мы просто заставили все клетки в одной колонке использовать один проксимальный дендрит.

Чтобы избежать такой ситуации, когда клетка никогда не побеждает в конкуренции со своими соседями за активацию, клетки HTM сами усиливают свою прямую активацию, если они достаточно долго не активировалась относительно своих соседей. То есть, между клетками идет постоянное соперничество и конкуренция. Хотя в HTM мы заменяем его, строго говоря, на конкуренцию между колонками, а не клетками. На нашей диаграмме такое соперничество не показано.

Наконец, проксимальный дендрит имеют у себя множество потенциальных синапсов, являющееся подмножеством всех входных битов данного региона. По мере обучения клетки, она увеличивает или уменьшает специально введенную нами величину перманентности всех потенциальных синапсов проксимального дендрита. При этом, реально действующими являются только те потенциальные синапсы, чья перманентность выше некоторого порога.

Как уже упоминалось ранее, такая концепция («перманентности») потенциальных синапсов была взята нами из биологии, где она означала аксон и дендрит расположенные достаточно близко друг к другу, чтобы сформировать синапс. Мы несколько расширили эту концепцию на более широкое множество потенциальных связей HTM клетки. Дендриты и аксоны биологических нейронов могут расти и уменьшаться по мере обучения, что приводит к изменению множества потенциальных синапсов по мере такого роста. Сделав множество потенциальных синапсов HTM клетки достаточно большим, мы достигли приблизительно тех же результатов, что дает такой рост дендритов с аксонами. Множество потенциальных синапсов на нашем рисунке также не показано.

Описанная комбинация конкуренции между колонками за активацию, обучение множества потенциальных синапсов и повышение чувствительности для неиспользуемых колонок дает региону HTM очень высокую пластичность, подобную той, что мы наблюдаем в мозге. (При этом, HTM регион автоматически подбирает, что будет представлять (хранить) каждая его колонка (путем изменения синапсов на проксимальных дендритах) при изменении состава входных данных, или при изменении числа своих колонок.)

Дистальные дендриты

Каждая клетка HTM содержит в себе список сегментов дистальных дендритов. При этом, каждый такой сегмент работает как некоторый пороговый детектор. Если число активных синапсов любого из таких сегментов клетки (показанных синими кружочками на вышеприведенной диаграмме) становится выше его порогового значения, данный сегмент становится активным и связанная с ним клетка переходит в состояние предчувствия (предсказания) поскольку это ее состояние является функцией логического ИЛИ от активации всех ее дистальных сегментов.

Сегмент дистальных дендритов запоминает нужное ему состояние региона HTM путем формирования связей с его клетками, которые ранее были активны в один

и тот же момент времени. Таким образом, сегмент запоминает состояние региона, которое предшествовало активации его клетки прямыми входными данными. А потом сегмент просто следит за состоянием этих «своих» клеток региона HTM, активация которых предсказывает, что и его клетка станет сейчас активной. Типичный порог для дендритного сегмента равен 15. Если 15 действующих синапсов такого сегмента становятся сразу активными, данный дендрит также активизируется. Поблизости могут быть активными сотни и даже тысячи клеток региона, но существенными являются связи только с 15 из них для распознания большего паттерна.

Кроме того, каждый дистальный дендритный сегмент клетки HTM имеет также свое множество потенциальных синапсов, которое проецируется на подмножество всех клеток в регионе. По мере обучения этого сегмента, он увеличивает или уменьшает величину перманентности всех своих потенциальных синапсов. Только те синапсы, перманентность которых выше некоторого порога, являются действующими для данной клетки HTM.

В одной из своих реализаций HTM мы использовали фиксированное число дендритных сегментов у каждой клетки. В другой реализации мы добавляли и удаляли сегменты прямо во время обучения. Оба этих метода оказались работоспособными. Если у нас есть фиксированное число дендритных сегментов у каждой клетки, то мы можем запомнить несколько различных наборов синапсов для одного сегмента. Например, пусть у нас есть 20 действующих синапсов на сегменте и порог равный 15. (Вообще говоря, нам желательно, чтобы порог был меньше чем число синапсов, чтобы увеличить устойчивость к шумам.) Такой сегмент может теперь распознать одно конкретное состояние своих ближайших клеток. Что произойдет, если мы добавим еще 20 действующих синапсов на тот же сегмент, которые будут представлять совершенно другое состояние близлежащих клеток? Это дает возможность ошибки, потому что на сегменте могут активизироваться 8 синапсов из одного паттерна и 7 из другого, что даст его ошибочную суммарную активизацию. Но мы экспериментально обнаружили, что до 20 различных паттернов можно запомнить на одном сегменте, до появления подобных ошибок. Следовательно, клетка HTM с дюжиной дендритных сегментов может участвовать во множестве различных предсказаний.

Синапсы

Синапсы HTM клетки имеют бинарный вес. Хотя ничто в HTM модели не исключает использование «привычных» скалярных весов синапсов, но благодаря использованию разреженных пространственных паттернов у нас пока не появилось в этом никакой необходимости.

Тем не менее у синапсов HTM клетки есть скалярная величина называемая «перманентностью», которая изменяется во время обучения. Значение перманентности синапса равное 0.0 говорит о том, что он не действующий (отключен) и даже никак не продвигается, чтобы им стать. Значение перманентности немногим выше порога (обычно равного 0.2) представляет собой синапс, который только что подключился и может легко отключиться в дальнейшем. Высокое значение перманентности, например 0.9, представляет собой синапс, который надежно подключен и не может быстро отключиться и перестать действовать.

Число действующих синапсов проксимального и дистальных дендритных сегментов клетки HTM не фиксировано. Оно изменяется по мере предъявления клетке входных паттернов (т.е. ее обучения). При этом, число действующих синапсов на дистальных дендритах существенно зависит от временной структуры входных данных. Если в них отсутствуют устойчивые временные паттерны, то все синапсы дистальных сегментов будут иметь низкую перманентность и очень малое их число будет действующими. А если во входном потоке данных наблюдается много упорядоченных временных структур, то тогда мы получим много действующих синапсов с высокой перманентностью.

Результат (выход) клетки

Клетка HTM имеет два различных положительных бинарных состояний (выходов): 1) активность клетки из-за прямого воздействия входных данных (через ее

проксимальный дендрит) и 2) активность клетки благодаря латеральным (боковым) связям (через дистальные дендритные сегменты). Первое из них мы называем просто «активным состоянием», а второе – «состоянием предсказания» (или «предчувствия»).

На приведенной выше диаграмме эти два выхода представлены двумя линиями, исходящими из прямоугольного тела клетки. Левая линия это прямое активное состояние, а правая – состояние предчувствия. Только прямые активные состояния влияют на другие клетки региона, гарантируя, что все предсказания основываются только на текущем прямом входе (плюс его контекст). Мы не хотим делать предсказания на основе других предсказаний. Если мы это допустим, то практически все клетки в регионе будут в состоянии предчувствия всего после нескольких итераций.

Выходом (или результатом) для всего региона у нас является вектор, представляющий состояния всех клеток. Он передается на вход следующему региону в иерархии, если такие существуют. Такой результат является логическим ИЛИ активных состояний и состояний предчувствия клеток. Комбинируя вместе оба этих состояния, наш результат работы региона становится значительно более стабильным (медленнее изменяется во времени) чем его вход. Такая стабильность является важным результатом всей работы региона.

Рекомендуемое чтение

Нас очень часто просят рекомендовать литературу для дальнейшего изучения нейрофизиологии. Однако эта область сейчас настолько обширна, что даже самое общее введение в нее требует изучения нескольких весьма солидных источников. А новые находки постоянно публикуются только в академических журналах, которые и сложны для прочтения и весьма труднодоступны, если только вы не связаны с соответствующими ВУЗами и институтами.

Есть две доступные для прочтения [на английском] книги, которые заинтересованный читатель возможно захочет прочесть и в которых освещаются темы относящиеся к данному приложению:

Greg Stuart, Nelson Spruston, Michael Hauser «Dendrites, second edition» (New York: Oxford University Press, 2008)

[частично имеется по адресу (на английском языке), для ознакомления: http://books.google.ru/books?id=hbn1RRrPOO4C&printsec=frontcover#v=onepage&q&f=false)]

Эта книга является хорошим источником знаний по всему, что касается дендритов. В ее 16-ой главе обсуждаются нелинейные свойства дендритных сегментов, используемые в алгоритмах обучения HTM. Эта глава написана Бартлетом Мелом (Bartlett Mel), который сделал множество работ в данной области.

Vernon B. Mountcastle «Perceptual Neuroscience: The Cerebral Cortex» (Cambrige, Mass.: Harvard University Press, 1998)

[частично имеется по адресу (на английском языке), для ознакомления: http://books.google.ru/books?id=egQPpf-vGh8C]

Данная книга является хорошим введением для всего, что касается коры головного мозга. В некоторых ее главах обсуждаются типы клеток и их связи. Там вы можете многое узнать о нейронах и их связях, хотя эта книга уже слишком старая, чтобы включить в себя все последние знания о дендритных свойствах.

[Дополнение переводчика: имеется перевод на русский язык работы В. Мунткасла «Организующий принцип функции мозга: элементарный модуль и распределенная система» расположенный по адресу: http://dmitry.bancorp.ru/Hawkins/mountcastl.html (оттуда перевод скопирован во многие другие русскоязычные электронные библиотеки).]

Приложение Б: Сравнение слоев коры мозга и регионов НТМ

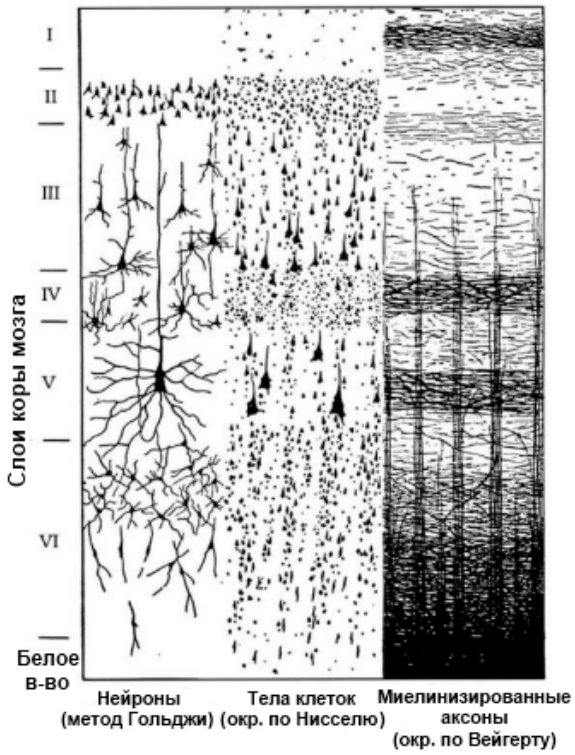
В данном приложении описывается соотношение между описанными нами регионами НТМ и нейробиологическими областями реальной коры головного мозга.

Более конкретно, здесь рассматривается, как алгоритмы самообучения НТМ, с их колонками и клетками, соотносятся с многослойным и колончатым строением биологического неокортекса. Многие люди часто путаются в концепции «уровней» (или слоев) в коре мозга и как они соотносится с уровнями в НТМ. Мы надеемся, что данное приложение полностью устранит эту путаницу и дополнительно разъяснит биологические основания кортикальных алгоритмов самообучения НТМ.

Связи в коре мозга

Кора головного мозга человека представляет собой поверхность из нейронной ткани, приблизительно 1000 см2 по площади и 2 мм толщиной. По этим параметрам она весьма напоминает большую ресторанную салфетку, которой прикрывают одежду во время еды. Кора подразделяется на десятки функциональных областей, некоторые из которых относятся к зрению, другие к звукам, третьи к разговорному языку и т.д. Однако, все они под микроскопом выглядят на удивление похожими друг на друга.

В каждой такой зоне неокортекса можно наблюдать несколько организационных принципов их построения.



Слои

Считается, что кора мозга имеет, в общем случае, шесть слоев. Пять из них содержат нейроны, а один, в основном, только их связи. Эти слои были открыты более ста лет назад с применением техник окрашивания клеток. Рисунок выше (по Кахалю) показывает небольшой срез коры окрашенный с помощью трех

различных методов. Боковая сторона рисунка представляет собой толщину коры мозга, приблизительно 2 мм. С левой стороны обозначены шесть слоев коры, вплоть до начала белого вещества мозга (внизу рисунка), где аксоны нейронных клеток проходят в другие области коры и в другие части мозга.

С правой стороны рисунка показана окраска, проявляющая только миелинизированные аксоны (Миелин, это жироподобное вещество покрывающее некоторые (но не все) аксоны.) В этой части рисунка вы можете наглядно видеть два главных принципа организации неокортекса: его слои и колонки. Кроме того, большинство из аксонов разделяются на две ветви сразу после выхода из тела нейрона. Одна из них идет, в основном, горизонтально, а другая больше вертикально. Горизонтальная ветвь создает множество связей с другими клетками в том же или ближайших уровнях коры, почему они и становятся столь заметными при такой окраске. Не забывайте, что это вертикальный срез реальной коры, в котором большинство аксонов выходят и отклоняются из плоскости рисунка в обе стороны, так что в реальности они существенно длиннее, чем это здесь изображено. Подсчитано, что в каждом кубическом миллиметре коры мозга находится от 2 до 4 километров аксонов и дендритов нейронов.

В средней части нашего рисунка показаны окрашенные тела клеток нейронов, без их дендритов или аксонов. Здесь вы можете увидеть, что размеры и плотность нейронов значительно варьируется от слоя к слою. И проявление колонок на данной части рисунка совсем незначительно. Также можно наблюдать несколько нейронов в слое 1, но число их там настолько мало, что он по-прежнему считается практически безнейронным слоем. Нейрофизиологи оценивают число нейронов в кубическом миллиметре коры как приблизительно 100 000 клеток.

В левой части рисунка показана окраска проявляющая тела, аксоны и дендриты только нескольких из всех нейронов. На ней вы можете видеть, что толщина дендритных «ветвей» также существенно варьируется у клеток в различных слоях коры. Также видны некоторые апикальные дендриты, которые поднимаются из тела клеток и создают связи в других слоях. Наличие и адресация апикальных дендритов специфична для каждого уровня.

Подводя итог, многоуровневая и колончатая организация коры мозга довольно наглядно проявляется с помощью окраски нейронной ткани и наблюдения под микроскопом.

Различия в уровнях коры для разных ее областей

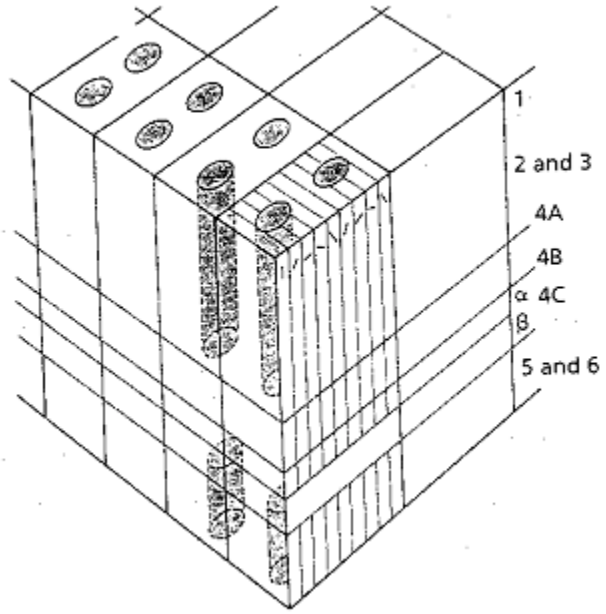
Толщина слоев в различных областях неокортекса несколько отличается друг от друга и, кроме того, существуют разные мнения относительно общего числа таких слоев. Эти отличия зависят от того, мозг какого животного изучается, какие области коры при этом рассматриваются, и кто именно все это делает. Например, на нашем рисунке вверху довольно просто различить слои 2 и 3, но в общем случае это не так. Некоторые ученые отмечали, что они не могли разделить эти два слоя в областях коры, которые они изучали, поэтому часто слои 2 и 3 объединяют вместе и называют «слой 2/3». Другие ученые, наоборот, выделяют там подслои, например, 3А и 3В.

Слой номер 4 наиболее легко выделяется в тех регионах неокортекса, которые ближе всего к восприятию сенсорной информации. Например, у некоторых животных (таких как обезьяны или люди) уровень 4 легко выделяется в первичных зрительных областях неокортекса. У других животных он не выделяется. Кроме того, уровень 4 практически исчезает в областях коры иерархически удаленных от восприятия сенсорной информации.

Колонки

Вторым основным принципом организации неокортекса являются колонки. Его колончатое строение визуально наблюдается при исследовании окрашенных срезов, но более важным обоснованием колонок являются реакции нейронов на различные входные образы.

Когда ученые использовали микроэлектроды, чтобы узнать, что заставляет нейроны активизироваться, они обнаружили, что сходно реагирующие нейроны выстраиваются в коре мозга вертикально, проходя сквозь различные ее слои.



На этом рисунке показаны некоторые реакции нейронов визуальной зоны V1, которая первой в неокортексе обрабатывает зрительную информацию, приходящую с сетчатки глаза.

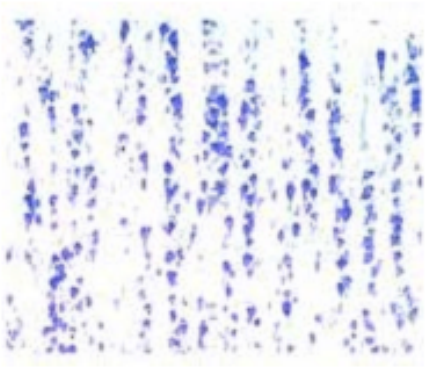
Одним из первых открытий в данной области было, что большинство нейронов в V1 реагируют на линии или грани различной ориентации, появляющиеся в специфических зонах сетчатки глаза. При этом, клетки, располагающиеся в одних вертикальных колонках, все реагировали на грани одной и той же ориентации. Если вы посмотрите внимательно на рисунок, то вы можете увидеть небольшие черточки наверху каждой вертикальной секции коры, которые обозначают указанную ориентацию, на которую реагируют нейроны. Клетки находящиеся в одной вертикальной секции реагируют на линии одной ориентации.

Кроме того, в V1 наблюдается еще несколько свойств колончатой организации, два из которых показаны на нашем рисунке. Там наблюдаются «колонки глазного доминирования», в которых нейроны реагируют на сходные комбинации раздражителей от правого либо левого глаза. Кроме того, там наблюдаются «пятна» (blobs), где нейроны в основном реагируют на цветовые раздражители. На нашем рисунке, колонки глазного доминирования представлены большими вертикальными блоками, которые состоят из колонок реагирующих на ориентацию. «Пятна» изображены темными овалами.

Общее правило организации нейрокортекса состоит в том, что в нем несколько областей сходного реагирования перекрываются между собой, как например колонки ориентации и глазного доминирования. Если вы будете продвигаться горизонтально по поверхности коры, то комбинация воспринимаемых раздражителей нейронов будет изменяться. Тем не менее, вертикально расположенные нейроны будут повторять набор воспринимаемых комбинаций. Такая колончатая организация проверена для аудиальной, визуальной и соматосенсорной областей неокортекса. Среди нейрофизиологов идет дискуссия верно ли это для всей коры, но уже сейчас ясно, что это соблюдается для большинства областей коры, если вообще не для всех.

Миниколонки

Наименьшей колончатой структурой неокортекса являются миниколонки. Они составляют приблизительно 30мкр в диаметре и состоят из 80 – 100 нейронов, расположенных на всех пяти клеточных уровнях. Весь неокортекс состоит из таких миниколонок. Их можно себе представить, как маленькие отрезки спагетти, которые прилегают своими сторонами друг к другу. Между колонками существуют крошечные промежутки, почти без клеток, что делает их иногда видимыми на окрашенных срезах коры.



На левом окрашенном срезе коры вы можете видеть отдельные тела нейронов. Он явно демонстрирует вертикальную структуру состоящую из миниколонок. Справа показана концептуальная схема миниколонки (по Питерсу (Piters) и Юлмезу (Yilmez)). В реальности, она гораздо меньше, чем здесь изображено. Обратите внимание, что на каждом слое миниколонки располагается множество нейронов, и все они реагируют на сходный входной паттерн. Например, на предыдущем изображении зоны V1, миниколонка будет состоять из клеток, реагирующих на линии сходной ориентации, поступающие преимущественно от одного из глаз. А клетки в прилегающей миниколонке могут реагировать на линии немного другой ориентации или поступающие от другого глаза.

Очень большую роль в формировании миниколонок играют подавляющие (ингибиторные) нейроны. Они не отображены на рисунке среза или на концептуальной схеме миниколонки, но они посылают свои аксоны в вертикальном направлении прямо между миниколонками, что частично и определяет их физическое разделение друг от друга. Полагают, что подавляющие нейроны также заставляют все нейроны миниколонки реагировать на сходный входной паттерн.

Описанные миниколонки неокортекса являются прототипом колонок используемых в кортикальных алгоритмах обучения HTM.

Исключение в колончатом реагировании.

В описанном колончатом реагировании на стимулы существует одно исключение, которое имеет отношение к нашим кортикальным алгоритмам обучения HTM. Обычно, ученые наблюдали, как реагирует нейрон, демонстрируя экспериментальному животному определенный стимул. Например, они могли показать животному небольшую линию в определенном участке визуального поля, чтобы пронаблюдать реакцию нейронов в зоне V1. При использовании таких простых стимулов исследователи обнаружили, что все нейроны всегда на них реагируют одинаково. Однако, если тот же стимул включить в видеоролик с естественными природными сценами, клетки становятся гораздо более избирательными. Некоторые клетки, которые четко реагировали на отдельную вертикальную линию, теперь вовсе не всегда будут реагировать на вертикальную линию в сложных движущихся образах из естественных, природных сцен.

В кортикальных алгоритмах обучения HTM все клетки в одной колонке имеют один входной паттерн. Но, при восприятии выученных временных последовательностей, только одна из клеток HTM колонки становится активной. Этот механизм является основой представления в HTM последовательностей переменного порядка, и он же является аналогом только что описанного свойства нейронов в миниколонках. Простой входной паттерн без исторического контекста активизирует все нейроны в колонке. Но тот же паттерн в составе выученной последовательности приведет к активизации всего лишь одной клетки.

При этом, мы вовсе не утверждаем, что в миниколонке коры активизируется

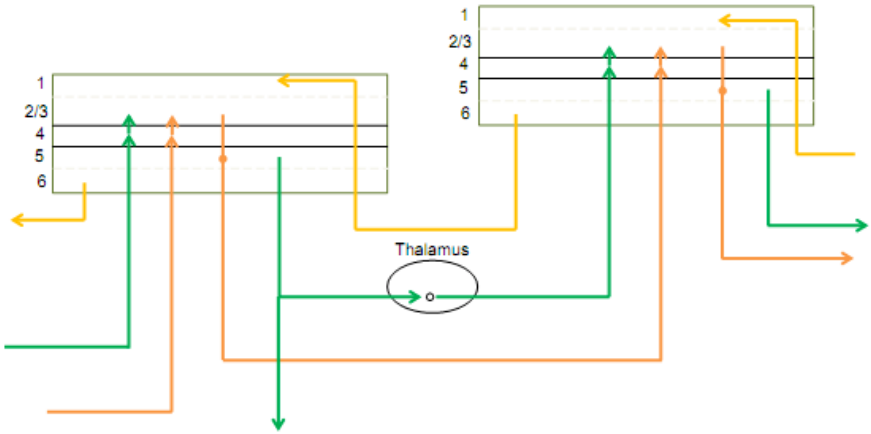
только один нейрон. Кортикальный алгоритм обучения HTM предполагает, что внутри колонки все нейроны (на своем уровне) будут активными для непредвиденного входного паттерна и только некоторое их подмножество будет активным для предполагаемого входного образца.

Зачем нужны все эти слои и колонки?

На сегодняшний день никто точно не знает, почему и зачем существуют все эти слои и колонки в коре головного мозга. Тем не менее, теория HTM может предложить некоторый ответ на этот вопрос. Кортикальный алгоритм обучения HTM показывает нам, что слой клеток, организованных в колонки может служить памятью значительного объема для переходов между состояниями переменного порядка. Или, проще выражаясь, слой клеток может выучить множество последовательностей событий. Колонки клеток с одним общим входом, являются ключевым механизмом для запоминания переходов переменного порядка.

Эта гипотеза объясняет нам зачем нужны колонки, но как насчет пяти слоев в неокортексе? Если даже один слой может научится последовательностям событий и делать предсказания, почему мы видим пять слоев в коре головного мозга?

Мы выдвигаем предположение, что различные слои, наблюдаемые в неокортексе, занимаются выявлением и запоминанием последовательностей, используя для этого один базовый механизм. Но последовательности, выделяемые на каждом таком слое, отличаются друг от друга. Мы пока еще многого не понимаем здесь, однако можем рассказать основную идею. Но перед этим, будет очень полезно рассмотреть, как нейроны в каждом слое соединены друг с другом.



На этой диаграмме показаны две области неокортекса и основные связи между ними. Такие связи повторяются по всему неокортексу, когда две его области связаны между собой. Область слева находится ниже в иерархии, чем область справа. Поэтому, передача информации происходит на рисунке слева направо. Стрелка вниз означает проекции на другие области мозга. Обратная связь идет справа налево. Каждая из изображенных областей мозга разделена на слои. Слои 2 и 3 рассматриваются вместе, как слой 2/3.

Цветные линии на нашем рисунке обозначают выходные сигналы нейронов различных слоев, то есть пучки их аксонов. Напомним, что многие аксоны почти сразу разделяются на две ветви. Одна из них идет горизонтально внутри своей области, и обычно внутри своего слоя в нем. Вот почему все клетки одного слоя очень тесно связаны между собой. Однако, сами нейроны и все их горизонтальные связи внутри слоев не показаны на нашем рисунке.

На рисунке имеется два пути распространения прямого входа. Прямой путь показан оранжевым цветом, а косвенный путь показан зеленым. Слой 4 является основным адресатом прямого входа сигналов и получает их от обоих указанных путей. Слой 4 посылает свои проекции на слой 3.

Слой 3 при этом является источником для прямого пути распространения

входного сигнала. То есть прямой путь ограничивается слоями 3 и 4 неокортекса.

Некоторые входные сигналы сразу минуют слой 4 и идут на слой 3. И как мы уже упоминали ранее, слой 4 практически исчезает в областях коры далеких от восприятия сенсорной информации. Для них прямой путь для входных сигналов просто идет из 3-го слоя в 3-ий слой следующей зоны коры.

Второй путь распространения входных сигналов (показанный зеленым цветом) начинается на слое 5. Клетки слоя 3 соединяются с клетками уровня 5 по дороге в следующий по иерархии регион. После выхода из коры мозга аксоны слоя 5 опять разделяются. Одна их ветвь уходит в субкортикальные структуры мозга, которые участвуют генерации моторного поведения. Полагают, что эти аксоны являются командными для моторных действий (показаны на рисунке как стрелка направленная вниз). Вторая ветвь уходит в другую часть мозга, называемую таламус, которая работает как пропускной пункт, который либо передает информацию в следующий регион, либо блокирует ее.

И, наконец, основной путь обратной связи, показанный желтым цветом, начинается в слое 6 и приходит в слой 1. Нейроны из слоев 2,3 и 5 соединяются со слоем 1 с помощью своих апикальных дендритов (не показаны на рисунке). Слой номер 6 получает входные сигналы из слоя 5.

Наше описание является очень кратким изложением того, что известно на текущий момент о связях между слоями неокортекса. Однако, его достаточно для понимания нашей гипотезы, почему наблюдается несколько слоев в коре мозга, если все они работают с последовательностями событий.

Гипотеза о том, что делают различные слои неокортекса

Мы выдвигаем предположение, что уровни 3,4 и 5 являются слоями восприятия входных сигналов, которые выявляют и запоминают в этих сигналах последовательности событий. Слой 4 выбирает последовательности первого порядка. Слой 3 занимается последовательностями переменного порядка. А слой 5 работает с последовательностями событий переменного порядка учитывая временные отметки (интервалы) событий. Давайте рассмотрим все их более подробно.

Слой 4

Изучать последовательности событий первого порядка с помощью кортикального алгоритма обучения HTM очень просто. Если мы не будем заставлять клетки в колонке подавлять друг друга, то так и получится, что клетки не будут различать контекст предыдущих входных сигналов и это будет изучением последовательностей первого порядка. В неокортексе это возможно сделать, если убрать любой эффект подавления между клетками в одной колонке. А в нашей компьютерной модели кортикального алгоритма обучения мы можем просто назначить каждой колонке только одну клетку, что даст нам тот же самый результат.

Последовательности первого порядка нужны, например, для формирования инвариантного представления пространственных трансформаций наблюдаемых сигналов. Для зрения, например, перемещение по осям X и У, масштабирование и поворот, все являются пространственными трансформациям. Когда HTM регион с памятью первого порядка обучается на движущихся объектах, он тем самым изучает, что различные пространственные входные паттерны являются эквивалентными. Полученные таким образом HTM клетки будут вести себя как то, что мы называем «сложными клетками» (complex cells) в неокортексе. Такие HTM клетки будут оставаться активными (в состоянии предчувствия) не смотря на множество пространственных трансформаций входных сигналов.

В компании Numenta мы провели ряд экспериментов с визуальными образами, для проверки работы этого механизма. Как и ожидалось, некоторая пространственная инвариантность действительно достигается на каждом уровне. Однако, подробное рассмотрение результатов этих экспериментов находится вне рамок данного приложения.

Предположение о том, что слой 4 изучает последовательности первого порядка,

согласуется с тем фактом, что сложные клетки находили именно в этом 4-ом слое и что данный слой практически исчезает в областях мозга, более высоких по иерархии обработки сенсорной информации. По мере подъема по такой иерархии, в какой-то момент становится уже просто невозможно выявлять еще большую пространственную инвариантность образов, поскольку их представления становятся уже полностью инвариантными.

Слой 3

Слой 3 ближе всего к кортикальному алгоритму обучения HTM, который мы описали в главе 2. Он изучает последовательности событий переменного порядка и формирует предсказания, которые более стабильны, чем входные образы. Слой 3 всегда посылает свои выходы в следующую по иерархии область обработки сенсорных сигналов и, следовательно, вносит свой вклад в увеличение временной стабильности образов внутри этой иерархии. Память последовательностей переменного порядка ведет к появлению т.н. «ориентированных сложных клеток» («directionally-tuned complex cells»), которые впервые были найдены в слое 3. Такие клетки различают временной контекст событий, например, что «их» линия движется вправо, а не влево.

Слой 5

Последним слоем, изучающим входные сигналы, является слой 5. Мы предполагаем, что слой 5 подобен слою 3, но с тремя существенными отличиями. Первое из них состоит в том, что 5-й слой добавляет концепцию времени. Слой 3 предсказывает только «что» должно случиться далее, но не говорит «когда» это произойдет. Тем не менее многие практические задачи просто требуют учета времени, например, распознавание произносимых слов, где относительные интервалы между звуками очень важны. Моторные движения являются другим таким примером, в котором координация по времени очень важна для управления усилиями отдельных мускулов. Мы предполагаем, что слой 5 предсказывает следующее состояние только после прошествия ожидаемого для него времени. И имеется несколько биологических фактов, которые подкрепляют эту гипотезу. Один из них в том, что слой 5 является источником моторных выходных сигналов для неокортекса. Другой факт в том, что слой 5 получает входные сигналы от слоя 1, которые приходят из таламуса (они не показаны на нашем рисунке). Мы предполагаем, что в этой информации закодировано время, распространяемое множеству клеток через таламус в слой 1 (также не показано на нашем рисунке).

Второе отличие между слоем 3 и слоем 5 в том, что мы хотим получать предсказания от слоя 3 настолько далеко в будущее, насколько это вообще возможно, что дает нам временную стабильность представлений. Кортикальный алгоритм обучения описанный в главе 2 делает именно это. Напротив, от слоя 5 нам нужно предсказание только следующего элемента (в заданное время). Мы не моделировали это отличие, но это легко сделать если запоминать нужный переход с соответствующим временем на его совершение.

Третье отличие между слоями 3 и 5 можно увидеть на нашей схеме связей областей коры. Выход из слоя 5 всегда уходит в подкорковые моторные центры, а вход их в другие области коры контролируется таламусом. То есть сигнал со слоя 5 в другую область коры иногда проходит, а иногда блокируется. Мы (как и некоторые другие исследователи) предполагаем, что этот контроль связан со скрытым вниманием (скрытое внимание, это когда вы обращаете внимание на входной сигнал без какого либо моторного движения).

Подводя итог, слой 5 комбинирует в себе временные интервалы, внимание и моторное поведение. Сейчас очень многое непонятно в том, как все они объединяются вместе. Но нам хотелось бы отметить, что небольшое дополнение к кортикальному алгоритму обучения HTM, может легко включить в него временные интервалы и смоделировать работу отдельного слоя в коре головного мозга.

Слои 2 и 6

Из слоя 6 начинаются аксоны несущие обратную связь нижним по иерархии

областям коры. Гораздо меньше на сегодня известно о 2-ом слое неокортекса. Как уже упоминалось ранее, само существование 2-го уровня, отличающегося от уровня 3, иногда оспаривается некоторыми исследователями. Мы же ничего более не хотим сказать о слоях 2 и 6, за исключением того, что в них, как и в других слоях, отмечаются массовые горизонтальные связи между нейронами и свойство колончатого реагирования на стимулы, что дает нам право предположить, что в них также работает некоторый вариант кортикального алгоритма обучения HTM.

Чему соответствует регион HTM в неокортексе?

Мы описали реализацию кортикального алгоритма обучения HTM в двух вариантах, один со многими клетками в колонках, для реализации памяти переменного порядка, и второй с одной клеткой в колонке для памяти первого порядка. Мы предполагаем, что эти два варианта соответствуют слоям 3 и 4 в неокортексе мозга. Мы не пытались как-то скомбинировать эти два варианта в одном регионе HTM.

Хотя кортикальный алгоритм обучения HTM (со многими клетками в колонке) и очень близок к слою 3 в коре мозга, мы сохранили в нашей модели гибкость, которой нет в мозге. Следовательно, у нас есть возможность создавать гибридные слои нейронов, которые не будут соответствовать биологическим слоям в неокортексе. Например, в нашей модели мы знаем порядок, в котором синапсы формируются на дендритных сегментах. Мы можем использовать эту информацию для извлечения того, что предсказано произойти следующим, от всего того, что вообще предсказано в будущем. Наверное, подобным образом можно добавить в нашу модель и заданные интервалы времени. Следовательно, возможно создать один уровень региона HTM, который будет комбинировать в себе функциональность и слоя 3 и слоя 5.

Заключение

Наш кортикальный алгоритм обучения HTM воплощает в себе то, что мы считаем базовым строительным блоком всей нейронной организации в неокортексе. Он наглядно показывает, как слои горизонтально связанных между собой нейронов могут выделять и запоминать последовательности пространственно распределенных представлений входных паттернов. Различные варианты кортикального алгоритма обучения HTM предположительно используются в различных слоях неокортекса для разных, хотя и сходных между собой целей.

Мы предполагаем, что прямой вход сигналов в кортикальную область, в ее слой 3 или 4, псевдослучайно проектируется на проксимальные дендриты, которые, при помощи ингибиторных нейронов, создают пространственно распределенное представление входных сигналов. Мы также предполагаем, что клетки в слоях 2, 3, 4, 5 и 6 разделяют между собой эти пространственно распределенные представления. Это достигается тем, что все клетки в колонке на всех уровнях воспринимают один и тот же прямой входной сигнал.

Кроме того, мы предполагаем, что клетки 4-го слоя, когда он присутствует в области коры, используют подобие кортикального алгоритма обучения HTM для выявления и запоминания временных переходов между состояниями первого порядка, что дает нам представления, которые инвариантны для пространственных трансформаций. Клетки 3-го слоя используют кортикальный алгоритм обучения HTM для выявления и сохранения временных переходов переменного порядка и тем самым формируют стабильные представления, которые передаются выше по кортикальной иерархии. Клетки слоя 5 занимаются переходами переменного порядка с учетом временных интервалов. У нас нет конкретных предположений насчет слоев 2 и 6. Однако, учитывая их типичную горизонтальную связанность, весьма вероятно, что они также работают с одной из форм памяти последовательностей.

Глоссарий

Термины	**Расшифровка с *пояснениями*.** *Замечание: все определения терминов приводятся здесь согласно тому, как они употребляются в данном документе и могут отличаться от общепринятых значений. Слова с заглавной буквы указывают на другие термины из этого глоссария.*
Активное состояние	Состояние Клетки, когда она активна благодаря прямому входу.
Снизу-вверх	Синоним Прямого входа (ввода) сигналов [данный термин не использовался в данном переводе]
Клетки	HTM эквивалент Нейронов *Все Клетки организованы в Колонки внутри HTM региона.*
Совместная активность	Две или более клеток активны в одно и то же время.
Колонка	Группа из одной или более Клеток, которые функционируют совместно в HTM регионе *Клетки в одной колонке представляют один вход, но в разных контекстах.*
Дендритный Сегмент	Группа интегрированных Синапсов ассоциированные с Клетками и Колонками *В HTM есть два различных типа дендритных сегментов. Один ассоциирован с латеральными (боковыми) связями клеток. Когда число активных синапсов в дендритном сегменте превышает определенный порог, его клетка переходит в состояние предсказания. Другой тип сегмента ассоциирован с прямым входом для всей колонки. Число его активных синапсов суммируется для генерации прямой входной активации колонки.*

Желательная Плотность	Желательный процент Колонок активных благодаря их прямым входным данным. *Этот процент применяется только внутри радиуса, который изменяется в зависимости от входной области колонок. Она только «желательная» поскольку данный процент меняется при каждом конкретном входе.*
Прямой Вход	Данные (сигналы) поступающие непосредственно от входа всей HTM Сети или от нижележащего уровня HTM к вышележащему уровню.
Обратная связь	Данные передаваемы по направлению ко входу для всей HTM или от вышележащих уровня HTM к нижележащему (иногда их называют Сверху-вниз).
Предсказание первого порядка	Предсказание, основанное только на текущих входных данных, без учета всех предыдущих входов (контекста) – сравните с Предсказанием Переменного порядка.
Иерархическая Темпоральная память (HTM)	Технология, которая воспроизводит некоторые структурные и алгоритмические аспекты коры головного мозга.
Иерархия	Сеть связанных элементов, где все связи между элементами однозначно идентифицируются либо как прямые, либо как обратные.
Кортикальные Алгоритмы Обучения HTM	Набор функций Пространственного Группировщика и Темпорального Группировщика по обучению и забыванию, которые формируют HTM Регион, также называемые алгоритмами обучения HTM.
HTM Сеть	Иерархия Регионов HTM
HTM Регион	Основной блок памяти и предсказания в HTM *HTM регионы представляют из себя уровни плотно связанных клеток, организованных в колонки. На сегодня HTM регион имеет один уровень клеток, тогда как в коре мозга (и когда-нибудь в HTM) они имеют множество слоев клеток. При описании с точки зрения положения в Иерархии HTM регион может обозначаться как Уровень.*

Распознавание (Inference)	Распознавание пространственного и временного входного паттерна схожего с уже выученными ранее паттернами.
Радиус Подавления	Определяет область вокруг Колонки, которую она подавляет.
Латеральные связи	Связи между клетками внутри одного региона.
Уровень	HTM Регион в контексте его Иерархии.
Нейрон	Клетка мозга обрабатывающая информацию. *В данном документе мы обозначаем словом Нейрон именно биологические клетки мозга, а термин Клетки используем для обозначения вычислительных элементов в HTM.*
Перманентность	Скалярное (вещественное) число показывающее состояние соединения Потенциального Синапса *Значение перманентности ниже установленного порога показывает, что данный синапс не сформирован. А если оно выше этого порога, то синапс действует. Обучение в HTM регионе происходит путем изменения значений перманентности Потенциальных Синапсов.*
Потенциальный синапс	Подмножество всех Клеток, которые могут потенциально формировать Синапсы с конкретным Дендритным Сегментом. *Только часть из всех потенциальных синапсов будет действующими, основываясь на их значении перманентности.*
Предсказание (Предчувствие)	Активация Клеток (состояние предсказания), которые вероятно вскоре станут активными благодаря прямым входным данным. *HTM регион часто предсказывает множество возможных будущих входов в каждый момент времени.*

Рецепторное Поле	Множество входов к которым подключена Колонка или Клетка. *Если вход HTM Региона организован в виде двухмерного массива битов, тогда рецепторное поле можно определить как радиус в этом пространстве входа.*
Сенсор	Источник входных данных для всей Сети HTM
Разреженное Пространственное представление	Представление данных, состоящее из многих битов, из которых лишь малая часть активно и ни один из битов не достаточен для определения этих данных.
Пространственный Группировщик	Процесс формирования Разреженных Пространственных Представлений (Репрезентаций) входных данных. *Одно из их свойств состоит в том, что перекрывающиеся входные паттерны отображаются на одни и те же разреженные пространственные представления.*
Под-выборка	Распознавание большого распределенного паттерна путем сопоставления только небольшого подмножества активных битов хранимого паттерна.
Синапс	Соединение между Клетками, формируемое при обучении.
Темпоральный Группировщик	Процесс формирования последовательностей входных паттернов, при котором получающиеся представления более стабильны, чем исходные входные данные.
Сверху-вниз	Синоним Обратной связи.
Предсказание Переменного порядка	Предсказание, основанное на переменной глубине предыдущего контекста – сравните с Предсказанием первого порядка *Он называется «переменным» поскольку память для хранения предыдущего контекста выделяется по необходимости. Т.е. системы имплементирующие предсказание переменного порядка могут использовать контекст, уходящий значительно в прошлое, без задействования экспоненциальных объемов памяти.*