Numenta®

MEMORIA TEMPORAL JERÁRQUICA (HTM)

incluye

Algoritmos de Aprendizaje Cortical HTM

VERSIÓN 0.2.1, SEPTIEMBRE 12, 2011

Traducción de Garikoitz Lerma Usabiaga
garikoitz (at) gmail (dot) com

Numenta Translation License

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

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

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

Licencia de traducción de Numenta

El texto, algoritmos, códigos de ejemplo, pseudo-código y otros trabajos de este documento están basados o han sido traducidos del trabajo realizado sobre la tecnología de memoria temporal jerárquica (“HTM” en sus siglas en inglés) publicada por Numenta Inc. Numenta cuenta con el copyright de la obra original y las patentes relacionadas con HTM y los algoritmos traducidos. Numenta acuerda no hacer valer sus derechos sobre la patente en los desarrollos independientes de sistemas HTM, siempre y cuando el desarrollo o el uso sea para investigación y nunca para explotarlo comercialmente. Cualquier uso de la tecnología HTM con fines comerciales o de producción que infrinja las patentes de Numenta requerirá una licencia comercial de Numenta.

Con base en lo anterior, Numenta concede una licencia de uso de esta obra y los algoritmos para fines de investigación y no para uso comercial o de producción. Para el propósito de esta licencia, "el uso comercial o de producción" incluye el entrenamiento de una red HTM con la intención de más adelante hacer uso de la salida de la red entrenada en aplicaciones con fines comerciales o de producción, y/o permitir a otros utilizar la salida de la tecnología HTM con fines comerciales o de producción. La distribución, publicación, o copia de este trabajo debe incluir el texto completo de la licencia de traducción en Inglés y el idioma de destino.

ESTA LICENCIA NO OTORGA NINGUN DERECHO DE PATENTE, NI EXPRESAMENTE NI IMPLICITAMENTE. NUMENTA NIEGA ESPECÍFICAMENTE CUALQUIER RESPONSABILIDAD SOBRE LA CALIDAD O LA PRECISIÓN DE LA TRADUCCIÓN.

Lea esto primero!
Esta es una versión preliminar del documento. Faltan varias cosas de las que debería ser consciente.

Qué CONTIENE este documento:
Este documento describe en detalle nuevos algoritmos para el aprendizaje y la predicción desarrollados por Numenta. Los nuevos algoritmos han sido descritos con suficiente detalle para que un programador los pueda comprender e implementar si así lo desea. El documento comienza con un capítulo introductorio. Si ha estado siguiendo el desarrollo de Numenta y ha leído nuestros anteriores documentos publicados, el material en el capítulo introductorio le será familiar. El resto del material es nuevo.

Qué NO CONTIENE este documento:
Hay algunos temas relacionados con la implementación de estos nuevos algoritmos que no han podido entrar en este borrador preliminar.

- A pesar de que casi la totalidad de los aspectos de los algoritmos hayan sido implementados y testeados en software, ninguno de esos resultados han sido incluidos en el documento.

- No se incluyen descripciones de cómo se pueden aplicar estos algoritmos a problemas prácticos. No se incluye tampoco la descripción de cómo se podrían convertir los datos de un sensor o base de datos a una representación distribuida adecuada para los algoritmos.

- Los algoritmos son capaces de aprender en línea. No todos los detalles necesarios para implementar el aprendizaje en línea para algunos casos poco comunes han sido descritos.

- Otros añadidos planificados incluyen el desarrollo de las propiedades de las representaciones distribuidas dispersas, descripciones de aplicaciones y ejemplos, y citas para los apéndices.

Ponemos este documento a disposición en su forma actual porque pensamos que los algoritmos pueden ser de interés para el público. Las partes que faltan en el documento no deberían impedir a investigadores motivados comprender y experimentar con los algoritmos. Revisaremos este documento regularmente para reflejar el progreso.

Índice de Contenidos

Prefacio

Hay muchas tareas que son fáciles de realizar para los humanos pero que los ordenadores actuales no son capaces de hacer. Tareas como el reconocimiento de patrones visuales, entender el lenguaje hablado, reconocer y manipular objetos por el tacto y manejarse a través de un mundo complejo es fácil para los seres humanos. A pesar de los años de investigación, contamos con muy pocos algoritmos fiables para conseguir comportamientos similares en un ordenador.

En los humanos estas capacidades son generalmente desarrolladas en el neocórtex. La Memoria Temporal Jerárquica (HTM) es una tecnología modelada a la imagen de cómo el neocórtex ejecuta estas funciones. HTM ofrece la promesa de construir máquinas que se aproximen o superen el nivel humano para muchas tareas cognitivas.

Este documento describe la tecnología HTM. El Capítulo 1: proporciona una visión general de HTM, subrayando la importancia de la organización jerárquica, las representaciones distribuidas dispersas y el aprendizaje de las transiciones temporales. El Capítulo 2: descripción en detalle de los algoritmos de aprendizaje cortical HTM. Capítulos 3 y 4: proporcionan pseudocódigo para los algoritmos de aprendizaje HTM divididos en dos partes, agrupador espacial y agrupador temporal. Después de leer los capítulos 2 al 4, los desarrolladores de software avanzados deberían ser capaces de reproducir y experimentar con los algoritmos. Con suerte, algunos lectores irán más allá y extenderán nuestro trabajo.

Audiencia prevista

Este documento está enfocado a una audiencia con conocimientos técnicos. No se asumen conocimientos previos en neurociencias, pero sí se asume la comprensión de conceptos matemáticos e informáticos. Hemos escrito este documento de manera que pueda ser utilizado como lectura asignada en clase. El lector que hemos imaginado es un estudiante de informática o ciencias cognitivas, o un desarrollador de software que esté interesado en construir un sistema cognitivo artificial que funcione bajo los mismos principios que el cerebro humano.

Los lectores no técnicos se podrán beneficiar de ciertas partes del documento, en particular del *Capítulo 1: Visión general de HTM.*

Relación con la documentación previa

Partes de la teoría HTM ya fueron descritas en el libro *On Intelligence* del 2004, en otros documentos publicados por Numenta y en artículos científicos revisados escritos por empleados de Numenta. No hemos asumido que el lector haya leído estos trabajos con anterioridad, ya que la mayor parte de la información ha sido incorporada y actualizada en este documento. Se ha de tener en consideración que los algoritmos de aprendizaje descritos en los Capítulos 2-4 no habían sido publicados con anterioridad. Estos nuevos algoritmos sustituyen los algoritmos de primera generación denominados Zeta 1. Por un periodo corto de tiempo, llamamos los nuevos algoritmos "Representaciones Distribuidas de densidad fija" o "FDR" (según siglas en inglés), pero ya abandonamos esa terminología. Llamamos a los nuevos algoritmos los Algoritmos de Aprendizaje Cortical HTM, o a veces sólo los Algoritmos de Aprendizaje HTM.

Te animamos a que leas *On Intelligence*, escrito por el co-fundador de Numenta Jeff Hawkins junto con Sandra Blakeslee (en el siguiente enlace encontrarás la versión en castellano: http://www.numenta.com/htm-overview/education/onintelligence.php). A pesar de que el libro no mencione HTM con ese nombre, proporciona una explicación no técnica y fácil de leer de la teoría HTM y de la neurociencia que se encuentra detrás. En el momento en el que se escribía *On Intelligence* entendíamos los principios básicos inherentes a HTM pero no sabíamos como implementar esos principios algorítmicamente. Puedes pensar en este documento como una continuación del trabajo comenzado en *On Intelligence.*

Sobre Numenta

Numenta, Inc. (www.numenta.com) se formó en 2005 para desarrollar la tecnología HTM tanto para su uso comercial como para su uso científico. Para poder lograr este objetivo estamos documentando nuestro progreso y descubrimientos. También publicamos nuestro software de manera que otras personas lo puedan utilizar tanto para investigación como para desarrollo comercial. Hemos estructurado nuestro software con el objetivo de animar la creación de una comunidad de desarrolladores de aplicaciones independiente. La utilización del software y la propiedad intelectual es gratuita para la investigación. Generaremos ingresos vendiendo soporte, licenciando software, y licenciando propiedad intelectual para desarrollos comerciales. Siempre buscaremos el éxito de nuestros socios desarrolladores así como el nuestro propio.

Numenta está basada en Redwood City, California. Es de capital privado.

Sobre los autores

Este documento es un esfuerzo colaborativo entre los empleados de Numenta. Los nombres de los autores principales de cada sección están listados en el control de versiones.

Control de versiones

Anotamos en la siguiente tabla los cambios principales entre versiones. Los cambios menores tales como pequeñas clarificaciones o cambios de formato no son anotados.

Versión	Fecha	Cambios	Autores principales
0.1	Nov 9, 2010	1. Prefacio, Capítulos 1,2,3,4, y Glosario: primera versión	Jeff Hawkins, Subutai Ahmad, Donna Dubinsky
0.1.1	Nov 23, 2010	1. Capítulo 1: la sección Regiones fue editada para clarificar la terminología, tales como niveles, columnas y capas. 2. Apéndice A: primera versión	Hawkins & Dubinsky Hawkins
0.2	Dic 10, 2010	1. Capítulo 2: varias clarificaciones 2. Capítulo 4: referencias de línea actualizadas; cambios de código en líneas 37 y 39 3. Apéndice B: primera versión	Hawkins Ahmad Hawkins
0.2.1	Sep 12, 2011	1. Lea esto primero: eliminada referencia a 2010 2. Prefacio: eliminada la sección de versiones de software	

Capítulo 1: Visión General de la HTM

La Memoria Temporal Jerárquica (HTM) es una tecnología de aprendizaje de máquina que pretende capturar las propiedades estructurales y algorítmicas del neocórtex.

El neocórtex es la base del pensamiento inteligente en el cerebro de los mamíferos. La visión de alto nivel, escuchar, tocar, mover, el lenguaje y la planificación son llevadas a cabo por el neocórtex. Contando con esta diversidad de funciones cognitivas, se podría esperar que el neocórtex implementara la misma diversidad de algoritmos neuronales. Este no es el caso. El neocórtex muestra una gran uniformidad de patrones en su circuitería neuronal. La evidencia biológica sugiere que el neocórtex implementa un grupo común de algoritmos para llevar a cabo diferentes funciones relacionados con la inteligencia.

HTM proporciona el marco teórico para entender el neocórtex y sus múltiples capacidades. Hasta este momento hemos implementado un pequeño subgrupo de este marco teórico. Con el tiempo, más y más partes de la teoría serán implementadas. A día de hoy creemos que hemos implementado un subgrupo suficiente como para que pueda ser de utilidad comercial y científica.

Programar HTM-s es diferente a la programación en ordenadores tradicionales. Con los ordenadores actuales, los desarrolladores crean programas específicos para solucionar problemas específicos. Por contraste, los programas HTM son entrenados mediante la exposición a un flujo de datos sensoriales. En gran medida las capacidades del HTM son determinadas según a los datos a los que ha sido expuesto.

Los HTM pueden verse como un tipo de red neuronal. Por definición, cualquier sistema que trate de modelar los detalles arquitecturales del neocórtex es una red neuronal. De todas maneras, aisladamente, el término “red neuronal” no es de mucha utilidad ya que ha sido utilizado para una gran variedad de sistemas. Los HTM modelan neuronas (llamadas células en terminología HTM), las cuales son organizadas en columnas, capas, regiones y en una jerarquía. Los detalles son importantes, en este respecto los HTMs son una nueva forma de red neuronal.

Tal como indica su nombre, HTM es fundamentalmente un sistema basado en la memoria. Las redes HTM son entrenadas con muchos tipos de datos variados y confían en el almacenamiento de grandes grupos de patrones y secuencias. El modo en el que los datos son guardados y accedidos es lógicamente diferente del modelo estándar utilizado por los programadores. La memoria de ordenador clásica tiene una organización plana y no cuenta con ninguna noción inherente del tiempo. Un programador puede implementar cualquier tipo de organización y estructura de datos. Tienen control sobre cómo y dónde es guardada la información. Por contraste, la memoria HTM es más restrictiva. La memoria HTM tiene una organización jerárquica y es inherentemente basada en el concepto de tiempo. La

información siempre es guardada de manera distribuida. El usuario especifica el tamaño de la jerarquía y en qué entrenar al sistema, pero el HTM controla dónde y cómo se almacena la información.

A pesar de que las redes HTM son substancialmente diferentes a la informática clásica, podemos utilizar ordenadores de propósito general para modelarlos siempre y cuando se incorporen las funciones claves: la jerarquía, el tiempo y las representaciones distribuidas dispersas (serán descritos en detalle más adelante). Creemos que con el tiempo se creará hardware especializado con la capacidad de general redes HTM para propósitos específicos.

En este documento muchas veces ilustramos propiedades del HTM haciendo uso de ejemplos recogidos de la visión, tacto, oído, lenguaje y comportamiento humanos. Estos ejemplos son de utilidad porque son intuitivos y se entienden de manera fácil. De todas maneras, es importante tener en cuenta que las capacidades de las HTM son genéricas. Pueden ser fácilmente expuestas a flujos de datos de sensores no humanos, tales como radares o infrarrojos, o flujos puramente informacionales tales como datos del mercado financiero, del tiempo, patrones de tráfico Web o texto. Las HTMs son máquinas de aprendizaje y predicción que pueden ser aplicadas en muchos tipos de problemas.

Principios del HTM

En esta sección se cubrirán algunos de lo principios básicos del HTM: por qué la organización jerárquica es importante, como son estructuradas las regiones HTM, por qué los datos son guardados como representaciones distribuidas dispersas y por qué la información basada en el tiempo es crítica.

Jerarquía

Una red HTM consiste en regiones organizadas en una jerarquía. La región es la unidad principal de memoria y predicción en un HTM, y será discutida en detalle en la siguiente sección. Típicamente, cada región HTM representa un nivel en la jerarquía. A medida que se va ascendiendo la jerarquía siempre existe convergencia, esto es, múltiples elementos en una región hijo convergen en un elemento de una región padre. De todas maneras, debido a las conexiones de retroalimentación, la información también diverge a medida que desciendes en la jerarquía. (Una “región” o un “nivel” son casi sinónimos. Usamos la palabra “región” al describir la función interna de la región, y usamos la palabra “nivel” cuando nos referimos específicamente al rol de la región dentro de la jerarquía.)

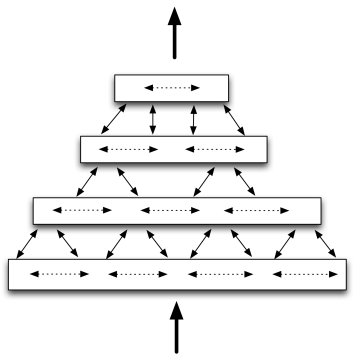


Figura 1.1: Diagrama simplificado de cuatro regiones HTM organizadas en una jerarquía de cuatro niveles, comunicando información dentro de cada nivel, entre niveles, y desde/hacia el exterior de la jerarquía

Es posible combinar múltiples redes HTM. Este tipo de estructura tiene sentido si se tienen datos de más de una fuente o sensor. Por ejemplo, una red podría estar procesando información auditiva y la otra red información visual. Existe convergencia dentro de cada red, pero en las redes separadas se juntan únicamente en la parte de arriba.

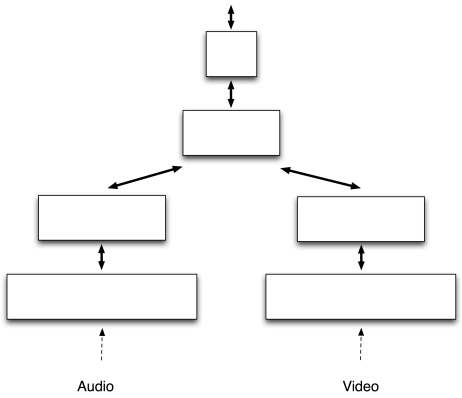


Figure 1.2: Redes de diferentes sensores convergiendo

El beneficio de la organización jerárquica es la eficiencia. Reduce significativamente el tiempo de entrenamiento y el uso de la memoria, ya que los patrones aprendidos en cada nivel de la jerarquía son reutilizados cuando se combinan de manera distinta en los niveles superiores. Utilizaremos la visión como ejemplo. En el nivel más bajo de la jerarquía, el cerebro almacena información sobre pequeñas secciones del campo visual tales como bordes y esquinas. Los bordes son un componente fundamental de muchos objetos en el mundo. Estos patrones de los niveles inferiores son recombinados en los niveles medios en componentes más complejos tales como curvas y texturas. Un arco puede ser el borde de una oreja, la parte de arriba de un volante o el asa de una taza de café. Estos patrones de los niveles

intermedios son recombinados también para representar características de objetos de alto nivel, tales como cabezas, coches o casas. Para aprender un nuevo objeto de alto nivel no hace falta reaprender sus componentes.

Otro ejemplo: al aprender una nueva palabra, no hace falta que reaprender las letras, sílabas o fonemas.

Compartir representaciones en una jerarquía conlleva conseguir la generalización del comportamiento esperado. Cuando ves un nuevo animal, si ves una boca y dientes, predecirás que el animal come con su boca y que puede ser que te muerda. La jerarquía permite que un nuevo objeto en el mundo herede las propiedades conocidas de sus subcomponentes.

¿Cuánto puede aprender un único nivel en una jerarquía HTM? Si lo preguntamos de otra manera: ¿cuántos niveles hacen falta en la jerarquía? Existe una solución de compromiso entre cuanta memoria es reservada a cada nivel y cuantos niveles son necesarios. Afortunadamente, los HTM aprenden automáticamente las mejores posibles representaciones en cada nivel conociendo las estadísticas de los datos de entrada y el número de recursos reservados. Si reservas más memoria a un nivel, ese nivel formará representaciones que son mayores y más complejas, lo cual significa que podrían hacer falta menos niveles jerárquicos. Si reservas menos memoria, el nivel formará representaciones que serán más pequeñas y simples, lo que significa que podrían ser necesarios más niveles jerárquicos.

Hasta ahora hemos estado describiendo problemas difíciles, tales como la inferencia visual (la “inferencia” es similar al reconocimiento de patrones). Pero muchos problemas de alto valor añadido son más simples que la visión, y una única región HTM podría ser suficiente. Por ejemplo, en Numenta aplicamos un HTM para predecir el enlace que seleccionaría un usuario navegando por una página web. Este problema significaba que se tendrían que cargar los datos de pulsaciones en la página. En este problema apenas había jerarquía espacial, la solución consistió en descubrir las estadísticas temporales, ej.: predecir que enlace seleccionaría el usuario a través del reconocimiento del patrón de uso de un usuario típico. Los algoritmos de aprendizaje temporales del HTM son ideales para problemas de este tipo.

Resumiendo, las jerarquías reducen el tiempo de entrenamiento, reducen el uso de memoria, e introducen una nueva forma de generalización. De todas maneras, muchos problemas simples que conllevan predicción pueden ser resueltos con una única región HTM.

Regiones

La noción de regiones ligadas dentro de una jerarquía viene de la biología. El neocórtex es una gran capa de tejido neuronal de unos 2mm de espesor. Los biólogos dividen el neocórtex entre diferentes áreas o “regiones”, principalmente

basándose en cómo se conectan estar regiones entre ellas mismas. Algunas regiones reciben la entrada de información directamente de los sentidos y otras regiones sólo lo hacen después de que ésta haya pasado antes por varias otras regiones. Es la conectividad entre las regiones la que define la jerarquía.

Todas las regiones neocorticales parecen similares en una observación de detalle. Varían sólo en tamaño y en su ubicación dentro de la jerarquía. Si cogiéramos un corte a través de los 2mm de una región neocortical, veríamos seis capas, 5 capas de células y una capa no celular (hay algunas excepciones pero esta es la regla general). Cada capa en una región neocortical tiene varias neuronas organizadas en columnas.

Las regiones HTM también están compuestas por una capa de células altamente concentradas en columnas. La "Capa 3" en el neocórtex es una de las principales capas neuronales de alimentación directa. Las celdas en una región HTM son equivalentes a las neuronas en la capa 3 neocortical.

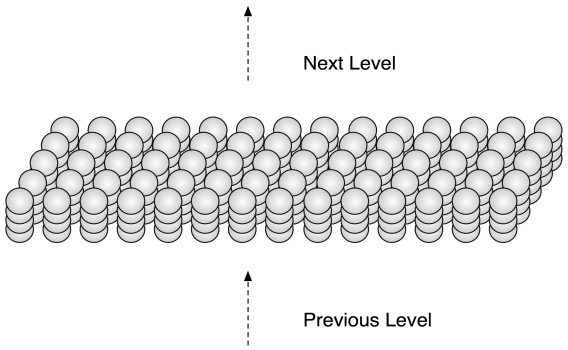


Figura 1.3: Sección de una región HTM. Las regiones HTM se componen de varias celdas. Las celdas están organizadas en una matriz bidimensional de columnas. Esta figura muestra una pequeña sección de una región HTM con 4 celdas por columna. Cada columna conecta con una parte de la entrada de datos y cada celda conecta con otras celdas en la región (las conexiones no se muestran). Notar que esta región HTM, incluyendo su estructura en columnas, es equivalente a una capa de neuronas neocorticales.

A pesar de que una región HTM equivale a una porción de la región neocortical, tiene la capacidad de inferir y predecir trabajando con complejos flujos de datos y por lo tanto puede ser útil en mucho problemas.

Representaciones Distribuidas Dispersas

A pesar de que las neuronas en el neocórtex estén altamente interconectadas, las neuronas inhibidoras garantizan que sólo un pequeño porcentaje de las neuronas estén activas en cada momento. Por lo tanto, la información en el cerebro es siempre representada por un pequeño porcentaje de neuronas activas. Este tipo de codificación se denomina "representación distribuida dispersa". "Dispersa" significa que sólo un pequeño porcentaje de neuronas están activas en cada momento.

“Distribuida” significa que las activaciones de muchas neuronas son requeridas para poder representar algo. Una única neurona activa transmite algo de sentido, pero se ha de interpretar dentro del contexto de otras neuronas para poder transmitir el sentido completo.

Las regiones HTM también utilizan las representaciones distribuidas dispersas. De hecho, los mecanismos de la memoria dentro de una región HTM dependen del uso de la representación distribuida dispersa, no podrían trabajar de otra manera. La entrada a una región HTM es siempre una representación distribuida, pero puede que no sea dispersa, por lo tanto lo primero que hace una región HTM es convertir estos datos de entrada en una representación distribuida dispersa.

Por ejemplo, una región puede recibir 20.000 bits de entrada. El porcentaje de bits de entrada que son “1” y “0” puede variar significativamente durante el tiempo. En un momento pueden haber 5.000 “1” bits y en otro momento pueden haber 9.000 “1” bits. La región HTM podría convertir esta entrada en una representación interna de 10.000 bits de los cuales el 2% o 200, están activos al mismo tiempo, independientemente del número que fueran “1”-s en la entrada. Ya que los datos de entrada a la región del HTM van variando con el tiempo, la representación interna también cambiará, pero siempre habrá aproximadamente unos 200 bits activos del total de 10.000.

Puede parecer que este proceso genera una gran perdida de información ya que el número posible de patrones de entrada es mucho mayor que el número posible de representaciones en la región. De todas maneras, los dos números son increíblemente grandes. Las entradas vistas por la región serán una fracción minúscula de todas las posibles entradas. Más adelante describiremos como una región crea una representación dispersa a partir de su entrada de datos. La pérdida de información teórica no tendrá ningún efecto práctico.

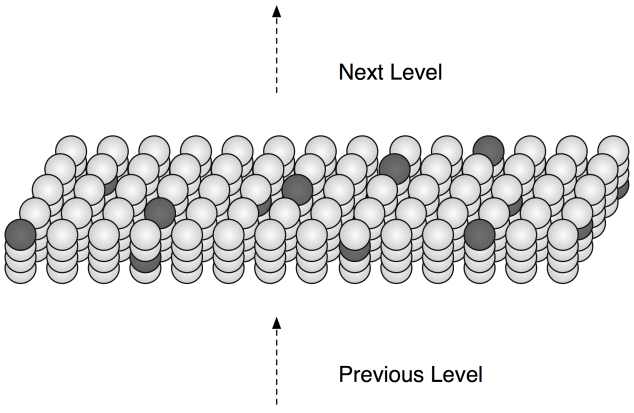


Figura 1.4: Una región HTM mostrando activación celular distribuida dispersa

Las representaciones distribuidas dispersas tienen varias propiedades deseables y son vitales para la correcta operación de los HTMs. Más adelante nos volveremos a centrar en ellos.

El rol del tiempo

El tiempo juega un rol crucial en el aprendizaje, inferencia y predicción.

Empecemos con la inferencia. Sin usar el tiempo, no podríamos inferir casi nada de nuestros sentidos táctiles y auditivos. Por ejemplo, si te vendaran los ojos y te colocaran una manzana encima de la mano, podrías identificar el objeto si lo pudieras mover durante un segundo aproximadamente. Mientras mueves los dedos sobre la manzana, a pesar de que la información táctil cambia constantemente durante el tiempo, el objeto en sí – la manzana, así como tu concepto de alto nivel para "manzana" – se mantiene constante. De todas maneras, si una manzana fuera colocada en tu mano estirada, y si no fueras capaz de mover tu mano o dedos, te sería mucho más difícil adivinar que era una manzana en lugar de un limón.

Ocurre lo mismo con el oído. Un sonido estático transmite poca información. Una palabra como "manzana", o el sonido al morderla, sólo puede ser reconocido desde las docenas de cientos de rápidos cambios secuenciales del espectro de sonido en el tiempo.

La visión, en contraste, es un caso mixto. Los humanos son capaces de reconocer imágenes cuando son mostradas tan rápidamente que al ojo no le de tiempo a moverse. De esta manera, la inferencia visual no siempre requiere de entradas que vayan cambiando durante el tiempo. De todas maneras, en una situación normal, estamos constantemente moviendo los ojos, cabeza y cuerpo, y los objetos alrededor también se mueven. Nuestra habilidad de inferir basándonos en la exposición visual rápida es un caso especial posibilitado gracias a las propiedades estadísticas de la visión y años de entrenamiento. Generalizando: tanto para la visión como para el oído y el tacto la inferencia requiere entradas que cambien durante el tiempo.

Habiendo cubierto ya el caso general para la inferencia, y el caso especial de la inferencia visual de imágenes estáticas, ahora nos centraremos en el aprendizaje. Para poder aprender, todos los sistemas HTM tienen que ser expuestos a entradas que varíen en el tiempo durante el entrenamiento. Incluso en el caso de la visión, donde la inferencia estática es a veces posible, tenemos que poder ver imágenes cambiantes de objetos para aprender que aspecto tiene un objeto. Por ejemplo, imaginemos que un perro está corriendo hacia ti. A cada instante en el tiempo el perro genera un patrón de actividad en tu retina. Percibes estos patrones como visiones diferentes del mismo perro, pero matemáticamente los patrones no se parecen para nada entre ellas. El cerebro aprende que esos patrones diferentes significan lo mismo viéndolos en secuencia. El tiempo es el "supervisor", enseñándote que patrones espaciales van juntos.

Hay que hacer notar que no es suficiente con que las entradas sensoriales vayan cambiando en el tiempo. Una sucesión de patrones sensoriales sin relación generaría sólo confusión. Las entradas que cambian con el tiempo tienen que venir de una misma fuente. Notar también que a pesar de que usemos sentidos o sensores humanos como ejemplos, el caso general aplica también a sensores no humanos. Si queremos entrenar un HTM para reconocer patrones de temperatura, vibración y ruido provenientes de sensores de una planta de producción de electricidad, el HTM tendrá que ser entrenado con datos de estos sensores cambiando en el tiempo.

Típicamente, una red HTM necesita ser entrenada con muchos datos. Aprendiste a identificar perros viendo muchas instancias de muchas razas de perros, no sólo una visualización de un perro en concreto. El trabajo de los algoritmos HTM es aprender las secuencias temporales de una serie de datos de entrada, ej.: construir un modelo para saber que patrones siguen a que patrones. El trabajo es difícil ya que puede ser que no sepa cuando empiezan y terminan las secuencias, pueden haber secuencias superpuestas ocurriendo al mismo tiempo, el aprendizaje tiene que ocurrir continuamente, y el aprendizaje tiene que ocurrir en la presencia de ruido.

Aprender y reconocer secuencias es la base de la formación de las predicciones. Una vez que un HTM aprende qué patrones son los que siguen a otros patrones, puede predecir los que se supone serán los siguientes conociendo la entrada actual y las inmediatamente anteriores. Trataremos la predicción con más detalle posteriormente.

Ahora pasaremos a ver las cuatro funciones básicas del HTM: aprendizaje, inferencia, predicción y comportamiento. Cada región HTM ejecuta las tres primeras: aprendizaje, inferencia y predicción. El comportamiento, por otro lado, es diferente. Sabemos por la biología que casi todas las regiones neocorticales tienen un rol creando comportamiento pero creemos que no es esencial para muchas aplicaciones interesantes. Por lo tanto no hemos incluido el comportamiento en nuestra implementación actual del HTM. Lo hemos mencionado en este apartado sólo para poder completar la descripción.

Aprendizaje

Una región HTM aprende sobre su mundo encontrando patrones y luego secuencias de patrones en la información sensorial. La región no “sabe” que representa la información de entrada, trabaja en un campo puramente estadístico. Busca combinaciones de entradas que ocurran juntas normalmente, lo que llamamos patrones espaciales. Entonces busca en qué secuencia aparecen estos patrones en el tiempo, lo que llamamos patrones temporales o secuencias.

Si la entrada a una región representa sensores medioambientales de un edificio, la región puede descubrir que ciertas combinaciones de temperaturas y humedades en el lado norte del edificio ocurren habitualmente y que en el lado sur del edificio ocurren combinaciones diferentes. Entonces podría aprender que secuencias de esas combinaciones ocurren a medida que los días van pasando.

Si la entrada a una región representa información relacionada con compras dentro de una tienda, la región HTM puede descubrir que ciertos tipos de artículos son comprados los fines de semana, o que cuando el tiempo es frío ciertos rangos de precios se ven favorecidos por las tarde. Entonces podría aprender que personas diferentes siguen patrones secuenciales similares en sus compras.

Una simple región HTM tiene una capacidad de aprendizaje limitada. Una región automáticamente ajusta lo que aprende basándose en cuanta memoria tiene y la complejidad de la entrada que está recibiendo. Los patrones espaciales aprendidos por una región necesariamente se simplificaran si la memoria asignada a una región ha sido reducida. Los patrones espaciales aprendidos se complicarán si la memoria asignada es incrementada. Si los patrones espaciales aprendidos por una región son simples, entonces la jerarquía de regiones puede ser necesaria para comprender imágenes complejas. Vemos este mismo comportamiento en el sistema visual humano, donde la región neocortical que recibe la entrada de información desde la retina aprende patrones espaciales para pequeñas partes del espacio visual. Sólo después de varios niveles de jerarquía se combinan los patrones espaciales y representan la casi totalidad del espacio visual.

Igual que un sistema biológico, los algoritmos de aprendizaje en una región HTM son capaces de “aprendizaje en tiempo real”, ej.: aprenden continuamente de cada nueva entrada. No existe la necesidad de una fase de aprendizaje separada de la fase de inferencia, a pesar de que la inferencia vaya mejorando con el entrenamiento. A medida de que los patrones en la entrada vayan cambiando, la región HTM cambiará gradualmente también.

Después del entrenamiento inicial, un HTM puede continuar aprendiendo o puede ser desactivado. Otra opción es desactivar el aprendizaje sólo en los niveles más bajos de la jerarquía pero continuar aprendiendo en los niveles superiores. Una vez un HTM ha aprendido la estructura estadística básica de su mundo, casi todo el aprendizaje nuevo ocurre en los niveles superiores de la jerarquía. Si un HTM es expuesto a nuevos patrones que contengan nuevas estructuras en los niveles bajos, le llevará más tiempo al HTM aprender estos nuevos patrones. También observamos esta característica en los seres humanos. Aprender nuevas palabras en un idioma que ya conoces es relativamente fácil. Pero, si tratas de aprender nuevas palabras de una lengua extranjera con sonidos a los que no estás acostumbrado, te será mucho más difícil.

Simplemente descubrir patrones puede ser una capacidad potencialmente valiosa. Entender los patrones de alto nivel en las fluctuaciones del mercado de valores, la

epidemiología, el tiempo, la producción industrial, o fallos en sistemas complejos, tales como redes eléctricas, puede ser valioso en sí mismo. En cualquier caso, aprender patrones espaciales y temporales suele ser un precursor de la inferencia y la predicción.

Inferencia

Una vez un HTM ha aprendido los patrones en su mundo, puede desarrollar la inferencia sobre entradas nuevas. Cuando un HTM recibe una entrada, lo mapeará a patrones espaciales y temporales aprendidos con anterioridad. Mapear satisfactoriamente nuevas entradas a secuencias guardadas previamente es la esencia de la inferencia y el mapeo de patrones.

Piensa en como reconoces una melodía. Escuchar sólo la primera nota dice poco. La segunda nota disminuye las posibilidades significativamente pero puede que no sea suficiente. Normalmente suelen hacer falta tres, cuatro o más notas antes de que reconozcas las melodía. La inferencia en una región HTM es similar. Está constantemente mirando a flujos de entrada y mapeándolos a secuencias aprendidas previamente. Una región HTM puede encontrar mapeos desde el principio de la secuencia o desde cualquier ubicación. Ya que las regiones HTM utilizan representaciones distribuidas, el uso que hace de la memoria secuencial y de la inferencia son más complicados que el ejemplo de la música, pero pensamos que los ejemplos dan una idea de cómo trabaja el sistema.

Puede que no sea obvio a primera vista, pero cada experiencia sensorial que has tenido ha sido siempre única, pero has sabido encontrar patrones comunes en cualquiera de estas experiencias únicas. Por ejemplo, puedes entender la palabra “desayuno” independientemente de quien la pronuncie, da igual su edad, sexo, si lo dicen rápido o lento, o si tienen un acento marcado o no. Aunque le pidas a la misma persona que diga la misma palabra “desayuno” cien veces, el sonido nunca estimulará tus cócleas (receptores auditivos) de la misma manera dos veces.

Una región HTM encara el mismo problema que tu cerebro: las entradas puede que no sean exactamente las mismas jamás. Consecuentemente, igual que tu cerebro, una región HTM debe manejar nuevas entradas durante la inferencia y el entrenamiento. Una de las maneras en las que una región HTM gestiona las nuevas entradas es mediante el uso de las representaciones distribuidas dispersas. Una propiedad clave de las representaciones distribuidas dispersas es que sólo hace falta que coincida una porción del patrón para estar seguros de que el mapeo es relevante.

Predicción

Cada región HTM almacena secuencias de patrones. Mapeando secuencias almacenadas con las nuevas entradas, una región forma una predicción sobre las entradas que vendrán a continuación. Las regiones HTM guardan transiciones entre representaciones distribuidas dispersas. En algunas instancias las transiciones pueden parecerse a secuencias lineales, como las notas de una melodía, pero en general varias posibles futuras entradas suelen ser predichas al mismo tiempo. Una región HTM predecirá diferentemente basado en el contexto que puede haber sido generado tiempo atrás. La mayoría de la memoria en un HTM se dedica a la memoria secuencial, guardando transiciones entre patrones espaciales.

A continuación se presentan algunas propiedades clave de la predicción HTM.

1) La predicción es continua
A pesar de no ser consciente, estamos todo el día prediciendo. Los HTMs hacen lo mismo. Cuando escuchas una canción, tratas de predecir la siguiente nota. Cuando estás bajando unas escaleras, estás prediciendo que tu pie tocará el siguiente escalón. Cuando ves lanzar a un lanzador de beisbol, predices que la pelota irá cerca del bateador. En una región HTM, la predicción y la inferencia son casi la misma cosa. La predicción no es un paso separado, es integral en la manera de trabajar de una región HTM.

2) La predicción ocurre en cada región y a cada nivel de la jerarquía
Si tienes una jerarquía de regiones HTM, la predicción ocurrirá a cada nivel. Las regiones harán predicciones sobre los patrones que han aprendido. En un ejemplo del lenguaje, los niveles inferiores pueden predecir los posibles siguientes fonemas, mientras que las regiones superiores pueden predecir palabras o frases.

3) Las predicciones son sensibles al contexto
Las predicciones están basadas en lo que ha ocurrido en el pasado, así como en lo que está ocurriendo en este mismo momento. De esta manera, una entrada producirá diferentes predicciones basado en el contexto previo. Una región HTM aprende a utilizar tanto contexto previo como el que le haga falta, y puede mantener el contexto tanto para periodos cortos o largos de tiempo. Esta habilidad es conocida como memoria de “orden variable”. Piensa en un discurso memorizado, por ejemplo la conocida como la Gettysburg Address. Predecir la siguiente palabra, conociendo únicamente la palabra anterior es raramente suficiente; la palabra “and” es seguida por “seven” y por “dedicated” sólo en la primera frase. A veces, sólo un poco de contexto ayudará a la predicción; saber “four score and” será suficiente para predecir “seven”. En otros casos, hay frases repetitivas, y hará falta el contexto en un periodo de tiempo más amplio para saber en que momento del discurso nos encontramos y poder predecir la siguiente palabra.

4) La predicción conduce a la estabilidad
La salida de una región es su predicción. Una de las propiedades de los HTMs es que las salidas de las regiones se estabilizan – esto es, cambian más despacio y aguantan más en el tiempo – cuando más altos estén en la jerarquía. Esta propiedad es resultado de la manera en que predice una región. Una región no predice que pasará inmediatamente después. Si puede, predecirá varios pasos en el futuro. Digamos, como ejemplo, que una región puede predecir cinco pasos en el futuro. Cuando llega una nueva entrada, la predicción justamente posterior puede cambiar pero los siguientes cinco no tienen porqué. Consecuentemente, a pesar de que cada nueva entrada es completamente diferente, sólo una parte de la salida se está modificando, haciendo las salidas más estables que las entradas. Esta característica copia nuestra experiencia en la vida real, donde los conceptos de alto nivel – por ejemplo el nombre de una canción – cambia más despacio que los conceptos de los niveles bajos – las notas en la canción.

5) Una predicción nos dice si una nueva entrada es esperada o inesperada
Cada región HTM es un detector de novedades. Ya que cada región predice qué va a ocurrir a continuación, "sabe" cuando algo inesperado ha ocurrido. Los HTMs pueden predecir muchas posibles siguientes entradas simultáneamente, no sólo una. Puede que no sea capaz de predecir exactamente que pasará a continuación, pero si la siguiente entrada no coincide con ninguna de las predicciones la región HTM sabrá que ha ocurrido una anomalía.

6) La predicción ayuda a hacer el sistema más robusto al ruido
Cuando un HTM predice lo que parece que ocurrirá, la predicción puede polarizar al sistema hacia inferir lo predicho. Por ejemplo, si un HTM estuviera procesando lenguaje hablado, predeciría qué sonidos, palabras e ideas serían las que se pronunciarían a continuación. Esta predicción ayuda al sistema a añadir los datos que falten. Si un sonido ambiguo es recibido, el HTM interpretará el sonido atendiendo a lo que estaba esperando, ayudando de esta manera a inferir incluso en la presencia de ruido.

En una región HTM, la memoria secuencial, la inferencia y la predicción están estrechamente integrados. Son funciones principales de una región.

Comportamiento

Nuestro comportamiento condiciona lo que percibimos. En la medida que movemos nuestros ojos, nuestra retina recibe información sensorial cambiante en el tiempo. Mover nuestras manos y dedos causa variación en las sensaciones del tacto. Casi cualquier acción produce cambios en lo que sentimos. La información sensorial y el comportamiento motor están estrechamente relacionados.

Las últimas décadas la visión principal consistía en que una única región en el neocórtex, la corteza motora primaria, era donde se generaban las órdenes motoras. Se ha ido descubriendo que casi todas las regiones del neocórtex envían órdenes motoras, incluso en los niveles sensoriales bajos. Parece que todas las regiones de la corteza cerebral integran funciones sensoriales y motoras.

Esperamos que se podrá añadir una salida motora a cada región HTM dentro de la arquitectura tecnológica existente, ya que generar órdenes motoras es similar a hacer predicciones. De todas maneras, hasta la actualidad todas las implementaciones de HTM han sido puramente sensoriales, sin ninguna componente motora.

Progreso hacia la implementación de HTM

Hemos realizado avances significativos en el proceso de convertir el marco teórico HTM en una tecnología práctica. Hemos implementado y testeado varias versiones de los algoritmos de aprendizaje cortical HTM y hemos encontrado que la arquitectura base es sólida. A medida que testeemos los algoritmos en nuevos grupos de datos, los refinaremos y añadiremos las piezas que falten. Iremos actualizando este documento en paralelo. Los siguientes tres capítulos describen el estado de avance actual de los algoritmos.

Hay muchos componentes de la teoría que no han sido implementados todavía, como la atención, la retroalimentación entre regiones, la temporización específica y la integración comportamental/sensomotora. Estos componentes que faltan deberían poder ser integrados en la arquitectura ya creada.

Capítulo 2: Algoritmos de Aprendizaje Cortical HTM

Este artículo describe los algoritmos de aprendizaje dentro de una región HTM. Los Capítulos 3 y 4 describen la implementación de los algoritmos de aprendizaje utilizando pseudocódigo; este capítulo es más conceptual.

Terminología

Antes de empezar, una nota sobre la terminología será de ayuda. Utilizamos el lenguaje neurocientífico a la hora de describir los algoritmos de aprendizaje HTM. Términos como células, sinapsis, sinapsis potenciales, segmentos dendríticos y columnas son utilizados a lo largo del documento. Esta terminología es lógica considerando que los algoritmos de aprendizaje han sido derivados de detalles neurocientíficos mapeados a las necesidades teóricas. De todas maneras, en el proceso de implementar los algoritmos nos enfrentamos a problemas de rendimiento, por lo que en el momento que sentíamos que entendíamos como funcionaba algo buscábamos la manera de acelerar el procesamiento de datos. Esto a veces ha significado alejarnos de la biología siempre y cuando pudiéramos obtener los mismos resultados. Si eres nuevo en las neurociencias este tema no debería ser un problema. Pero, si estás familiarizado con los términos neurocientíficos, podrás sentirte confundido ya que el uso que hacemos de la terminología variará de tus expectativas. En los apéndices sobre biología desarrollamos las diferencias y parecidos entre los algoritmos de aprendizaje HTM y sus equivalentes neurobiológicos en detalle. Ahora sólo mencionaremos algunas de las desviaciones que puedan causar más confusión.

Estados de la célula
Las células HTM tienen tres estados de salida: activo por una entrada de alimentación, activo por una entrada lateral (representa una predicción) e inactivo. El primer estado de salida corresponde a una ráfaga corta de potenciales de acción en una neurona. El segundo estado de salida corresponde a potenciales de acción más lentos y a un ritmo constante en la neurona. No hemos encontrado la necesidad de modelar potenciales de acción individuales o incluso tasas de actividad escalares más allá de estos dos estados activos. El uso de representaciones distribuidas parece que salva la necesidad de modelar las tasas de actividad escalares.

Segmentos dendríticos
Las células HTM tienen un modelo dendrítico relativamente realista (y por lo tanto complejo). En teoría cada célula HTM tiene un segmento dendrítico proximal y una o dos docenas de segmentos dendríticos distales. El segmento dendrítico proximal recibe la alimentación y los segmentos dendríticos distales reciben entradas laterales desde las células cercanas. Un tipo de células inhibidoras fuerza todas las células de una columna a responder a entradas de alimentación similares. Para simplificar, hemos quitado el segmento dendrítico proximal de cada célula y lo

hemos sustituido por un segmento dendrítico compartido por cada columna de células. La función del agrupador espacial (descrito abajo) opera sobre el segmento dendrítico compartido, al nivel de las columnas. La función del agrupador temporal opera sobre los segmentos dendríticos distales, al nivel de las células individuales dentro de las columnas. Esta simplificación consigue la misma funcionalidad, a pesar de que en biología no exista un equivalente para el segmento dendrítico para una columna.

Sinapsis
Las sinapsis HTM toman dos valores binarios. Las sinapsis biológicas tienen valores variados pero también son parcialmente estocásticas, esto sugiere que una neurona biológica no puede fiarse de valores sinápticos precisos. El uso de las representaciones distribuidas en HTM, además de nuestro modelo de operación dendrítica, nos permite asignar valores binarios a las sinapsis HTM sin efectos perniciosos. Para modelar la creación y la destrucción de sinapsis usamos dos conceptos adicionales de las neurociencias con los que puede ser que no estés familiarizado. Uno es el concepto de "sinapsis potencial". Representa todos los axones que pasan suficientemente cerca de un segmento dendrítico, de manera que podrían potencialmente formar una sinapsis. El segundo concepto es llamado "permanencia". Es un valor escalar asignado a cada sinapsis potencial. La permanencia de una sinapsis representa el grado de conexión entre un axón y una dendrita. En biología, el rango iría desde completamente desconectado a empezando a formar la sinapsis pero sin conexión todavía, a sinapsis conectada mínimamente, a sinapsis conectada completamente. La permanencia de una sinapsis es un valor escalar que va desde el 0,0 al 1,0. Aprender consiste en incrementar o disminuir la permanencia de una sinapsis. Cuando la permanencia de una sinapsis supera un umbral, está se considera conectada con un valor de "1". Cuando está por debajo de este umbral, está desconectada con un valor de "0".

Visión general

Imagina que eres una región de un HTM. Tus entradas consisten en miles o decenas de miles de bits. Estas entradas pueden representar datos sensoriales o pueden venir desde otra región inferior de la jerarquía. Se encienden y apagan de manera muy compleja. ¿Qué se supone que tendrías que hacer con esta entrada?

Ya hemos comentado la respuesta en su forma más simple. Cada región HTM evalúa su entrada y busca patrones comunes y a continuación aprende secuencias de esos patrones. A partir de su memoria de secuencias, cada región hace predicciones. Esta descripción a alto nivel puede parecer muy sencilla, pero realmente hay muchas tareas que ocurren a la vez. Podemos dividir las tareas principales de la siguiente manera:

1) Formar una representación distribuida de los datos de entrada

2) Formar una representación de la entrada sobre el contexto de las entradas previas
3) Formar una predicción basada en la entrada actual y el contexto de las entradas previas

Vamos a ver cada una de estas etapas en detalle.

1) Formar una representación distribuida dispersa de los datos de entrada
Cuando imagines la entrada a una región, piensa en un gran número de bits. En el cerebro serían axones de neuronas. En cualquier momento en el tiempo algunas de estas entradas estarán activas (valor 1) y otras inactivas (valor 0). El porcentaje de bits de entrada que está activo varia, digamos que entre el 0% y el 60%. La primera acción que acomete la región HTM es convertir esta entrada en una nueva representación que sea dispersa. Por ejemplo, la entrada podría tener el 40% de los datos de entrada en "on" pero la nueva representación tendrá sólo el 2% de sus bits en "on".

Una región HTM está compuesta a nivel lógico por un grupo de columnas. Cada columna está compuesta por una o varias células. Las columnas se pueden ordenar lógicamente en una matriz 2D, pero no es un requerimiento. Cada columna en una región de una región está conectada a un subgrupo único de bits de entrada (normalmente superponiéndose a otras columnas pero nunca exactamente el mismo subgrupo de bits de entrada). Como resultado, diferentes patrones de entrada resultan en distintos niveles de activación de las columnas. Las columnas con la activación más fuerte inhiben, o desactivan, las columnas con las activaciones más débiles. (La inhibición ocurre dentro de un radio que puede ir desde muy local hasta una región entera). La representación distribuida de la entrada será codificada según las columnas que están activas y las que son inactivas después de la inhibición. La función de inhibición está definida para lograr que un porcentaje relativamente pequeño de columnas quede activa, incluso cuando el número de bits de entrada que es activo varíe significativamente.

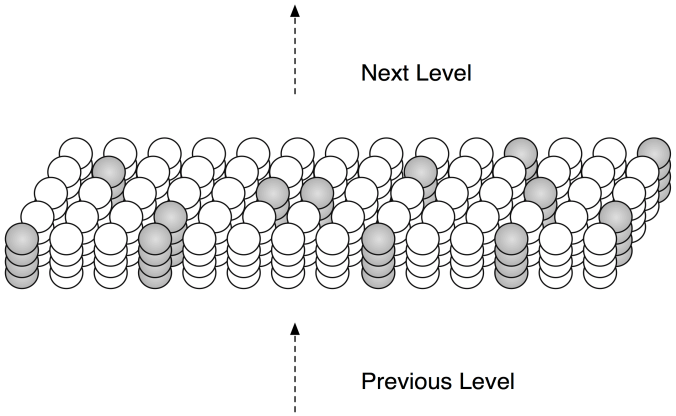


Figura 2.1: Una región HTM consiste en columnas de células. Sólo una pequeña porción de la región es mostrada. Cada columna de células recibe la activación desde un subgrupo único de datos de entrada. Las columnas más activadas inhiben las columnas con menos activación. El resultado es una representación distribuida dispersa de la entrada. La figura muestra las columnas activas en gris claro. (Cuando no existe un estado previo, cada célula en la columna activa estará activa, tal como se muestra en la figura).

Imaginemos ahora que el patrón de entrada cambia. Si sólo unos cuantos bits de entrada cambian, algunas de las columnas recibirán algunas menos o algunas más en el estado "on", pero el grupo de columnas activas no debería cambiar mucho. Por lo tanto, los patrones de entrada similares (los que tienen muchos bits activos en común) mapearan un grupo bastante estable de columnas activas. La estabilidad de la codificación depende en gran medida a las entradas a las que esté conectada cada columna. Estas conexiones serán aprendidas con un método descrito más adelante.

Todos estos pasos (aprender las conexiones para cada columna desde un subgrupo de entradas, determinar el nivel de entradas de cada columna, y usar la inhibición para seleccionar un grupo disperso de columnas activas) conforman el "Agrupador Espacial". El termino indica que los patrones que son "espacialmente" similares (significa que comparten un gran número de bits activos) están "agrupados" (significa que son agrupados en una representación común).

2) Formar una representación de la entrada en el contexto de las entradas previas

La siguiente funcionan que ejecuta la región es convertir la representación en columnas de la entrada en una nueva representación que incluya el estado, o contexto, del pasado. La nueva representación es formada activando un subgrupo de células dentro de cada columna, típicamente sólo una célula por columna (Figura 2.2).

Consideremos estas dos frases habladas, "I ate a pear" y "I have eight pears". Las palabras "ate" y "eight" en inglés son homónimas; el sonido es idéntico. Podemos estar seguros de que en algún momento en el cerebro hay neuronas respondiendo idénticamente a los sonidos "ate" y "eight". Después de todo, los sonidos que están

entrando en el cerebro son idénticos. También podemos estar seguros de que en otro lugar del cerebro las neuronas que responden a esta entrada son diferentes, debido a los diferentes contextos. La representación para el sonido “ate” será diferente cuando escuchas “I ate” que cuando escuchas “I have eight”. Imagina que has memorizado las dos frases “I ate a pear” y “I have eight pears”. Escuchar “I ate…” llevará a una predicción diferente que “I have eight…”. Tienen que existir representaciones internas diferentes para el caso de “I ate” y “I have eight”.

Este principio de codificar una entrada de manera diferente en contextos diferentes es una cualidad universal de la percepción y de la acción y es una de las funciones más importantes de una región HTM. Es difícil exagerar la importancia de esta capacidad.

Cada columna en una región HTM es constituida por varias células. Todas las células en una columna obtienen la misma entrada de alimentación. Cada célula en cada columna puede estar activa o no activa. Seleccionando diferentes células activas en cada columna activa, podemos representar exactamente la misma entrada de maneras distintas en contextos distintos. Un ejemplo concreto puede ser de ayuda. Digamos que cada columna tiene cuatro células y la representación de cada entrada consiste en 100 columnas activas. Si sólo una célula por columna es activada en cada momento, obtendremos 4^100 maneras de representar esa misma entrada. La misma entrada resultará siempre en esas 100 columnas activas, pero en diferentes contextos diferentes células de esas mismas columnas estarán activas. Ahora ya tenemos la capacidad de representar esa misma entrada en muchos contextos diferentes, ¿Cómo de únicas serán cada una de esas representaciones? Aproximadamente cada par seleccionado aleatoriamente de los 4^100 patrones posibles se superpondrá por unas 25 células. De esta manera dos representaciones de una entrada particular pero en contextos diferentes tendrá unas 25 células en común y 75 células que son diferentes, haciéndolas muy fáciles de diferenciar.

La regla general utilizada por una región HTM es la siguiente. Cuando una columna se activa, mira en todas las células de la columna. Si una o más células de la columna están ya en un estado predictivo, sólo esas células pasan a activarse. Si en la columna no había células en estado predictivo, entonces todas las células se activan. Se puede pensar de la siguiente manera: si un patrón de entrada era esperado entonces el sistema confirma esa expectativa activando sólo las células en el estado predictivo. Si el patrón de entrada es inesperado, entonces el sistema activa todas las células en la columna como diciendo “la entrada ha ocurrido inesperadamente así que cualquier posible interpretación es válida”.

Si no existe un estado previo, y por lo tanto ni contexto ni predicción, todas las células en la columna se activarán cuando ésta se active. Este escenario es similar a escuchar la primera nota de una canción. Sin contexto no podrías predecir que vendría a continuación; todas las opciones están abiertas. Si hay un estado previo pero la entrada no concuerda con lo esperado, toda las células en la columna activa

se activarán. Esta actividad se determina columna a columna, de esta manera una concordancia o no concordancia nunca es un evento “todo o nada”.

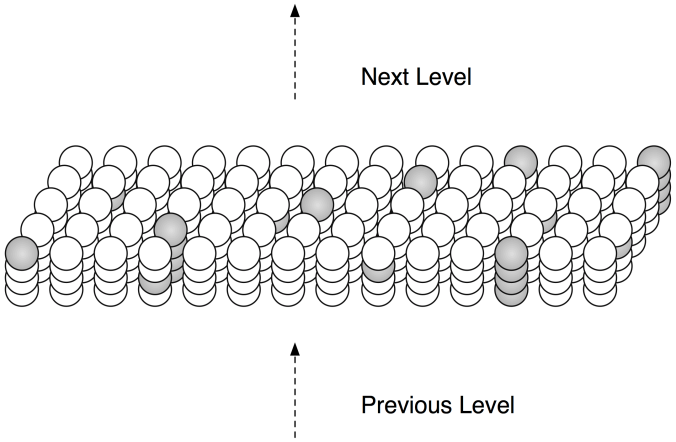


Figura 2.2: Activando un subgrupo de células en cada columna, una región HTM puede representar la misma entrada en muchos contextos diferentes. Las columnas sólo activan las células que ya estaban en estado de predicción. Las columnas sin células en estado de predicción activan todas las células en la columna. La figura muestra algunas de las columnas con una célula activa y otras columnas con todas las células activas.

Tal como se mencionó anteriormente en la sección de terminología, las células HTM pueden tener tres estado diferentes. Si una célula está activa debida a la entrada de alimentación, usaremos el término “activo”. Si la célula está activa debido a las conexiones laterales a otras células cercanas diremos que está en “estado predictivo” (Figura 2.3).

3) Formar una predicción basada en la entrada actual y el contexto de las entradas previas

El último paso para nuestra región es predecir lo que previsiblemente ocurrirá después. La predicción se basa en la representación formada en la etapa 2), la cual incluye contexto de todas las entradas anteriores.

Cuando una región predice, activa (pasa al estado predictivo) todas las células que presumiblemente se activarán debido a la entrada de alimentación futura. Ya que todas las representaciones en una región son distribuidas, se pueden realizar múltiples predicciones al mismo momento. Por ejemplo, si el 2% de las columnas están activas debido a una entrada, se pueden esperar 10 predicciones distintas, con un 20% de las columnas conteniendo células en estado predictivo. O se podrían dar 20 predicciones distintas, con 40% de las columnas conteniendo una célula en estado predictivo. Si cada columna tuviera 4 células, con una activa en cada momento, entonces el 10% de las células estarían en estado predictivo.

En un próximo capitulo dedicado a las representaciones distribuidas dispersas veremos que aunque las predicciones se fusionen entre ellas, una región puede saber con altas probabilidades si una entrada en concreto fue predicha o no.

¿Cómo predice una región? Cuando los patrones de entrada cambian en el tiempo, diferentes grupos de columnas y células se activan en secuencia. Cuando una celda se activa, forma conexiones a un subgrupo de células cercanas que estaban activas en el momento inmediatamente anterior. Estas conexiones se pueden formar rápido o lento dependiendo de la velocidad de aprendizaje requerida por la aplicación. A continuación todo lo que tiene que hacer la célula es mirar a todas esas conexiones para encontrar actividad coincidente. Si las conexiones se activan, la célula puede esperar su próxima activación y entra en estado predictivo. La activación por alimentación de un grupo de células llevará a la activación predictiva de otros grupos de células. Se ha de pensar en este efecto como el momento en el que se reconoce una canción y se empiezan a predecir las siguientes notas.

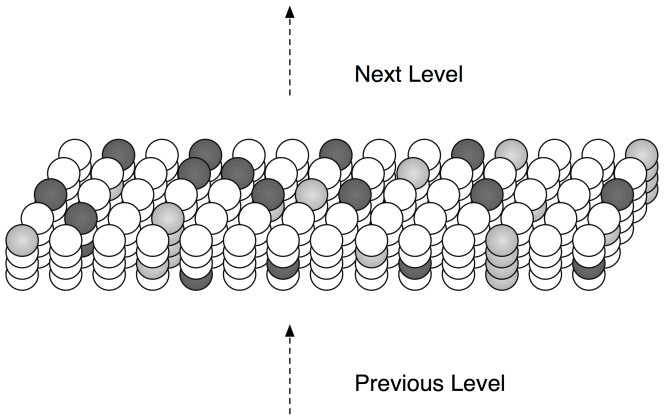


Figura 2.4: En cualquier momento en el tiempo, algunas células en una región HTM estarán activas debido a la entrada de retroalimentación positiva (mostrados en gris claro). Otras células que reciben entradas laterales de células activas estarán en estado predictivo (mostrados en gris oscuro)

Resumiendo, cuando una nueva entrada llega, conlleva la activación distribuida de columnas. Una o varias de las células en cada columna se activa, y éstas a su vez causan otras células a entrar en un estado predictivo a través de las conexiones aprendidas entre las células de la región. Las células activadas por conexiones dentro de la región constituyen una predicción de lo que probablemente ocurrirá a continuación. Cuando la siguiente entrada de alimentación llegue, seleccionará otro grupo de columnas activas. Si una columna que haya sido activada no se había predicho por ninguna célula, activará todas las células en la columna. Si una columna activada tenía una o más células en estado predictivo, sólo esas células pasarán a estar activas. La salida de una región es resultado de la actividad de todas las células en la región, incluyendo las células activas por la alimentación y las células activas en estado predictivo.

Tal como se ha mencionado, las predicciones no son sólo para el siguiente paso en el tiempo. Las predicciones en una región HTM pueden ser para varios pasos en el futuro. Usando melodías como ejemplo, una región HTM no sólo predeciría la siguiente nota, sino las 4 siguientes. Esta manera de funcionar proporciona una propiedad muy útil. La salida de una región (la unión de todas las células activas y predictivas en la región) cambia más despacio que la entrada. Imaginemos que la región está prediciendo las cuatro siguientes notas en una melodía. Representaremos la melodía con la secuencia de letras A, B, C, D, E, F, G. Después de escuchar las dos primeras notas, la región reconoce la secuencia y empieza a predecir. Predice C,D,E,F. Las células "B" están ya activas así que las células para B,C,D,E,F estarán en cualquiera de las dos formas activas. Ahora la región escucha la nota "C". El grupo de células activas y predictivas ahora representa "C,D,E,F,G". El patrón de entrada ha cambiado completamente yendo de "B" a "C", pero sólo el 20% de las células han cambiado.

Ya que la salida de una región HTM es un vector representando la actividad de todas las células de la región, la salida en este ejemplo es cinco veces más estable que la entrada. En una organización jerárquica de regiones, veremos un incremento en estabilidad temporal a medida que ascendemos en la jerarquía.

Utilizamos el término "agrupador temporal" para describir los dos pasos correspondientes a añadir contexto a la representación y a la predicción. Mediante la creación de salidas para las secuencias de patrones que cambian lentamente, en esencia estamos "agrupando" diferentes patrones que se siguen en el tiempo.

Ahora bajaremos a otro nivel de detalle. Empezaremos con los conceptos que son compartidos por el agrupador espacial y temporal. Después veremos conceptos y detalles específicos del agrupador espacial seguidos de conceptos y detalles específicos del agrupador temporal.

Conceptos compartidos

El aprendizaje en el agrupador espacial y temporal es similar. En los dos casos aprender significa establecer conexiones, o sinapsis entre las células. El agrupador temporal aprende sobre las conexiones entre las células de la misma región. El agrupador espacial aprende sobre la entrada de alimentación entre los bits de entrada y las columnas.

Pesos binarios

Las sinapsis HTM sólo tienen un efecto de 0 o 1: su "peso" es binario, a diferencia de otros modelos de redes neuronales en el que utilizan valores con un rango entre 0 y 1.

Permanencia
Las sinapsis se forman y se deshacen constantemente durante el aprendizaje. Tal como se ha mencionado antes, asignamos un valor escalar a cada sinapsis (0,0 a 1,0) para indicar la permanencia de la conexión. Cuando una conexión se ve reforzada, su permanencia es incrementada. Bajo otras condiciones, la permanencia se ve disminuida. Cuando la permanencia está por encima de un umbral (ej. 0,2), la sinapsis se considera establecida. Si la permanencia está por debajo de este umbral, la sinapsis no tendrá efecto.

Segmentos dendríticos
Las sinapsis se conectan a segmentos dendríticos. Hay dos tipos de segmentos dendríticos, proximales y distales.
- Un segmento dendrítico proximal forma sinapsis con las entradas de alimentación. Las sinapsis activas en este tipo de segmento son sumadas linealmente para determinar la activación por la alimentación de una columna.
- Un segmento dendrítico distal forma sinapsis con células dentro de la región. Cada célula cuenta con varios segmentos dendríticos distales. Si la suma de todas las sinapsis activas en un segmento distal excede un umbral, entonces la célula asociada se activa en estado predictivo. Ya que hay múltiples segmentos dendríticos distales por célula, el estado predictivo de una célula funciona como una operación lógica OR constituida por varios detectores de umbrales.

Sinapsis potenciales
Tal como se ha mencionado con anterioridad, cada segmento dendrítico cuenta con una lista de sinapsis potenciales. Todas las sinapsis potenciales tienen un valor de permanencia y pueden convertirse en sinapsis funcionales si su valor de permanencia excede un cierto umbral.

Aprendizaje
Aprender implica incrementar o disminuir los valores de permanencia de las sinapsis potenciales en un segmento dendrítico. Las reglas utilizadas para hacer las sinapsis más o menos permanentes son similares a la reglas de aprendizaje "Hebbianas". Por ejemplo, si una célula post sináptica está activa debido a que el segmento dendrítico esté recibiendo entradas por encima de su umbral, en ese caso los valores de permanencia de las sinapsis en ese segmento son modificados. Las sinapsis que estén activas, y por lo tanto que contribuyan a la célula a estar activa, ven su permanencia incrementada. Sinapsis inactivas, que por lo tanto no han contribuido, ven su permanencia disminuida. Las condiciones exactas bajo las cuales los valores de las permanencias de las sinapsis son actualizadas difiere en el agrupador temporal o espacial. Los detalles se describen debajo.

Ahora veremos conceptos específicos a las funciones de agrupador espacial y agrupador temporal.

Conceptos del agrupador espacial

La función fundamental del agrupador espacial es convertir la entrada a una región en un patrón disperso. Esta función es importante, ya que el mecanismo utilizado para aprender y predecir las secuencias requiere empezar con patrones distribuidos dispersos.

El agrupador espacial tiene varios objetivos superpuestos, los cuales determinan como opera y aprende.

1) Utilizar todas las columnas

Una región HTM tienen un número fijo de columnas que aprenden a representar patrones comunes. Uno de los objetivos es asegurar que todas las columnas aprendan a representar algo útil independientemente del número de columnas que tengas. No queremos tener columnas que no se activen nunca. Para prevenir que ocurra, mantenemos un registro de cuantas veces se ha activado una columna respecto a sus vecinas. Si la actividad relativa de una columna es muy baja, se estimula su nivel de actividad en la entrada hasta que empieza a tomar parte en el grupo de columnas activas. En esencia, todas las columnas están compitiendo con sus vecinas para ser partícipes en las representaciones de los patrones de entrada. Si una columna no es muy activa, se convertirá en más agresiva. Cuando lo hace, las otras columnas se verán forzadas a modificar su entrada y empezar a representar patrones de entrada ligeramente diferentes.

2) Mantener la densidad deseada

Una región necesita formar un representación distribuida de su entrada. Las columnas con el mayor número de entradas inhiben a sus vecinas. Existe un radio de inhibición proporcional al tamaño de los campos receptores de las columnas (y por lo tanto van desde muy pequeñas hasta el tamaño de toda la región). Dentro del radio de inhibición, permitimos sólo a un porcentaje de las columnas con la entrada más activa ser las “ganadoras”. Las columnas restantes son deshabilitadas. (Un “radio” de inhibición implica una organización de columnas 2D, pero el concepto se puede adaptar a otras topologías).

3) Evita los patrones triviales

Queremos que todas las columnas representen patrones no triviales en la entrada. El objetivo se puede alcanzar estableciendo un umbral mínimo para que la columna se active. Por ejemplo, si establecemos el umbral en 50, significa que una columna necesita tener al menos 50 sinapsis activas en su segmento dendrítico para estar activo, garantizando un cierto nivel de complejidad del patrón que está representando.

4) Evita conexiones extras
Si no somos cuidadosos, una columna podría formar demasiadas sinapsis válidas. Entonces respondería fuertemente a muchos patrones de entrada no relacionados. Diferentes subgrupos de sinapsis responderían a patrones diferentes. Para evitar este problema, disminuimos el valor de permanencia de cualquier sinapsis que no esté contribuyendo a una columna ganadora. Asegurando que las sinapsis que no contribuyen son suficientemente penalizadas, garantizamos que una columna represente un número limitado de patrones de entrada, a veces sólo una.

5) Campos receptivos auto ajustables
Los cerebros reales son altamente "plásticos": las regiones del neocórtex pueden aprender a representar realidades completamente distintas en reacción a los cambios. Si parte del neocórtex ha sido dañada, otras partes se ajustarán para representar lo que la parte dañada solía representar. Si un órgano sensorial ha sido dañado o cambiado, la parte asociada del neocórtex se ajustará para representar cualquier otra cosa. El sistema se auto ajusta.

Queremos que las regiones HTM muestren la misma flexibilidad. Si reservamos 10.000 columnas para una región, debería aprender la mejor manera de representar esa entrada con esas 10.000 columnas. Si reservamos 20.000 columnas, debería aprender como hacer el mejor uso posible de ese número. Si las estadísticas de la entrada cambian, las columnas deberían cambiar para representar de la mejor manera posible la nueva realidad. Resumiendo, el diseñador de una HTM debería poder reservar cualquier número de recursos a una región y la región hará su mejor trabajo para representar la realidad con las columnas disponibles y las estadísticas de entrada. La regla general es que con más columnas en una región, cada columna representará patrones más grandes y más detallados. Habitualmente las columnas estarán activas menos veces, y mantendremos un nivel de dispersión relativamente constante.

No hacen falta nuevas reglas de aprendizaje para lograr este objetivo tan deseado. Estimulando las columnas inactivas, inhibiendo las columnas vecinas para mantener el nivel de dispersión constante, estableciendo umbrales mínimos para la entrada, manteniendo una larga agrupación de sinapsis potenciales, y añadiendo y olvidando sinapsis basándonos en su contribución, la suma de las columnas se configurará dinámicamente para obtener el efecto deseado.

Detalles del agrupador espacial

Ahora podemos revisar todo lo que hace la función del agrupador espacial.

1) Empezar con una entrada consistente en un número fijo de bits. Estos bits pueden representar datos sensoriales o puede venir de una región inferior de la jerarquía.

2) Asignar un número fijo de columnas a la región que está recibiendo esta entrada. Cada columna tiene asociada un segmento dendrítico. Cada segmento dendrítico tiene un grupo de sinapsis potenciales representando un subgrupo de bits de entrada. Cada sinapsis potencial cuenta con un valor de permanencia. Basado en el valor de permanencia, algunas de las sinapsis potenciales será valida.

3) Para cualquier entrada, determinar cuantas sinapsis válidas en cada columna están conectadas a bits de entrada activos.

4) El número de sinapsis activas es multiplicado por un factor "estimulador" determinado dinámicamente dependiendo de cuantas veces ha estado activa la columna respecto a sus vecinas.

5) Las columnas con las activaciones más altas después de la estimulación deshabilitan todas las columnas menos un porcentaje prefijado en el radio de inhibición. El radio de inhibición es determinado dinámicamente por la dispersión de los bits de entrada. A partir de este momento contamos con un grupo de columnas activas dispersas.

6) Por cada una de las columnas activas, ajustamos los valores de permanencia de todas las sinapsis potenciales. Los valores de permanencia de las sinapsis alineadas con los bits activos de entrada son incrementados. Los valores de permanencia de las sinapsis alineadas con bits inactivos de entrada son disminuidos. Los cambios en los valores de permanencia puede hacer que cambien algunas sinapsis de ser válidas a inválidas y viceversa.

Conceptos del agrupador temporal

Hay que recordar que el agrupador temporal aprende secuencias y predice. El método básico consiste en que cuando una célula se activa, forma conexiones con otras células que estaban activas justo en el momento anterior. Las células pueden predecir cuando se activarán mirando a sus conexiones. Si todas las células lo hacen colectivamente, pueden guardar y recordar secuencias, y pueden predecir que ocurrirá a continuación. No existe un almacenamiento central para una secuencia de patrones; en lugar de eso la memoria esta distribuida entre todas las células individuales. Ya que la memoria esta distribuida, el sistema es muy robusto contra el ruido y el error. Las células individualmente pueden fallar, pero típicamente afectan poco o nada.

Es importante anotar dos propiedades importantes de las representaciones distribuidas dispersas que explota el agrupador temporal.

Asumamos que tenemos una región hipotética que siempre forma representaciones utilizando 200 células activas de un total de 10.000 células (un 2% de las células están activas en cualquier momento). ¿Cómo puede recordar y reconocer un patrón particular de 200 células activas? Una manera simple de ejecutarlo es hacer una lista de las 200 células activas que nos interesan. Si vemos las mismas 200 células activas de nuevo reconoceremos el patrón. ¿Y si hiciéramos la lista de sólo 20 de ellas e ignoráramos las 180 restantes? ¿Qué pasaría? Podrás pensar que recordar sólo 20 células causaría muchos errores, que esas 20 células estarán activas en muchos patrones de 200. Pero este no es el caso. Ya que los patrones son grandes y dispersos (en este ejemplo 200 células activas de un total de 10.000), recordar 20 células activas es casi tan bueno como recordar todas las 200. Las posibilidades de error en un sistema aplicado son muy bajas y de esta manera se han reducido las necesidad de memoria considerablemente.

Las células en una región HTM toman ventaja de esta propiedad. Los segmentos dendríticos de cada célula tiene un grupo de conexiones a otras células en la región. Un segmento dendrítico forma esta conexiones para poder reconocer el estado de la red en algún momento en el tiempo. Pueden existir cientos o miles de células activas en las inmediaciones pero el segmento dendrítico sólo tiene que conectarse a 15 o 20 de ellos. Cuando el segmento dendrítico ve 15 de estas células activas, puede casi con total seguridad asumir que el patrón está ocurriendo. Esta técnica es llamada “sub-sampling” y es usada en los algoritmos HTM.

Cada célula participa en muchos diferentes patrones distribuidos y en muchas diferentes secuencias. Una célula particular puede ser parte de docenas o centenas de transiciones temporales. De esta manera cada célula cuenta con varios segmentos dendríticos, no sólo uno. Idealmente una célula podría tener un segmento dendrítico por cada patrón de actividad que desea reconocer. En la práctica un segmento dendrítico puede aprender conexiones para muchos patrones diferentes y a pesar de todo funcionar bien. Por ejemplo, un segmento podría aprender 20 conexiones por cada 4 patrones diferentes, lo que daría un total de 80 conexiones. Entonces establecemos el umbral de manera que el segmento dendrítico se active cuando cualquiera de sus 15 conexiones se active. Esto introduce la posibilidad de error. Es posible, por el azar, que la dendrita alcance su umbral de 15 conexiones activas mezclando partes de diferentes patrones. De todas maneras, este tipo de error es muy poco probable, gracias a la dispersión de la representación.

Ahora podremos ver como una célula con una o dos docenas de segmentos dendríticos y unos pocos miles de sinapsis puede reconocer cientos de patrones separados de actividad celular.

Detalles del agrupador temporal

Enumeramos los pasos que realiza el agrupador temporal. Empezamos donde lo dejó el agrupador espacial, con un grupo de columnas activas representando la entrada de alimentación.

1) Por cada columna activa, buscamos células en la columna que estén en estado predictivo, y las activamos. Si no hay células en estado predictivo, activamos todas las células en la columna. El grupo resultante de células activas es la representación de la entrada en el contexto de la entrada anterior.

2) Por cada segmento dendrítico en cada célula de la región, contamos cuantas sinapsis establecidas están conectadas a células activas. Si el número excede el umbral, ese segmento dendrítico es marcado como activo. Las células con segmentos dendríticos activos se pasan a estado predictivo a no ser que estén ya activos debido a la entrada de alimentación. Las células sin dendritas activas y las células no activas debido a la entrada de alimentación siguen inactivas. La colección de células en estado predictivo es la predicción de la región.

3) Cuando un segmento dendrítico se activa, modifica los valores de permanencia de todas las sinapsis asociadas con el segmento. Por cada sinapsis potencial en el segmento dendrítico activo, se incrementa la permanencia de esas sinapsis conectadas a las células inactivas. Estos cambios en la permanencia de las sinapsis son marcadas como temporales.

Esto modifica las sinapsis en los segmentos que han sido suficientemente entrenados para hacer que el segmento se active, y por lo tanto lleva a la predicción. De todas maneras, siempre querremos extender la predicción lo más atrás en el tiempo posible. Por lo tanto, seleccionamos un segundo segmento dendrítico en la misma célula para entrenarla. Para el segundo segmento, seleccionamos el que mejor encaje en el estado del sistema en el momento temporal previo. Para este segmento, utilizando el estado del sistema en el momento temporal previo, incrementamos la permanencia de esas sinapsis que están conectadas a las células activas y disminuimos la permanencia de las sinapsis que estén conectadas a las células inactivas. Estos cambios en las permanencias son marcadas como temporales.

4) En cualquier momento en el que una célula pasa de estar inactiva a activa debido a la entrada de alimentación, atravesamos cada sinapsis potencial asociada con la célula y eliminamos cualquier marca temporal. De esta manera sólo actualizamos la permanencia de las sinapsis sólo si han predicho correctamente la activación de la célula por la entrada de alimentación.

5) Cuando una célula cambia de cualquiera de los estados activos al inactivo, se deshacen todos los cambios de permanencia marcados como temporales por cada

sinapsis potencial. No queremos fortalecer la permanencia de las sinapsis si han predicho incorrectamente la activación de la célula.

Sólo las células que están activas debido a la alimentación propagan la actividad dentro de la región, de otra manera las predicciones conllevarían otras nuevas predicciones. Todas las células activas (por alimentación o las predictivas) forman la salida de la región y se propagan a la siguiente región en la jerarquía.

Secuencias de primer orden versus secuencias de orden variable y predicción

Antes de que terminemos con los agrupadores espaciales y temporales deberemos ver otro tema importante. Puede que no sea del interés de todos los lectores y no es necesaria su lectura para la comprensión de los Capítulos 3 y 4.

¿Cuál es el efecto de tener más o menos células por columna? Específicamente, ¿Qué pasa si tenemos sólo una célula por columna?

En el ejemplo anterior, mostramos que una representación de una entrada compuesta por 100 columnas activas con 4 células por columna se puede codificar de 4^100 maneras diferentes. Por lo tanto, la misma entrada puede aparecer en muchos contextos sin existir confusión. Por ejemplo, si los patrones de entrada representan palabras, entonces una región puede recordar muchas sentencias que usan las mismas palabras una y otra vez y no confundirse. Una palabra como “perro” tendría una única representación en contextos diferentes. Esta habilidad permite a una región HTM lo que se conoce como predicciones de “orden variable”. Una predicción de orden variable no está basada sólo en lo que está ocurriendo en la actualidad, sino en cantidades variables de contextos pasados. Una región HTM es una memoria de orden variable.

Si incrementamos a cinco células por columna, el número posible de codificaciones de cualquier posible entrada en nuestro ejemplo se incrementaría a 5^100, un incremento enorme sobre los 4^100 iniciales. Pero cualquiera de estos números son tan grandes que para muchos ejemplos prácticos el incremento en capacidad puede no ser útil.

Disminuir mucho el número de células por columna si que conlleva una gran diferencia.

Si bajamos hasta tener una célula por columna, perdemos la habilidad de incluir contexto en nuestras representaciones. Una entrada a una región siempre resulta en la misma predicción, independientemente de la actividad previa. Con una célula por columna, la memoria de una región HTM es una memoria de “primer orden”; las predicciones sólo se basarán en la entrada actual.

La predicción de primer orden está especialmente preparada para un tipo de problema que suele resolver el cerebro: la inferencia espacial estática. Tal como se dijo con anterioridad, un humano expuesto a una imagen visual durante muy poco tiempo puede reconocer el objeto aunque el tiempo de exposición haya sido tan corto que no de tiempo a mover el ojo. Con el oído, siempre hace falta escuchar una secuencia de patrones para reconocer de que se trata. La visión normalmente funciona de la misma manera, normalmente se procesa un flujo de imágenes visuales. Pero bajo ciertas condiciones podrás reconocer una imagen con una sola exposición.

El reconocimiento temporal y el estático parece que pudieran necesitar distintos mecanismos de inferencia. Uno requiere reconocer secuencias de patrones y hacer predicciones en un contexto de longitud variable. El otro requiere reconocer un patrón espacial estático sin utilizar el contexto temporal. Una región HTM con múltiples células por columna está especialmente preparada para reconocer secuencias temporales, y una región HTM con una célula por columna está especialmente preparada para reconocer patrones espaciales. En Numenta, hemos realizado muchos experimentos aplicados a problemas visuales haciendo uso de regiones con columnas de una única celda. Los detalles de estos experimentos escapan al alcance de este capítulo; de todas maneras cubriremos los conceptos más importantes.

Si exponemos una región a imágenes, las columnas en la región aprenderán a representar organizaciones espaciales de píxeles comunes. La clase de patrones que se aprenden son similares a lo que se observa en la región V1 del neocórtex (es una región neocortical muy estudiada en biología), típicamente líneas y esquinas con diferentes orientaciones. Mediante el entrenamiento con imágenes en movimiento, la región HTM aprende las transiciones de las formas básicas. Por ejemplo, una línea vertical en una posición muchas veces es seguida de una línea vertical torcida hacia la izquierda o la derecha. Todas las transiciones de patrones más habituales son recordadas por la región HTM.

¿Qué pasaría si exponemos una región a una imagen de una línea vertical moviéndose hacia la derecha? Si nuestra región cuenta sólo con una célula por columna, predecirá que la línea puede ser que se mueva hacia la izquierda o hacia la derecha. No puede utilizar el contexto para saber donde estaba la línea en el pasado y por lo tanto saber que se está moviendo hacia la derecha o hacia la izquierda. Lo que descubrimos es que estas columnas de una única celda se comportan como las "células complejas" en el neocórtex. La salida predictiva de una de estas células estará activa para una línea visible en diferentes posiciones, independientemente de si la línea se está moviendo hacia la derecha, izquierda, o esté parada. También hemos observado que una región de este tipo exhibe estabilidad hacia la translación, cambios en escala, etc. mientras mantiene la habilidad de diferenciar entre imágenes diferentes. Este comportamiento es el requerido para la invariabilidad espacial (reconocer el mismo patrón en diferentes posiciones de la imagen).

Si hacemos el mismo experimento en una región HTM con múltiples células por columna, encontraremos que las células se comportan como "células complejas direccionalmente sintonizadas" en el neocórtex. La salida predictiva de una célula estará activa para una línea moviéndose hacia la izquierda o para una línea moviéndose hacia la derecha, pero no para las dos.

Según todo lo anterior, podemos proponer las siguientes hipótesis. El neocórtex debe poder hacer predicciones tanto de primer orden como de orden variable. Hay cuatro o cinco capas de células en cada región del neocórtex. Las capas difieren de varias maneras pero todas ellas tienen la mismas propiedades de respuesta por columna y una gran conectividad horizontal dentro de cada capa. Especulamos que cada capa de células en el neocórtex está realizando una variación de las reglas de inferencia y aprendizaje HTM descritos en este capítulo. Las diferentes capas de células juegan un papel diferente. Por ejemplo, es sabido de estudios anatómicos que la capa 6 crea la retroalimentación en la jerarquía y que la capa 5 está relacionada con el comportamiento motor. Las dos capas de alimentación primarias son la 4 y la 3. Especulamos que una de las diferencias entre las capas 4 y 3 es que las células en la capa 4 actúan independientemente, i.e. una célula por columna, mientras que las células en la capa 3 actúan como múltiples células por columna. Por lo tanto las regiones en el neocórtex cerca de las entradas sensoriales cuentan memorias tanto de primer como de orden variable. La memoria de secuencia de primer orden (burdamente correspondiente a la capa 4 de las neuronas) es útil a la hora de formar representaciones que son invariantes a los cambios espaciales. La memoria de orden secuencial variable (burdamente correspondiente a la capa 3 de neuronas) es útil para inferencia y predicción de imágenes en movimiento.

En resumen, postulamos que algoritmos similares a los descritos en este capítulo están trabajando en todas las capas de neuronas del neocórtex. Las capas en el neocórtex varían en muchos detalles, lo que les hace jugar roles diferentes relacionados con la alimentación versus la retroalimentación, atención, y comportamiento motor. En regiones cercanas a la entrada sensorial, es útil tener una capa de neuronas generando la memoria de primer orden ya que esto conlleva la invariabilidad espacial.

En Numenta, hemos experimentado con regiones HTM de primer orden (una única célula por columna) para problemas de reconocimiento de imágenes. También hemos experimentado con regiones HTM de orden variable (varias células por columna) para reconocer y predecir secuencias de orden variable. En el futuro, sería lógico probar a combinar las dos en una región única y extender los algoritmos para otros cometidos. De todas maneras, creemos que muchos problemas muy interesantes se pueden atacar con el equivalente de la región de capa simple, múltiples células por columna, tanto en solitario como en una jerarquía.

Capitulo 3: Implementación del Agrupador Espacial y Pseudocódigo

Este capítulo contiene el pseudocódigo detallado para una primera implementación de la función del agrupador espacial. La entrada de este código es una matriz binaria que sube desde los sensores o el nivel anterior. El código computa activeColumns(t) – la lista de columnas que ganan debido a la entrada que sube en el momento t. Esta lista es entonces enviada como entrada a la rutina del agrupador temporal descrita en el siguiente capítulo, i.e. activeColumns(t) es la salida de la rutina agrupador espacial.

El pseudocódigo está dividido entre 3 distintas fases que ocurren en secuencia:

Fase 1: computar la superposición con la entrada actual para cada columna
Fase 2: computar las columnas ganadoras después de la inhibición
Fase 3: actualizar la permanencia de las sinapsis y las variables internas

A pesar de que el aprendizaje del agrupador espacial es inherentemente en línea, se puede desconectar el aprendizaje simplemente saltando la fase 3.

El resto del capítulo contiene el pseudocódigo para cada una de las tres etapas. Las varias estructuras de datos y rutinas de soporte utilizadas en el código serán definidas al final.

Inicialización

Antes de recibir cualquier entrada, la región es inicializada computando una lista de sinapsis potenciales iniciales para cada columna. Consiste de un grupo aleatorio de entradas seleccionadas del espacio de entradas. Cada entrada es representada por una sinapsis y asignada un valor de permanencia aleatorio. Los valores de permanencia aleatorios son elegidos con dos criterios. Primero, los valores son elegidos para que estén en un pequeño rango alrededor de connectedPerm (el mínimo valor para la permanencia para la cual la sinapsis se considera conectada). Esto permite a las sinapsis potenciales conectarse (o desconectarse) después de unas pequeñas iteraciones de entrenamiento. En segundo lugar, cada columna tiene un centro natural sobre le región de entrada, y los valores de permanencia tienen una tendencia hacia este centro (tienen valores más altos cerca del centro).

Fase 1: Superposición

Dado un vector de entrada, la primera fase calcula la superposición de cada columna con ese vector. La superposición de cada columna es simplemente el número de sinapsis conectadas con entradas activas, multiplicado por su respectivo estímulo. Si el valor es menor que minOverlap, definimos el valor de superposición a cero.

```
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)
```

Fase 2: Inhibición

La segunda fase calcula qué columnas quedan como vencedoras después de la fase de inhibición. desiredLocalActivity es un parámetro que controla el número de columnas que acaban ganando. Por ejemplo, si desiredLocalActivity es 10, una columna será la ganadora si su valor de superposición es mayor que el valor de la décima columna con el valor más alto en un radio de inhibición.

```
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.
```

Fase 3: Aprendizaje

La tercera fase ejecuta el aprendizaje; actualiza los valores de la permanencia en todas las sinapsis que la necesiten, así como los radios de estímulo e inhibición.

La regla de aprendizaje principal está implementada entre las líneas 20-26. Para las columnas ganadoras, si una sinapsis está activa, su valor de permanencia es incrementado, en otro caso es disminuido. Los valores de permanencia están limitados para estar entre el 0 y el 1.

Las líneas 28-36 implementan el estímulo. Existen dos mecanismos de estímulo separados para ayudar a las columnas a aprender las conexiones. Si una columna no gana lo suficiente (según se mide en activeDutyCycle), su valor de estímulo general será incrementado (líneas 30-32). Alternativamente, si las sinapsis conectadas a una columna no encajan bien con ninguna entrada (según se mide en overlapDutyCycle), sus valores de permanencia serán estimulados (líneas 34-36). Nota: una vez el aprendizaje ha sido apagado, boost(c) se congela.

Finalmente, al final de la Fase 3 el radio de inhibición es recalculado (línea 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.
```

Estructuras de datos y rutinas de soporte

Las siguientes variables y estructuras de datos son usados en el pseudocódigo:

columns	Lista de todas las columnas
input(t,j)	La entrada a este nivel en el momento t. input(t, j) es 1 si la posición de entrada j está activada.
overlap(*c*)	La superposición del agrupador espacial de la columna c con un patrón de entrada particular.
activeColumns(*t*)	Lista de los índices de las columnas que han resultado ganadoras debido a la entrada de alimentación.
desiredLocalActivity	Un parámetro que controla el número de columnas que serán ganadoras después de la etapa de inhibición.
inhibitionRadius	Tamaño medio del campo receptivo de las columnas.
neighbors(c)	Lista de todas las columnas que están dentro del inhibitionRadius de la columna c.
minOverlap	El número mínimo de entradas que tiene que estar activa para que una columna pueda ser considerada en la etapa de inhibición.
boost(c)	El valor de estímulo de la columna c cuando está aprendiendo – utilizado para incrementar el valor de superposición para las columnas inactivas.
synapse	Una estructura de datos que representa una sinapsis – contiene un valor de permanencia y el índice de la fuente de datos.
connectedPerm	Si el valor de permanencia para una sinapsis es mayor que este valor, se dirá que está conectada.
potentialSynapses(c)	La lista de sinapsis potenciales y sus valores de permanencia.
connectedSynapses(c)	Un subgrupo de potentialSynapses(c), donde la permanencia es mayor que connectedPerm. Son las entradas de alimentación que están conectadas a la columna c en el momento actual.
permanenceInc	Cantidad por la que se incrementan los valores de permanencia de las sinapsis durante el aprendizaje.
permanenceDec	Cantidad por la que se disminuyen los valores de permanencia de las sinapsis durante el aprendizaje.

activeDutyCycle(c)	Media variable que representa cuantas veces la columna c ha estado activa después de la inhibición (por ejemplo, durante las últimas 1000 iteraciones).
overlapDutyCycle(c)	Media variable que representa cuantas veces la columna c ha tenido una superposición significativa (o sea, mayor que minOverlap) con sus entradas (por ejemplo durante las últimas 1000 iteraciones).
minDutyCycle(c)	Variable que representa la tasa de disparo mínima deseada para una célula. Si la tasa de disparo cae por debajo de este valor, será estimulada. Este valor está calculado como el 1% de la tasa de disparo máxima de sus células vecinas.

Las siguientes rutinas de apoyo se usan en el código anterior.

kthScore(cols, k)

Pasándole la lista de columnas, devuelve el k'esimo mayor valor de superposición.

updateActiveDutyCycle(c)

Calcula una media variable de cuantas veces la columna c ha sido activada después de la inhibición.

updateOverlapDutyCycle(c)

Calcula una media variable de cuantas veces la columna c tiene el valor de superposición mayor que minOverlap.

averageReceptiveFieldSize()

El radio medio del tamaño de campo receptivo conectado de todas las columnas. El tamaño de campo receptivo conectado de una columna incluye sólo las sinapsis conectadas (aquellas con valores de permanencia >= connectedPerm). Se utiliza para determinar el grado de inhibición lateral entre las columnas.

maxDutyCycle(cols)

Devuelve el máximo ciclo de servicio activo de las columnas pasándole la lista de las columnas.

increasePermanences(c, s)

Incrementa el valor de permanencia de cada sinapsis en la columna c por un factor de escala s.

boostFunction(c)

Devuelve el valor de estimulación de una columna. El valor de estimulación es un escalar >= 1. Si activeDutyCyle(c) está por encima de minDutyCycle(c), el valor de estimulación es 1. La estimulación se incrementa linealmente una vez el activeDutyCyle de la columna empieza a bajar por debajo de su minDutyCycle.

Capítulo 4: Implementación y Pseudocódigo del Agrupador Temporal

Este capítulo contiene el código detallado para una primera implementación de la función de agrupador temporal. La entrada para este código es activeColumns(t), después de ser computado por el agrupador espacial. El código computa el estado activo y predictivo para cada célula en el momento temporal actual, t. El booleano OR de cada estado activo o predictivo en cada célula forma la salida del agrupador temporal para la siguiente etapa.

El pseudocódigo ha sido separado en tres distintas fases que ocurren en secuencia:

Fase 1: computar el estado activo, activeState(t), para cada célula.
Fase 2: computar el estado predictivo, predictiveState(t), para cada célula
Fase 3: actualiza las sinapsis

La Fase 3 es sólo necesaria para el aprendizaje. De todas maneras, al contrario que con el agrupador espacial, las Fases 1 y 2 contienen algunas operaciones específicas del aprendizaje. Ya que el agrupador temporal es significativamente más complicado que el agrupador espacial, primero listaremos la versión de sólo inferencia, seguido de una versión que combina inferencia y aprendizaje. Se ha añadido al final del capítulo una descripción de algunos de los detalles de implementación, terminología y rutinas de soporte.

Pseudocódigo del agrupador temporal: sólo inferencia

Fase 1

La primera fase calcula el estado activo de cada célula. Para cada columna ganadora determinamos que células deberían activarse. Si la entrada de alimentación fuera predicha por cualquier célula (esto es, su predictiveState era 1 debido al segmento de secuencia en el paso de tiempo previo), entonces esas células se activan (líneas 4-9). Si la entrada de alimentación hubiera sido inesperada (esto es, ninguna célula tenía la salida predictiveState en "on"), entonces cada célula en la columna se activa (líneas 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
```

Fase 2

La segunda fase calcula el estado predictivo de cada célula. Una célula cambiará a "on" su predictiveState si cualquiera de sus segmentos se activa, esto es, si suficientes conexiones horizontales están actualmente disparando debido a la entrada de alimentación.

```
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
```

Pseudocódigo del agrupador temporal: inferencia y aprendizaje combinados

Fase 1

La primera fase calcula el activeState para cada célula de una columna ganadora. Para esas columnas, el código selecciona un célula por columna como la célula de aprendizaje (learnState). La lógica funciona de la siguiente manera: si la entrada de alimentación ha sido predicha por cualquier célula (esto es, su salida predictiveState es 1 debido a un segmento de secuencia), entonces esas células se activan (líneas 23-27). Si ese segmento se activa desde células con el learnState "on", esta célula es seleccionada como la célula de aprendizaje (líneas 28-30). Si la entrada de alimentación no ha sido predicha, entonces todas las células se activan (líneas 32-34). Adicionalmente, la célula que mejor encaje es seleccionada como la célula de aprendizaje (líneas 36-41) y un nuevo segmento es añadido a esa célula.

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

Fase 2

La segunda fase calcula el estado predictivo para cada célula. Una célula activará su estado predictivo si uno de sus segmentos se activa, esto es, si suficientes de sus entradas laterales están activas debido a la entrada de alimentación. En este caso, la célula encadena los siguientes cambios: a) refuerza el segmento activo actual (líneas 47-48), y b) refuerzo de un segmento que podría haber predicho esta activación, esto es, un segmento que tiene un (potencialmente débil) encaje con la actividad en el paso de tiempo anterior (líneas 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)
```

Fase 3

La tercera y última fase es la que realiza el aprendizaje. En esta fase las actualizaciones de los segmentos que han sido puestos en cola son efectivamente implementados una vez que tenemos la alimentación y la célula es elegida como la célula que realizará el aprendizaje (líneas 56-57). Al contrario, si la célula deja de predecir por cualquier razón, reforzaremos negativamente los segmentos (líneas 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.
```

Detalles y terminología de la implementación

En esta sección describiremos algunos de los detalles de la implementación y terminología de nuestro agrupador temporal. Cada célula se indexa utilizando dos números: un indexador de columna, c, y un indexador de célula, i. Las células mantienen una lista de segmentos dendríticos, donde cada segmento contiene una lista de las sinapsis y el valor de la permanencia de cada sinapsis. Los cambios en las sinapsis de una célula son marcados como temporales hasta que la célula se activa por la entrada de alimentación. Estos cambios temporales son mantenidos en segmentUpdateList. Además cada segmento mantiene un señalizador booleano, sequenceSegment, indicando si el segmento predice o no la entrada de alimentación en el siguiente paso temporal.

La implementación de las sinapsis potenciales en el agrupador temporal es diferente de la implementación en el agrupador espacial. En el agrupador espacial, la lista completa de sinapsis potenciales se representa como una lista explícita. En el agrupador temporal en cambio, cada segmento puede tener su propia (y probablemente grande) lista de sinapsis potenciales. En la práctica, mantener una larga lista para cada segmento es computacionalmente costoso e intensivo en memoria. Por lo tanto, en el agrupador temporal, aleatoriamente añadimos sinapsis activas a cada segmento durante el aprendizaje (controlado por el parámetro newSynapseCount). Esta optimización tiene un efecto similar a mantener la lista completa de sinapsis potenciales, pero la lista por cada segmento es considerablemente más pequeña y al mismo tiempo mantiene la posibilidad de aprender nuevos patrones temporales.

El pseudocódigo además utiliza una pequeña máquina de estados para hacer el seguimiento de los estados de las células en diferentes momentos en el tiempo. Mantenemos tres estados diferentes por cada célula. Las matrices activeState y predictiveState hacen el seguimiento del estado activo y predictivo de cada célula en cada momento. La matriz learnState determina qué salidas de célula son usadas durante el aprendizaje. Cuando una entrada es inesperada, todas las células en una columna en particular se activan en el mismo instante en el tiempo. Sólo una de esas células (la que mejor encaje con la entrada) tiene su estado learnState activado. Sólo añadimos sinapsis de células que tengan el learnState en modo “on” (esto evita la sobre-representación de una columna completamente activa en los segmentos dendríticos).

Las siguientes estructuras de datos son utilizadas en el pseudocódigo del agrupador temporal:

cell(c,i)	Una lista de células, indexadas por i y c.
cellsPerColumn	Número de células en cada columna.
activeColumns(*t*)	Lista de los índices de las columnas ganadoras debido a la entrada de alimentación (salida del agrupador temporal).
activeState(c, i, t)	Un vector booleano con un número por célula. Representa el estado activo de la columna c célula i en el momento t pasándole la entrad de alimentación y el contexto temporal pasado. activeState(c, i, t) es la contribución desde la columna c célula i momento t. Si es 1, la célula tiene entrada de alimentación así como el contexto temporal adecuado.
predictiveState(c, i, t)	Vector booleano con un número por célula. Representa la predicción de la columna c célula i en el momento t, pasándole la alimentación de otras columnas y el contexto temporal pasado. predictiveState(c, i, t) es la contribución de la columna c célula i en el momento t. Si es 1, la célula predice la entrada de alimentación en el contexto temporal actual.
learnState(c, i, t)	Un booleano indicando si la célula i en la columna c ha sido elegida como la célula para aprender.
activationThreshold	Umbral de activación para un segmento. Si el número de sinapsis conectadas en un segmento es superior que activationThreshold, se dice que el segmento está activo.
learningRadius	El área alrededor de una célula del agrupador temporal desde la cual puede recibir conexiones laterales.
initialPerm	Valor de permanencia inicial para una sinapsis.
connectedPerm	Si el valor de permanencia para una sinapsis es mayor que este valor, se dice que está conectada.
minThreshold	Umbral de actividad mínima del segmento para el aprendizaje.
newSynapseCount	Número máximo de sinapsis añadidas al segmento durante el aprendizaje.
permanenceInc	La cantidad por la que se incrementan los valores de la permanencia de las sinapsis cuando ocurre el aprendizaje.
permanenceDec	La cantidad por la que se disminuyen los valores de la permanencia de las sinapsis cuando ocurre el aprendizaje.

segmentUpdate	Estructura de datos que guarda tres tipos de información requeridos para actualizar el segmento: a) índice de segmentos (-1 si es un nuevo segmento), b) una lista de las sinapsis activas existentes, y c) un señalizador indicando si este segmento debería estar marcado como un segmento de secuencia (**false** por defecto).
segmentUpdateList	Una lista de estructuras de segmentUpdate. segmentUpdateList(c,i) es la lista de cambios para la célula i en la columna c.

Las siguientes rutinas de soporte han sido usadas en el código de arriba:

segmentActive(s, t, state)

Esta rutina devuelve **true** si el número de sinapsis conectadas en el segmento s que son activas debidas al estado en el momento t es mayor que activationThreshold. El estado del parámetro puede ser activeState, o learnState.

getActiveSegment(c, i, t, state)

Para la columna c célula i, devuelve un índice de segmentos tal que segmentActive(s,t, state) es **true**. Si hay múltiples segmentos activos, los segmentos de secuencia tienen preferencia. En otro caso, los segmentos con mayor actividad tienen preferencia.

getBestMatchingSegment(c, i, t)

Para la columna c célula i en el momento t, encuentra el segmento con el número más grande de sinapsis activas. Esta rutina es agresiva encontrando el mejor encaje. El valor de permanencia de las sinapsis puede estar por debajo de connectedPerm. El número de sinapsis activas puede estar por debajo de activationThreshold, pero ha de estar por encima de minThreshold. La rutina devuelve el índice de segmentos. Si no encuentra segmentos, devuelve un índice de -1.

getBestMatchingCell(c)

Para la columna, devuelve la célula con el mejor segmento coincidente (tal como se ha definido arriba). Si no hay células con segmentos coincidentes, devuelve la célula con el menor número de segmentos.

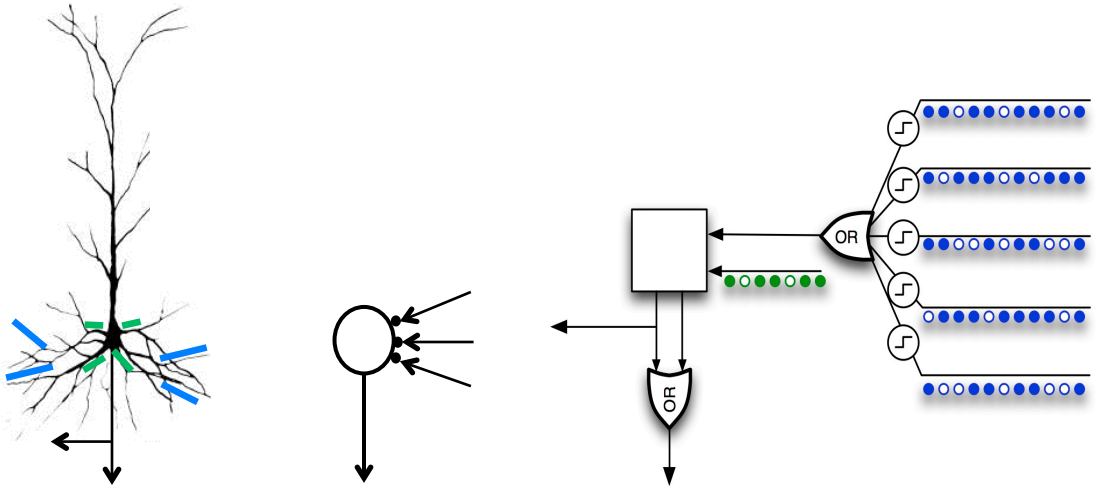
getSegmentActiveSynapses(c, i, t, s, newSynapses= **false**)

Devuelve una estructura de datos del tipo segmentUpdate que contiene una lista con los cambios propuestos para el segmento s. activeSynapses es la lista de sinapsis activas donde las células originales tienen su salida activeState output = 1 en el momento t. (Esta lista estará vacía si s = -1, ya que el segmento no existe.) newSynapses es un argumento opcional con el valor por defecto igual a **false**. Si newSynapses es **true**, entonces newSynapseCount - count(activeSynapses) sinapsis serán añadidas a activeSynapses. Estas sinapsis son elegidas aleatoriamente desde el grupo de células que tienen la salida learnState = 1 en el momento t.

adaptSegments(segmentList, positiveReinforcement)

Esta función itera sobre una lista de segmentUpdate-s, y refuerza cada segmento. Para cada elemento segmentUpdate, los siguientes cambios son realizados. Si positiveReinforcement es **true** entonces las sinapsis en la lista activa ven sus contadores de permanencia incrementados por permanenceInc. El resto de las sinapsis ven sus contadores de permanencia disminuidos por permanenceDec. Si positiveReinforcement es **false**, entonces las sinapsis en la lista activa ven sus contadores de permanencia disminuidos por permanenceDec. Después de esta etapa, cualquier sinapsis en segmentUpdate que todavía exista verá su contador de permanencia añadido por initialPerm.

Apéndice A: Comparación entre Neuronas Biológicas y Células HTM



La imagen superior muestra una neurona biológica en la izquierda, una neurona artificial simple en el medio, y una neurona o "célula" HTM en la derecha. El objetivo de este apéndice es aportar un mejor entendimiento de las células HTM y ver como funcionan mediante la comparación con neuronas reales y neuronas artificiales más simples.

Las neuronas reales son tremendamente complicadas y variadas. Nos enfocaremos en los principios más generales y sólo en los que apliquen a nuestro modelo. A pesar de que ignoremos muchos detalles de las neuronas reales, las células utilizadas en los algoritmos de aprendizaje cortical HTM son mucho más realistas que las neuronas artificiales utilizadas en la mayoría de las redes neuronales. Todos los elementos incluidos en las células HTM son necesarios para la operación de una región HTM.

Neuronas biológicas

Las neuronas son las células que transportan la información en el cerebro. La imagen de la izquierda es de una típica neurona excitatoria. La apariencia visual de una neurona está dominada por las ramas de dendritas. Todas las entradas excitatorias a una neurona son a través de las sinapsis alineadas a lo largo de las dendritas. En los últimos años el conocimiento de las neuronas ha avanzado considerablemente. El cambio mayor ha sido darse cuenta de que las dendritas de una neurona no son simples conductos que traen las entradas hacia el cuerpo

celular. Ahora se sabe que las dendritas son elementos complejos de procesamiento no lineal. El algoritmo de aprendizaje cortical HTM se beneficia de esas propiedades no lineales.

Las neuronas tienen varias partes.

Cuerpo celular
El cuerpo celular es el pequeño volumen en el centro de la neurona. La salida de la célula, el axón, se origina en el cuerpo celular. Las entradas a la célula con las sinapsis alineadas a lo largo de las dendritas que alimentan el cuerpo celular.

Dendritas Proximales
Las ramas de dendritas más cercanas al cuerpo celular son llamadas dendritas proximales. En el diagrama de arriba algunas de las dendritas proximales están marcadas con líneas verdes.

Muchas de las sinapsis activas en las dendritas proximales tienen casi un efecto lineal aditivo en el cuerpo celular. Cinco sinapsis activas conllevarán a casi cinco veces la despolarización en el cuerpo celular respecto a una única sinapsis activa. Al contrario, si la sinapsis única es activada repetidamente por una rápida sucesión de potenciales de acción, la segunda, tercera y sucesivos potenciales de acción tienen mucho menos efecto en el cuerpo celular que el primero.

Por lo tanto, podemos decir que las entradas a las dendritas proximales suman linealmente en el cuerpo celular, y que los picos rápidos llegando a una única sinapsis tendrá un efecto un poco mayor que un único pico de potencial de acción.

Las conexiones de alimentación a una región del neocórtex normalmente se conectan a las dendritas proximales. Al menos esto ha sido lo observado para las neuronas de la capa 4, la capa de entrada primaria en cada región.

Dendritas Distales
Las ramas de dendritas más alejadas del cuerpo celular se llaman dendritas distales. En el diagrama algunas de las dendritas distales han sido marcadas con rayas azules.

Las dendritas distales son más finas que las dendritas proximales. Se conectan a otras dendritas en las ramas del árbol dendrítico y no conectan directamente con el cuerpo celular. Estas diferencias aportan unas propiedad eléctricas y químicas únicas a las dendritas distales. Cuando una única sinapsis es activada en una dendrita distal, tienen un efecto mínimo en el cuerpo celular. La despolarización que ocurre localmente en la sinapsis se debilita para cuando llega al cuerpo celular. Por muchos años esto se vió como un misterio. Parecía que las sinapsis distales, la mayoría en una neurona, no podían hacer mucho.

Ahora se sabe que las secciones de las dendritas distales actúan como regiones de procesamiento semi-independientes. Si hay suficientes sinapsis activas al mismo

tiempo en una corta distancia a lo largo de la dendrita, puede generar un pico dendrítico que puede llegar al cuerpo celular con un gran efecto. Por ejemplo, veinte sinapsis activas a menos de 40 um entre cada una de ellas generara un pico dendrítico.

Por lo tanto, podemos decir que las dendritas distales actúan un grupo de detectores de umbrales.

Las sinapsis formadas en las dendritas distales son predominantemente de otras células cercanas en la misma región.

La imagen muestra una gran rama de dendritas extendiéndose hacia arriba llamada la dendrita apical. Una teoría afirma que esta estructura permite a la neurona localizar varias dendritas distales en un área donde pueden hacer más fácilmente conexiones a los axones que pasen. En esta interpretación, la dendrita apical actúa como una extensión más de la célula.

Sinapsis

Una neurona típica puede tener varios miles de sinapsis. La gran mayoría (puede que el 90%) de ellas estarán en dendritas distales, y el resto estará en las dendritas proximales.

Durante muchos años se ha asumido que el aprendizaje consistía en reforzar o debilitar el "peso" de las sinapsis. A pesar de que este efecto haya sido observado, cada sinapsis es de alguna manera estocástica. Cuando se activa, no liberará un neurotransmisor de manera fiable. Por lo tanto los algoritmos utilizados por el cerebro no pueden depender en la precisión o fidelidad de los pesos de las sinapsis individuales.

Además, ahora se sabe que las sinapsis se forman y deshacen muy rápidamente. Esta flexibilidad representa una manera de aprender muy potente y explica mejor la rápida adquisición de conocimiento. Una sinapsis solo se puede formar si un axón y una dendrita están lo suficientemente cerca, lo que nos lleva al concepto de sinapsis "potenciales". Atendiendo a los supuestos anteriores, el aprendizaje ocurre en gran medida mediante la formación de sinapsis válidas desde las sinapsis potenciales.

Salida de la neurona

La salida de una neurona es un pico, o "potencial de acción", que se propaga a lo largo del axón. El axón deja el cuerpo celular y casi siempre se separa en dos. Una de las ramas viaja horizontalmente haciendo conexiones con las células cercanas, y la otra rama se proyecta hacia otras capas o células o cualquier otro sitio en el cerebro. En la imagen de la neurona de arriba, el axón no es visible. Hemos añadido una línea y dos flechas para representarlo.

A pesar de que la salida de la neurona es siempre un pico, existen diferentes versiones de cómo se ha de interpretar. La visión predominante (especialmente

para el caso del neocórtex) es que lo que importa es la frecuencia de los picos lo que importa. Por lo tanto la salida de una célula se puede ver como un valor escalar.

Algunas neuronas también pueden tener un comportamiento en “ráfagas”, series cortas y rápidas diferentes a los patrones habituales de los picos.

El objetivo de esta descripción era poder dar una breve introducción a las neuronas. Se ha focalizado en los atributos que corresponden a las características de las células HTM y deja fuera múltiples detalles. No todas las características descritas son universalmente aceptadas. Las incluimos porque son necesarias para nuestros modelos. Lo que se sabe de las neuronas podría fácilmente ocupar varios libros, la investigación sobre las neuronas continua muy activa hoy en día.

Neuronas artificiales simples

La imagen del medio del principio de este Apéndice muestra un elemento parecido a una neurona utilizado en muchos modelos clásicos de redes neuronales artificiales. Estas neuronas artificiales tienen un grupo de sinapsis, cada una con un peso. Cada sinapsis recibe una activación escalar, la cual es multiplicada por el peso de la sinapsis. La salida de todas las sinapsis es sumada de manera no lineal para generar la salida de la neurona artificial. El aprendizaje ocurre ajustando los pesos de las sinapsis y a veces la función no lineal.

Tanto para este tipo de célula como para sus variantes se ha probado su utilidad en muchas aplicaciones. De todas maneras, no captura la complejidad y el poder computacional de las neuronas biológicas. Si queremos entender y modelar como funciona un grupo de neuronas reales en el cerebro, necesitamos un modelo neuronal más sofisticado.

Células HTM

En nuestra ilustración, la imagen de la derecha representa una célula utilizada en los algoritmos de aprendizaje cortical HTM. Una célula HTM captura muchas de las capacidades más importantes de las neuronas reales, pero además hace muchas simplificaciones.

Dendrita Proximal
Cada célula HTM tiene una única dendrita proximal. Todas las alimentaciones entrantes a la célula son realizadas a través de las sinapsis (mostradas como puntos verdes). La actividad de las sinapsis es sumada linealmente para producir la activación de la célula debido a la alimentación de entrada.

Requerimos que todas las células en una misma columna tengan la misma respuesta a la alimentación. En las neuronas reales esto probablemente se conseguirá con un tipo de células inhibitorias. En las HTM simplemente forzamos todas las células en una columna a compartir una única dendrita proximal.

Para evitar tener células que nunca ganen la competición con las células vecinas, una célula HTM estimulará su activación si no está ganando lo suficiente relativamente a sus vecinas. Por lo tanto existe una competición constante entre las células. En HTM lo modelamos como una competición entre columnas, no células. Esta competición no ha sido ilustrada en el diagrama.

Finalmente, la dendrita proximal tiene un grupo asociado de sinapsis potenciales el cual es a su vez un subgrupo de todas las entradas de la región. A medida que la célula va aprendiendo, incrementa o disminuye el valor de "permanencia" de todas las sinapsis potenciales en la dendrita proximal. Sólo esas sinapsis potenciales que están por encima del umbral son válidas.

Tal como se ha comentado previamente, el concepto de sinapsis potenciales viene de la biología, donde se refiere a axones y dendritas que están lo suficientemente cerca para formar una sinapsis. Extendemos este concepto a un grupo más grande de conexiones potenciales para una célula HTM. Las dendritas y axones en neuronas biológicas pueden crecer y retraerse a medida que ocurre el aprendizaje y por lo tanto el grupo de sinapsis potenciales va cambiando con el crecimiento. Haciendo que el grupo de sinapsis potenciales de una célula HTM sea grande, podemos aproximarnos al mismo resultado. El grupo de sinapsis potenciales no se ha mostrado.

La combinación de la competición entre columnas, aprender de un grupo de sinapsis potenciales, y la estimulación de las columnas poco utilizadas da a las neuronas de las regiones HTM una plasticidad muy potente, tal como se observa en el cerebro. Una región HTM ajustará automáticamente lo que representa cada columna (mediante cambios en las sinapsis en las dendritas proximales) si la entrada cambia, o si el número de columnas aumenta o disminuye.

Dendritas distales

Cada célula HTM mantiene una lista de segmentos de dendritas distales. Cada segmento actúa como un detector de umbrales. Si el número de sinapsis activas en cualquier segmento (mostrados como puntos azules en el diagrama anterior) es superior al valor de umbral, el segmento se activa, y la célula asociada entra en estado predictivo. El estado predictivo de una célula es el OR de las activaciones de sus segmentos.

Un segmento dendrítico recuerda el estado de la región formando conexiones a las células que estaban activadas conjuntamente en un momento en el tiempo. El segmento recuerda un estado que precede a la célula activándose debido a la entrada de alimentación. Por lo tanto el segmento está mirando un estado que

prediga que su célula se activará. Un umbral típico para un segmento dendrítico es 15. Si 15 sinapsis válidas en un segmento se activan a la vez, la dendrita se activará. Puede haber cientos o miles de células activas en las cercanías, pero conectarse sólo a 15 es suficiente para reconocer el patrón.

Cada segmento dendrítico distal tiene además asociado un grupo de sinapsis potenciales. El grupo de sinapsis potenciales es un subgrupo de todas las células de la región. A medida que el segmento va aprendiendo, aumenta o disminuye el valor de la permanencia de todas sus sinapsis potenciales. Sólo las sinapsis potenciales que están por encima de un umbral son válidos.

En una implementación, podemos usar un número fijo de segmentos dendríticos por célula. En otra implementación, añadimos y borramos segmentos mientras entrenamos. Los dos métodos pueden funcionar. Si tenemos un número fijo de segmentos dendríticos por célula, es posible guardar varios grupos diferentes de sinapsis en el mismo segmento. Por ejemplo, digamos que tenemos 20 sinapsis válidas en un segmento con un umbral de 15. (Generalmente querremos que el umbral sea menor que el número de sinapsis para mejorar la protección ante el ruido). El segmento ahora puede reconocer un estado particular de las células cercanas. ¿Qué pasaría si añadiéramos otras 20 sinapsis al mismo segmento de manera que representara un estado completamente distinto de las células vecinas? Introduce la posibilidad de error ya que el segmento podría añadir 8 sinapsis activas desde un patrón y 7 sinapsis activas del otro y se activaría incorrectamente. Experimentalmente hemos encontrado que hasta 20 patrones diferentes pueden ser almacenados en un segmento hasta que el error ocurra. Una célula HTM con una docena de segmentos dendríticos puede participar en muchas predicciones diferentes.

Sinapsis

Las sinapsis en una célula HTM tienen un peso binario. No hay nada en el modelo HTM que impida tener pesos escalares, pero debido al uso de los patrones distribuidor dispersos hasta ahora no nos ha hecho falta.

De todas maneras, las sinapsis en una célula HTM tienen un valor escalar llamado "permanencia" que es ajustado durante el aprendizaje. Un valor de permanencia 0,0 representa una sinapsis potencial que no es válida y que no ha evolucionado hasta convertirse en una sinapsis válida. Un valor de permanencia por encima del umbral (típicamente 0,2) representa una sinapsis que acaba de ser conectada pero que podría ser fácilmente desconectada. Un valor de permanencia alto, por ejemplo 0,9, representa una sinapsis que está conectada y que no se desconectará fácilmente.

El número de sinapsis válidas en los segmentos dendríticos proximales y distales de una célula HTM no son fijas. Cambian a medida que la célula es expuesta a patrones. Por ejemplo, el número de sinapsis válidas en las dendritas distales es dependiente de la estructura temporal de datos. Si no hay patrones temporales persistentes en la entrada de una región, entonces todas las sinapsis en los segmentos distales tendrán

valores de permanencia bajos y muy pocas sinapsis serán válidas. Si hay mucha estructura temporal en el flujo de entrada, entonces encontraremos muchas sinapsis válidas con valores de permanencia altos.

Salida de la Célula
Una célula HTM tiene dos salidas binarias diferentes: 1) la célula esta activada debido a la entrada de alimentación (a través de la dendrita proximal), y 2) la célula está activa debido a las conexiones laterales (a través de los segmentos dendríticos distales). Al primero le hemos llamado "estado activo) y al segundo "estado predictivo".

En el diagrama anterior, las dos salidas han sido representadas con dos líneas saliendo del cuerpo celular. La línea de la izquierda es el estado activo debido a la alimentación directa, y la línea de la derecha el estado predictivo.

Sólo el estado activo por alimentación directa está conectado con células en la región, asegurando que las predicciones son siempre basadas en la entrada actual (más contexto). No queremos hacer predicciones basadas en predicciones. Si lo hiciéramos, casi todas las células en la región estarían en el estado predictivo después de unas pocas iteraciones.

La salida de la región es un vector que representa el estado de todas las células. Este vector será la entrada de la siguiente región en la jerarquía si es que existe una. Esta salida es el OR de todas los estados predictivos activos. Combinando los dos estados activos y predictivos, la salida de nuestra región será más estable (cambiará más despacio) que la entrada. Esta estabilidad es una propiedad muy importante de la inferencia de una región.

Lecturas sugeridas

Nos preguntan muchas veces por lecturas sugeridas sobre neurociencias. El campo de las neurociencias es tan amplio que una introducción general requiere consultar muchas fuentes diferentes. Nuevos descubrimientos son publicados constantemente en revistas científicas que son a la vez difíciles de leer y de conseguir sin una afiliación a una universidad.

Estos dos libros que recomendamos a continuación son relevantes a los contenidos de este apéndice.

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

Este libro es una buena fuente para saberlo todo sobre las dendritas. El Capítulo 16 trata las propiedad no lineales de los segmentos dendríticos utilizados en los

algoritmos de aprendizaje cortical HTM. Ha sido escrito por Bartlett Mel, uno de los mayores expertos en la materia.

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

Este libro es una buena introducción al neocórtex. Muchos de los capítulos tratan tipos de células y sus conexiones. Podrás obtener una buena idea sobre las neuronas corticales y sus conexiones, a pesar de que es demasiado antiguo para poder incluir los últimos avances en las propiedades de las dendritas.

Apéndice B: Comparativa entre las Capas del Neocórtex y una Región HTM

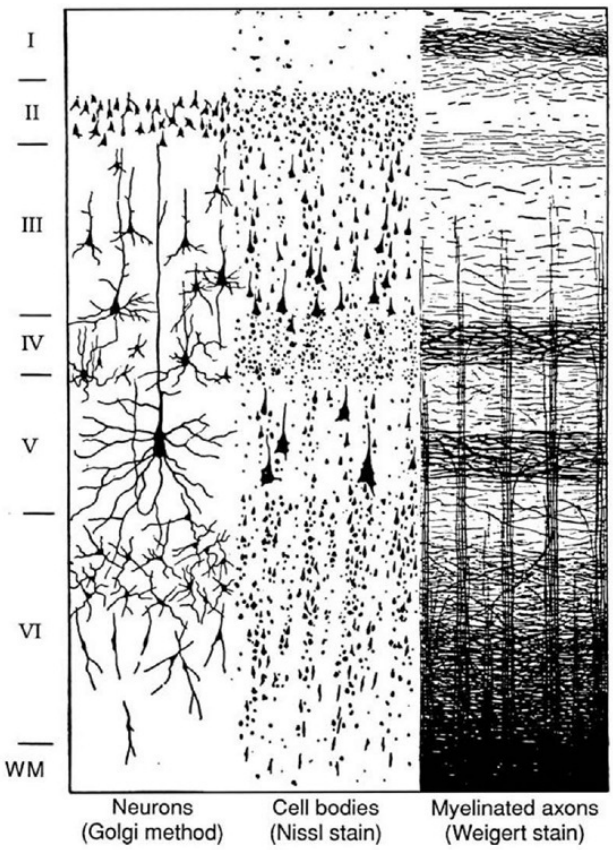
Este apéndice describe la relación entre una región HTM y una región del neocórtex biológico.

Específicamente, el apéndice cubre como el algoritmo de aprendizaje cortical HTM, con sus columnas y sus células, se relaciona con la arquitectura de capas y columnas del neocórtex. El concepto de "capa" en el neocórtex y como se relaciona con una capa HTM es confuso. Esperamos que este apéndice resuelva esta confusión así como pueda aportar un mayor entendimiento de la biología que subyace al algoritmo de aprendizaje cortical HTM.

Circuitería del neocórtex

El neocórtex humano es una hoja de tejido neuronal de aproximadamente 1.000 cm^2 de área y 2mm de espesor. Para poder visualizar esta hoja, piensa en una servilleta, ya que es una aproximación razonable al área y grosor del neocórtex. El neocórtex está dividido en docenas de regiones funcionales, algunas relacionadas con la visión, otras con la audición, otras con el lenguaje, etc. Bajo un microscopio, las características físicas de las diferentes regiones tienen un aspecto muy similar.

Existen varios principios organizativos en cada región a lo largo del neocórtex.



Capas

Generalmente se suele decir que el neocórtex tiene seis capas. Cinco de esas capas contienen células y la sexta casi en su totalidad conexiones. Las capas fueron descubiertas hace 100 años con el nacimiento de las nuevas técnicas de tinción. La imagen de arriba (de Cajal) muestra una pequeña sección del neocórtex expuesta mediante la utilización de 3 métodos de tinción diferentes. El eje vertical abarca el grosor del neocórtex, de aproximadamente 2mm. El lado izquierdo de la imagen indica las seis capas. La Capa 1, arriba, es la capa no celular. El "WM" al final indica el comienzo de la materia blanca, donde los axones viajan a otras partes del neocórtex y otras partes del cerebro.

El lado derecho de la imagen muestra sólo axones mielinizados. (La mielinización es un recubrimiento graso que cubre algunos pero no todos los axones.) En esta parte de la imagen se pueden ver dos de los principios organizativos del neocórtex, las capas y las columnas. La mayoría de los axones se dividen en dos nada más dejar el cuerpo celular. Una rama viajará horizontalmente y la otra verticalmente. La rama horizontal crea muchas conexiones a otras células en la misma o en las capas vecinas, es por esto que las capas se hacen visible en tinciones de este tipo. Hay que tener en cuenta de que este es un dibujo de un sección del neocórtex. La mayoría de los axones están entrando y saliendo del plano de la imagen, por lo que los axones

son más largos de lo que pudiera parecer por esta imagen. Se ha estimado que hay entre 2 y 4 kilómetros de axones y dendritas en cada milímetro cúbico de neocórtex.

La sección del medio de la imagen es una tinción que muestra los cuerpos celulares, pero no muestra ni dendritas ni axones. Se observa que la densidad y el tamaño de las neuronas varía por capa. Sólo existe una pequeña indicación de las columnas en esta imagen en particular. Podemos notar que hay algunas neuronas en la capa 1. El número de neuronas en la capa 1 es tan pequeña que todavía hoy en día se la conoce como la capa no celular. Los neurocientíficos han estimado que debe haber unas 100.000 neuronas en cada milímetro cúbico de neocórtex.

La parte de la izquierda de la imagen es una tinción que muestra el cuerpo, axones, y dendritas de sólo unas pocas neuronas. Se puede observar que el tamaño de los "ramajes" dendríticos varía significativamente en las células de distintas capas. También son visibles algunas "dendritas apicales" que suben desde el cuerpo celular haciendo conexiones en otras capas. La presencia y destino de las dendritas apicales es específico en cada capa.

Resumiendo, la organización en capas y columnas del neocórtex se hace evidente cuando el tejido neural es tincionado y visto a través de un microscopio.

Variaciones de capas en distintas regiones

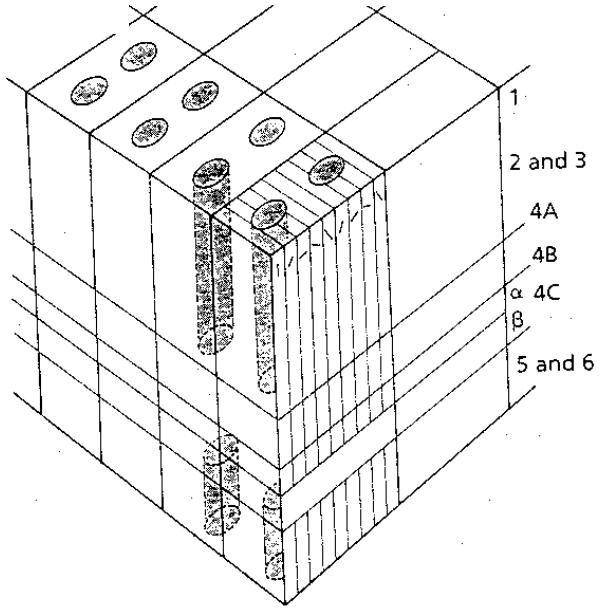
El espesor de las capas varía según la región del neocórtex y existen desavenencias sobre el número de capas. La variación depende del animal estudiado, qué región se está analizando, y de la persona que ejecute el análisis. Por ejemplo, en la imagen superior, las capas 2 y 3 aparecen muy diferenciadas, pero normalmente no suele ser el caso. Algunos científicos reportan que ellos no pueden diferenciar las dos capas en las regiones que ellos están estudiando, por lo que muchas veces las dos capas se agrupan y se llama la "capa 2/3". Otros científicos van en la dirección contraria, definiendo sub-capas tales como 3A y 3B.

La capa 4 está mejor definida en las regiones neocorticales más cercanas a los órganos sensores. En algunos animales (por ejemplo humanos y monos) la capa 4 en la primera región de la visión está claramente subdividida. En otros animales no está subdivida. La capa 4 desaparece generalmente en las zonas jerárquicamente lejanas de los órganos sensores.

Columnas

El segundo principio organizativo principal en el neocórtex son las columnas. Parte de la organización en columnas es visible en imágenes tintadas, pero la mayor parte de la evidencia de la existencia de las columnas se basa en como responden las células a estímulos diferentes.

Cuando los científicos utilizan sondas para ver qué hace activar a las neuronas, observan que las neuronas están alineadas verticalmente, a través de distintas capas, respondiendo más o menos a la misma entrada.



Este dibujo ilustra algunas de las propiedades de las respuestas de las células en V1, la primera región cortical que procesa información de la retina.

Uno de los primeros descubrimientos fue que la mayoría de las células en V1 respondía a líneas o esquinas con diferentes orientaciones en áreas específicas de la retina. Las células que están alineadas verticalmente en columnas responden a las esquinas con la misma orientación. Si miras con detalle, verás que el dibujo muestra una serie de pequeñas líneas a orientaciones diferentes organizadas a través de la parte superior de la sección. Estas líneas indican a que orientación de línea están respondiendo las células en esa ubicación. Las células que están verticalmente alineadas (dentro de las finas líneas verticales) responden a las líneas en la misma orientación.

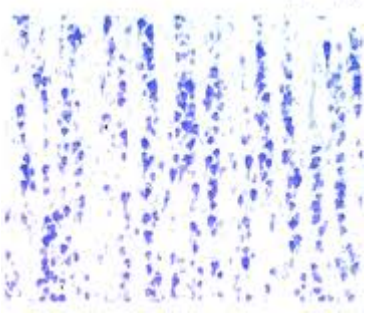
Se han observado muchas más propiedades de las columnas en V1, dos de las cuales se muestran en la imagen. Existen “columnas de dominancia ocular” donde las células responden a combinaciones similares de la influencia del ojo izquierdo y derecho. Existen también “manchas” donde las células son primariamente sensibles

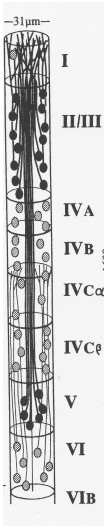
al color. Las columnas de dominancia ocular son los bloques grandes en el diagrama. Cada columna de dominancia ocular incluye un grupo de columnas de orientación. Las “manchas” son los óvalos oscuros.

La regla general del neocórtex es que varias propiedades de respuesta están superpuestas una encima de otra, tales como la orientación o la dominancia ocular. A medida que nos movemos horizontalmente a través de la superficie cortical, la combinación de propiedades de respuesta de las células va cambiando. De todas maneras, las neuronas alineadas verticalmente comparten el mismo grupo de propiedades de respuesta. Este alineamiento vertical existe para las áreas auditivas, visuales y somatosensoriales. Existe cierto debate entre los neurocientíficos de si esto cierto o no en todo el neocórtex, pero al menos parece que lo es en la mayoría de las áreas.

Mini-columnas

La estructura de columna más pequeña en el neocórtex es la mini-columna. Las mini-columnas tienen unos 30um de diámetro y contienen unas 80-100 neuronas a través de 5 capas celulares. Todo el neocórtex está formado por mini-columnas. Puedes visualizarlas como pequeñas piezas de espagueti apiladas una al lado de la otra. Existen pequeños huecos con pocas células entre las mini-columnas, a veces son visibles en imágenes tincionadas.





En la izquierda nos encontramos con una imagen tincionada que muestra cuerpos celulares de neuronas en una parte de un corte neocortical. La estructura vertical de las mini-columnas es evidente en esta imagen. En la derecha se muestra un dibujo conceptual de una mini-columna (de Peters y Yilmez). En la realidad es más fino. Notar que existen múltiples neuronas en cada capa de la columna. Todas las neuronas en una mini-columna responderán a entradas similares. Por ejemplo, en los dibujos de la sección de la región V1 mostrado previamente, una mini-columna contendrá células que responderán a líneas en una orientación particular con una

preferencia de dominancia ocular. Las células en la mini-columna adyacente podrían responder a una orientación de línea ligeramente diferente o a una preferencia de dominancia ocular diferente.

Las neuronas inhibitorias juegan un papel esencial en la definición de las mini-columnas. No son visibles ni en la imagen ni en el dibujo, pero las neuronas inhibitorias envían axones en línea recta entre las mini-columnas, dándoles parcialmente su separación física. Se cree también que las neuronas inhibitorias ayudan a forzar todas las células en la mini-columna a responder a estímulos similares.

La mini-columna es el prototipo utilizado para la columna en el algoritmo de aprendizaje cortical HTM.

Una excepción a las respuestas de las columnas

Existe una excepción en las respuestas de las columnas que es relevante a los algoritmos de aprendizaje cortical HTM. Habitualmente los científicos averiguan a qué responde una célula mediante la exposición de un estímulo simple a un animal experimental. Por ejemplo, enseñan una línea en una parte reducida del espacio visual para determinar las propiedades de la respuesta de las células en V1. Cuando se utilizan entradas simples, los investigadores han constatado que las células siempre responden a la misma entrada. De todas maneras, si la entrada simple está embebida en un video, las células se convierten más selectivas. Una célula que responde fiablemente a una línea vertical aislada no siempre responderá de la misma manera cuando la línea esté incluida en un vídeo complejo con escenas en movimiento.

En el algoritmo de aprendizaje cortical HTM, todas las células en una columna comparten las mismas propiedades de respuesta ante la entrada de alimentación, pero en secuencias temporales aprendidas, sólo una de las células de una columna HTM se activará. Este mecanismo es la manera de representar secuencias de orden variable y es análogo a la propiedad descrita para las neuronas. Una entrada simple sin contexto causará que todas las células en una columna se activen. La misma entrada dentro de una secuencia aprendida activará la activación de una única célula.

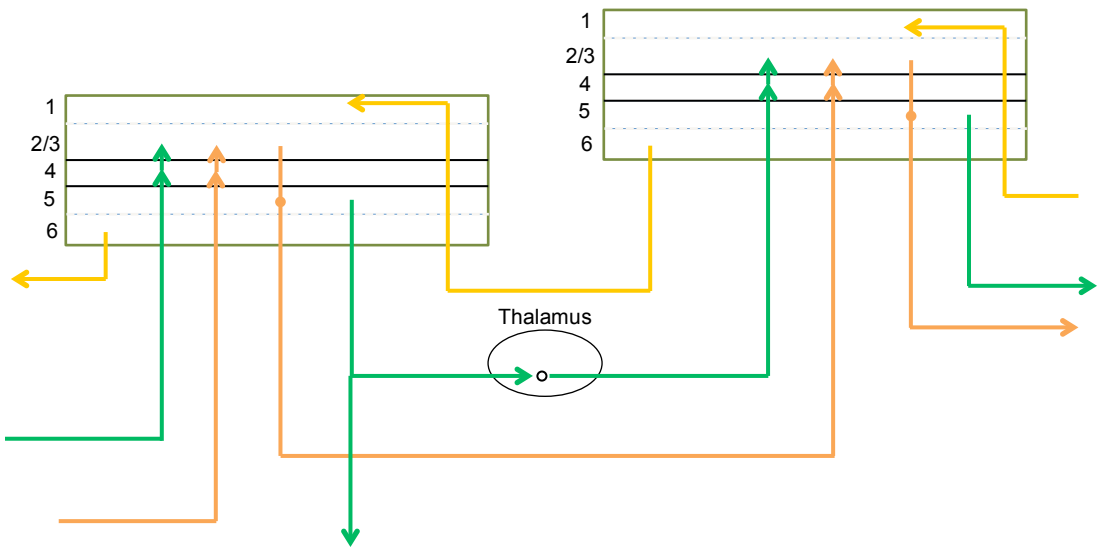
No estamos sugiriendo que en una mini-columna esté sólo una neurona activa en cada momento. El algoritmo de aprendizaje cortical HTM sugiere que dentro de una columna, todas las neuronas dentro de una capa se activarán para una entrada no esperada y que sólo un subgrupo de las neuronas se activarán para una entrada anticipada.

¿Por qué existen capas y columnas?

Nadie conoce con certeza el porqué de la existencia de las columnas y de las capas en el neocórtex. La teoría HTM postula una respuesta. El algoritmo de aprendizaje cortical HTM muestra que una capa de células organizada en columnas puede funcionar como una gran memoria de alta capacidad para transiciones de orden variable. Expuesto más simplemente, una capa de células puede aprender muchas secuencias. Las columnas de células que comparten la misma respuesta a la entrada de alimentación son el mecanismo clave para aprender las transiciones de orden variable.

Esta hipótesis explica porqué las columnas son necesarias, pero ¿qué pasa con las cinco capas? Si una única capa cortical puede aprender secuencias y hacer predicciones, ¿Por qué vemos cinco capas en el neocórtex?

Proponemos que las cinco capas observadas en el neocórtex son todas secuencias de aprendizaje que utilizan el mismo mecanismo básico, pero que las secuencias aprendidas en cada capa se utilizan de distinta manera. Hay mucho que no entendemos sobre este mecanismo, pero podemos describir la idea general. Antes de hacerlo, será de ayuda que describamos la conectividad de las neuronas en cada capa.



El diagrama de arriba ilustra dos regiones neocorticales y las conexiones principales entre ellas. Estas conexiones se pueden ver en todo el neocórtex, donde dos regiones se proyectan una a la otra. La caja de la izquierda representa una región cortical que es jerárquicamente más baja que la región (caja) de la derecha, por lo tanto la información de alimentación va de la izquierda a la derecha del diagrama. La flecha hacia abajo proyecta hacia otras áreas del cerebro. La información de

retroalimentación va de derecha a izquierda. Cada región se divide en capas. Las capas 2 y 3 se muestran juntas como la capa 2/3.

Las líneas de colores representan la salida de las neuronas en las distintas capas. Se conforman de agrupaciones de axones que se originan en las neuronas en la capa. Se ha de recordar que los axones inmediatamente se dividen en dos. Una rama se distribuye horizontalmente en la misma región, primordialmente dentro de la misma capa. Por lo tanto las células en la misma capa están altamente conectadas. Las neuronas y las conexiones horizontales no se muestran en el diagrama.

Existen dos caminos para la entrada de alimentación, un camino directo mostrado en naranja en la imagen, y un camino indirecto mostrado en verde. La Capa 4 es la capa principal para la entrada de alimentación y recibe entradas desde las dos vías. La Capa 4 proyecta hacia la Capa 3.

La Capa 3 es también el origen de la vía directa de alimentación. Por lo tanto, la entrada directa de alimentación se limita a las capas 4 y 3.

Algunas conexiones de alimentación se saltan la capa 4 y van directamente a la capa 3. Además, tal como se comentó arriba, la capa 4 desaparece en las regiones alejadas de las entradas sensoriales. En estos casos la vía va directamente desde la capa 3 a la capa 3 de las siguiente región.

La segunda vía de alimentación (mostrado en verde) se origina en la capa 5. Las células de la capa 3 se conectan a las células de la capa 5 de camino a la siguiente región. Después de dejar la capa cortical, los axones de las capa 5 se vuelven a dividir. Una de las ramas se proyecta a las áreas subcorticales del cerebro involucradas en la acción motora. Se cree que estos axones llevan órdenes motoras (mostradas con flechas que apuntan hacia abajo). La otra rama proyecta a una parte del cerebro llamada tálamo, que actúa como una compuerta. El tálamo o bien deja pasar la información hacia la siguiente región o bien la bloquea.

Finalmente, la vía de retroalimentación primaria, en amarillo, empieza en la capa 6 y se proyecta a la capa 1. Las células en las capas 2, 3, y 5 se conectan a la capa 1 mediante sus dendritas apicales (no mostradas). La capa 6 recibe su entrada desde la capa 5.

Esta descripción es un resumen limitado de lo que se conoce sobre las conexiones entre capas. Es suficiente para comprender nuestras hipótesis sobre el porqué existen múltiples capas si todas las capas al fin al cabo son secuencias de aprendizaje.

Hipótesis sobre lo que hace cada capa

Proponemos que las capas 3, 4 y 5 con capas de alimentación directa y son todas secuencias de aprendizaje. La capa 4 aprende secuencias de primer orden. La capa 3 aprende secuencias de orden variable. La capa 5 aprende secuencias de orden variable temporales. A continuación veremos cada uno de ellas con detalle.

Capa 4
Aprender secuencias de primer orden utilizando el algoritmo de aprendizaje cortical HTM es fácil. Si no forzamos a las células de las columnas a inhibirse entre ellas, esto es, que las células de una misma columna no diferencien el contexto de entradas previas, ocurrirá el aprendizaje de primer orden. El neocórtex obtendría el mismo resultado eliminando el efecto inhibitorio entre las células de la misma columna. En nuestros modelos de ordenador simplemente asignamos una célula por columna, produciendo un resultado similar.

Las secuencias de primer orden son necesarias para formar representaciones invariantes para las transformaciones espaciales de la entrada. Por ejemplo en visión, la translación x-y, la escala, y la rotación son todas transformaciones espaciales. Cuando una región HTM con memoria de primer orden es entrenada con objetos en movimiento, aprende que los distintos patrones espaciales son equivalentes. Las células HTM resultantes se comportarán como las que conocemos "células complejas" en el neocórtex. Las células HTM seguirán activas (en estado predictivo) durante un rango de transformaciones espaciales.

En Numenta hemos realizado experimentos visuales que verifican este comportamiento, y que en cada nivel se consigue algo de invariancia espacial. Los detalles de estos experimentos quedan fuera del alcance de este apéndice.

Aprender secuencias de primer orden en la capa 4 es consistente con que hayan células complejas en la capa 4, y también con que la capa 4 desaparezca en otras regiones del neocórtex. A medida que se sube en la jerarquía llegará un momento en el que no sea posible aprender más invariancias espaciales ya que las representaciones serán invariantes a las mismas.

Capa 3
La capa 3 es la más cercana al algoritmo de aprendizaje cortical HTM que hemos descrito en el Capítulo 2. Aprende secuencias de orden variable y forma predicciones que son más estables que su entrada. La capa 3 siempre proyecta hacia la siguiente región en la jerarquía y por lo tanto conlleva una estabilidad temporal mejorada. La memoria de secuencia de orden variable implica a las neuronas llamadas "células complejas tuneadas direccionalmente", observadas por primera vez en la capa 3. Las células complejas tuneadas direccionalmente diferencian por contexto temporal, tales como una línea moviéndose a la izquierda o a la derecha.

Capa 5

La última capa de alimentación es la 5. Postulamos que la capa 5 es similar a la capa 3 con tres diferencias. La primera diferencia es que la capa 5 añade un concepto de temporalidad. La capa 3 predice "qué" va a pasar a continuación, pero no "cuándo" va a ocurrir. Muchas tareas requieren de la temporalidad, por ejemplo el reconocimiento de palabras habladas, donde el tiempo relativo entre sonidos es importante. El comportamiento motor es otro ejemplo; la sincronización coordinada entre activaciones musculares es esencial. Postulamos que las neuronas de la capa 5 predicen el siguiente estado en el momento esperado. Existen varios detalles biológicos que soportan esta hipótesis. Una es que la capa 5 sea la capa de salida de las señales motoras del neocórtex. Otra es que la capa 5 recibe entradas de la capa 1 originadas en el tálamo (no se muestra en el diagrama). Postulamos que así es como el tiempo es codificado y distribuido a varias células, a través de la entrada a la capa 1 desde el tálamo (no se muestra en el diagrama).

La segunda diferencia entre las capas 3 y 5 es que queremos que la capa 3 haga predicciones lo más alejadas posible en el futuro, para así poder ganar estabilidad temporal. El algoritmo de aprendizaje cortical HTM descrito en el Capítulo 2 lo implementa. En contraste, sólo queremos que la capa 5 prediga el siguiente elemento (en un momento específico). No hemos modelado esta diferencia, pero ocurriría de manera natural si las transiciones se guardaran con el tiempo asociado.

La tercera diferencia entre la capa 3 y la capa 5 se puede ver en el diagrama. La salida de la capa 5 siempre proyecta a los centros motores subcorticales, y la entrada de alimentación va al tálamo. La salida de la capa 5 se pasa a veces a la siguiente región y a veces es bloqueada. Nosotros (y otros) postulamos que esto está relacionado con la atención encubierta (la atención encubierta ocurre cuando atiendes a un estímulo sin respuesta motora).

Resumiendo, la capa 5 combina temporalidad específica, atención, y comportamiento motor. Existen muchos misterios relacionados con conocer el funcionamiento combinado. El punto que queríamos dejar claro es que mediante variaciones sobre el algoritmo de aprendizaje cortical HTM se podría incorporar fácilmente la temporalidad específica y justificar una capa separada en el córtex.

Capa 2 y Capa 6

La capa 6 es el origen de los axones que retroalimentan a las regiones más bajas de la jerarquía. Se conoce mucho menos sobre la capa 2. Tal como hemos mencionado arriba, la mera existencia de la capa 2 es discutida. No tenemos nada más que decir sobre este tema más que resaltar que las capas 2 y 6, como el resto de capas, muestran un patrón de conexiones horizontales masivas y propiedades de respuesta en columna, por lo que postulamos que también corren una variante del algoritmo de aprendizaje cortical HTM.

A que corresponde una región HTM en el neocórtex?
Hemos implementado el algoritmo de aprendizaje cortical HTM de dos maneras distintas, uno con múltiples celdas por columna para la memoria de orden variable, y otro con una única célula por columna para la memoria de primer orden. Creemos que estas dos maneras corresponden a las capas 3 y 4 en el neocórtex. No hemos intentado combinar esas dos variantes en una única región HTM.

A pesar de que el algoritmo de aprendizaje cortical HTM (con múltiples células por columna) es parecida a la capa 3 del neocórtex, contamos con más flexibilidad que el cerebro en nuestros modelos. Podemos crear capas celulares híbridas que no corresponden a capas neocorticales específicas. Por ejemplo, en nuestro modelo conocemos el orden en el que las sinapsis son formadas en los segmentos dendríticos. Podemos utilizar esta información para extraer la predicción de lo que va a ocurrir a continuación desde la predicción general. Además podríamos añadir una temporización específica del mismo modo. Por lo tanto, debería ser posible crear una región HTM de una única capa que combinara las funciones de la capa 3 y la 5.

Resumen

El algoritmo de aprendizaje cortical HTM incorpora lo que creemos que es un componente básico de la organización neural del neocórtex. Muestra como una capa de neuronas conectadas horizontalmente aprende secuencias de representaciones distribuidas dispersas. Variaciones sobre el algoritmo de aprendizaje cortical HTM son utilizados en distintas capas del neocórtex para objetivos relacionados pero diferentes.

Postulamos que la entrada de alimentación a una región neocortical, bien sea a la 4 o a la 3, proyecta principalmente a las dendritas proximales, que con la asistencia de las células inhibitorias, crea una representación distribuida dispersa de la entrada. Postulamos que las células en las capas 2, 3, 4, 5, y 6 comparten esta representación distribuida dispersa. Esto se consigue forzando todas las células de una columna a responder a la misma entrada de alimentación.

Postulamos que las células de la capa 4, cuando existan, utilizan el algoritmo de aprendizaje cortical HTM para aprender las transiciones temporales de primer orden, que son las que consiguen las representaciones invariantes a las transformadas espaciales. Las células de la capa 3 utilizan el algoritmo de aprendizaje cortical HTM para aprender transiciones temporales de orden variable y formar representaciones estables que son transferidas hacia arriba en la jerarquía cortical. Las células de la capa 5 pueden aprender transiciones de orden variable con temporización. No tenemos postulados específicos para las capas 2 y 6. De todas maneras, debido a la conectividad horizontal típica de esas capas, parece que también han de formar algún tipo de aprendizaje de memoria de secuencia.

Glosario

Notas: las definiciones de este aparatado capturan el uso que se les ha dado en este documento, y puede ser que tengan otros significados en otros ámbitos. Los términos en mayúsculas se refieren a otros términos definidos en este glosario.

Estado Activo	el estado por el cual las células están activas debido a la entrada de alimentación
De-abajo-hacia-arriba	Sinónimo de Alimentación
Células	El equivalente HTM de la neurona *Las células se organizan en columnas en las regiones HTM.*
Actividad Coincidente	Dos o más células están activas en el mismo momento
Columna	Grupo de una o más células que funcionan como una unidad en una región HTM *Las células dentro de una columna representan la misma entrada de alimentación, pero en contextos diferentes.*
Segmento Dendrítico	Una unidad de integración de las sinapsis asociadas con las células y las columnas *Las regiones HTM tienen dos tipos diferentes de segmentos dendríticos. Una de ellas está asociada con las conexiones laterales a una célula. Cuando el número de sinapsis activas en el segmento dendrítico supera un umbral, la célula asociada entra en estado predictivo. La otra está asociada con las conexiones de alimentación a una columna. El número de sinapsis activas se suma para obtener la alimentación de activación de una columna.*
Densidad Deseada	Porcentaje deseado de columnas activas debido a la entrada de alimentación a la región *El porcentaje sólo aplica dentro de un radio el cual varia dependiendo de la apertura de las entradas de alimentación. Es "deseada" ya que el porcentaje varía dependiendo de cada entrada.*

Alimentación	Se mueve en la misma dirección que la entrada, o desde un nivel jerárquico más bajo a uno más alto (a veces es llamado de-abajo-hacia-arriba)
Retroalimentación	Se mueve en la dirección contraria a la entrada, o desde un nivel jerárquico más alto a uno más bajo (a veces es llamado de-arriba-hacia-abajo
Predicción de Primer Orden	Predicción basada únicamente en la entrada actual y no en entradas previas – comparar con Predicción de Orden Variable
Memoria Temporal Jerárquica (HTM)	Una tecnología que replica algunas de las funciones estructurales y algorítmicas del neocórtex
Jerarquía	Una red de elementos conectados donde la conexiones entre elementos están unívocamente identificados como de Alimentación o Retroalimentación
Algoritmos de Aprendizaje Cortical HTM	La suite de funciones para el agrupamiento espacial, agrupamiento temporal, aprendizaje y olvido de las que se compone una región HTM, también referido como Algoritmos de Aprendizaje HTM
Red HTM	Una jerarquía de regiones HTM
Región HTM	La unidad principal de memoria y predicción en un HTM *Una región HTM se compone de una capa de células altamente interconectadas organizadas en columnas. Una región HTM hoy en día tiene una única capa de células, mientras que en el neocórtex (y después en HTM), una región tendrá múltiples capas de células. Cuando nos referimos a una región dentro del contexto de su posición en una jerarquía, se le puede llamar nivel.*
Inferencia	Reconocer la similitud de un patrón de entrada espacial o temporal a un patrón aprendido previamente.
Radio de Inhibición	Define el área alrededor de una columna que ésta inhibe activamente
Conexiones Laterales	Conexiones entre las células de una misma región

Nivel	Una región HTM en el contexto de una jerarquía
Neurona	Una célula de procesamiento de información en el cerebro *En este documento se ha específicamente cuando nos referíamos a células biológicas, y "célula" cuando nos referíamos a la unidad de computación HTM.*
Permanencia	Un valor escalar que indica el estado de conexión de una sinapsis potencial *Un valor de permanencia por debajo de un umbral indica que la sinapsis no ha sido formada. Un valor de permanencia por encima del umbral indica que la sinapsis es válida. El aprendizaje en una región HTM se consigue mediante la modificación de valores de permanencia de las sinapsis potenciales.*
Sinapsis Potencial	El subgrupo de todas las células que potencialmente podrían formar sinapsis con un segmento dendrítico en particular *Sólo un subgrupo de sinapsis potenciales será valida en un momento en el tiempo basado en su valor de permanencia*
Predicción	Activar las células (a un estado predictivo) que probablemente pasen a estado activo próximamente debido a la entrada de alimentación *Una región HTM a veces predice muchos entradas futuras posibles al mismo tiempo.*
Campo Receptivo	El grupo de entradas a las que está conectada una columna o una célula *Si la entrada a una región está organizada como una matriz 2D de bits, entonces el campo receptivo se puede expresar como un radio dentro del espacio de entrada*
Sensor	Una fuente de entradas para una red HTM
Representación Distribuida Dispersa	Representación formada por muchos bits en los que sólo un pequeño porcentaje está activo y donde un único bit no es suficiente para llevar información

Agrupamiento Espacial	El proceso de formar representaciones distribuidas dispersas de una entrada *Una de las propiedades del agrupamiento espacial es superponer mapas de patrones de entrada a la misma representación distribuida dispersa*
Sub-Muestreo	Reconocer un gran patrón distribuido mapeando sólo un pequeño subgrupo de los bits activos
Sinapsis	Conexiones entre células que se forman durante el aprendizaje
Agrupamiento Temporal	El proceso de formar la representación de una secuencia de patrones de entrada donde la representación resultante sea más estable que la entrada
De-Arriba-hacia-Abajo	Sinónimo de retroalimentación
Predicción de Orden Variable	Una predicción basada en cantidades variables de contexto previo – comparar con Predicción de Primer Orden *Se llama "variable" ya que la memoria utilizada para mantener el contexto previo es reservado dinámicamente. Por lo tanto, un sistema de memoria que implementa una predicción de orden variable puede utilizar el contexto mirando hacia atrás en el tiempo sin necesidad de cantidades exponenciales de memoria.*