Universidad Nacional Autónoma de México

Facultad de Ciencias

MAPAS AUTOORGANIZADOS PARA EL ANÁLISIS DEL GENOMA

T E S I S

QUE PARA OBTENER EL TÍTULO DE:
LICENCIADO EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA:
ALFREDO JOSÉ HERNÁNDEZ ÁLVAREZ

DIRECTOR DE TESIS:
DR. PEDRO EDUARDO MIRAMONTES VIDAL

2016

1. Datos del alumno
   Hernández
   Alvarez
   Alfredo José
   5518557489
   Universidad Nacional Autónoma de México
   Facultad de Ciencias
   Licenciatura en Ciencias de la Computación
   096535299

2. Datos del tutor
   Dr. Pedro Eduardo
   Miramontes
   Vidal

3. Datos del sinodal 1
   Dra.
   María de Luz
   Gasca
   Soto

4. Datos del sinodal 2
   Dr. José de Jesús
   Galaviz
   Casas

5. Datos del sinodal 3
   Mat.
   Salvador
   López
   Mendoza

6. Datos del sinodal 4
   L. en C. C.
   Sergio
   Hernández
   López

7. Datos del trabajo escrito
   Mapas autoorganizados para el análisis del genoma
   89 páginas
   2016

Mapas autoorganizados para el análisis del genoma

Alfredo José Hernández Álvarez

A la Universidad Nacional Autónoma de México.
A la Facultad de Ciencias.
A Pedro y mis sinodales: Lucy, José, Salvador y Sergio.
A mis padres, Mercedes y José, a mis hermanas, Edita y Natalia.
A mis razones de vivir: Azalea, Ixchel y Mirelle.

Resumen

Las áreas de la genómica y la bioinformática están generando muchos datos en espera de ser analizados y comprendidos. Se han obtenido muchas secuencias genómicas de muchos organismos, algunas ya completadas y otras más en proceso de ser ensambladas. Con estos datos se está tratando de comprender más a fondo cómo y por qué es importante el genoma como repositorio de instrucciones para el desarrollo del organismo.

En el análisis del genoma se ha usado varios métodos de matemáticas aplicadas, estadística, ciencias de la computación, aprendizaje de máquinas y redes neuronales.

El presente trabajo de tesis tiene como objetivo el uso de un tipo especial de red neuronal artificial, el mapa autoorganizado (SOM, por sus siglas en inglés) de Kohonen, en el análisis de secuencias genómicas, usando una representación de su estructura basada en frecuencias de dinucleótidos, con el fin de obtener una agrupación de genomas. Lo que se espera es que genomas de organismos parecidos resulten estar en un mismo grupo.

Palabras clave: redes neuronales, mapas autoorganizados, SOM, genoma

Índice general

Índice de figuras

Índice de tablas

Introducción

Para muchos científicos, como los biólogos, matemáticos y físicos, la llegada de las computadoras causó una revolución en el sentido de que ha servido para confirmar teorías existentes y para generar nuevas teorías en sus campos de trabajo, y para crear nuevas áreas de trabajo por sí mismas.

Una de las áreas que más apoyo ha recibido de la computación es la inteligencia artificial o IA, que toma como base, y extiende, los conceptos de aprendizaje, razonamiento, pensamiento, inteligencia y hasta conciencia, que los filósofos, psicólogos y biólogos han manejado por siglos, para tratar de crear o replicar un ente artificial con inteligencia, entendiendo esto como un programa de cómputo que se comporte o funcione como lo haría un ser inteligente[52], o mejor dicho que sea capaz de realizar tareas que requieren inteligencia cuando las hace un humano.

En la IA se han creado métodos para procesamiento del lenguaje natural, aprendizaje, razonamiento automático, representación del conocimiento, reconocimiento de patrones, visión por computadora y robótica, con el fin de construir un ente inteligente que se comporte como el humano, e incluso que se parezca físicamente.

Uno de los comportamientos de la inteligencia es el aprendizaje. Para su estudio en el ámbito de la inteligencia artificial emergió el área de aprendizaje de máquinas, que busca hacer programas que aprendan, en el sentido de que puedan tomar decisiones inteligentes basados en datos (lo aprendido o la experiencia).

A fin de cuentas las computadoras lo que hacen es manejar información y de lo que se trata es que sepan qué hacer con ella. Mientras nosotros les ayudamos a eso tratando de imitarnos, por replicación o simulación, ellas nos ayudan a entendernos a nosotros mismos.

Con el avance de la investigación en inteligencia artificial, se han desarrollado nuevos métodos para problemas de análisis de datos, incluyendo clasificación, agrupación y optimización. Estos métodos complementan a los tradicionales de

estadística, y se ha visto que en ciertos casos han sido más convenientes. Uno de esos nuevos[1] métodos son las redes neuronales artificiales[2].

Las redes neuronales son sistemas inspirados en la idea de simular el funcionamiento de las neuronas en el cerebro: no sabemos cómo representa el cerebro la información, pero sí que usa muchas unidades de procesamiento altamente interconectadas, las neuronas. Así, se puede decir que los sistemas de redes neuronales realizan tareas que involucran procesamiento simultáneo de una gran cantidad de información (por ejemplo, siguiendo con la analogía del cerebro, tareas como visión, audición y tacto), además de tareas donde se requieren respuestas y reconocimiento rápido y preciso[52]. Algunas de esas tareas son las ya mencionadas clasificación, agrupación o *clustering*, asociación y predicción.

Otra de las áreas enormemente influenciada por las computadoras y la inteligencia artificial es la biología, dando lugar a lo que se conoce como bioinformática, y en un sentido más amplio ayudando al desarrollo de las ciencias genómicas.

En el estudio del desarrollo y evolución de los organismos vivos, a mediados del siglo XIX se explicó el mecanismo de la herencia usando unas unidades de información hereditarias[39], que después se llamarían genes. Esto dió lugar al nacimiento de la genética. Esta área fue impulsada a mediados del siglo XX cuando se describió la estructura de la molécula del componente principal de los genes, el ácido desoxirribonucléico o ADN [62].

Ahora se sabe más del funcionamiento de los genes, y se habla más del genoma, entendido como la secuencia completa de ADN, en donde están incluidos los genes. Con ayuda de las computadoras se han secuenciado ya los genomas completos de varios organismos, en principio muchas bacterias, y los últimos años esta área está más activa que nunca con el secuenciamiento y análisis del genoma humano [53].

Los métodos que se han desarrollado han permitido generar gran cantidad de datos genómicos, que a su vez crean la necesidad de la aplicación de métodos para su análisis y comprensión. Esta nueva área se conoce en general como las ciencias genómicas, que integra métodos de disciplinas como biología, genética, química, ciencias de la computación y estadística. En particular la aplicación de los conceptos computacionales al tratamiento de datos biológicos se conoce como bioinformática o biología computacional.

La necesidad del análisis de la enorme cantidad de datos genómicos[3] hace

[1] En realidad la idea no es nueva, pero en los últimos años han tenido un resurgimiento.

[2] Se usará el término *red neuronal* para referirse a una red neuronal artificial.

[3] Las bases de datos de secuencias genómicas han crecido de forma exponencial en los últimos

que sea conveniente el uso de métodos de análisis inteligente de datos, no sólo análisis estadístico, si no además técnicas de inteligencia artificial para explorar, analizar y extraer información. El uso y desarrollo de algoritmos computacionales en genómica es relativamente reciente, pero cada vez mayor. Las bases de datos de secuencias genómicas almacenan las secuencias de ADN y proteínas como cadenas de letras, algo que un programa de computadora maneja perfectamente. La computadora nos sirve como herramienta para aplicar o desarrollar los métodos de análisis de datos sobre las secuencias genómicas, para obtener información de ellas, y posiblemente para inferir conocimiento, por ejemplo de su estructura tridimensional o del funcionamiento del material genético contenido. Esto puede llevar a preguntas como si ¿se puede inferir algo del funcionamiento basándose en la estructura del genoma? ¿genomas estructuralmente parecidos son de organismos parecidos? Ya hay mucho trabajo en esta área. El posterior entendimiento puede ayudar a lograr avances importantes en áreas emergentes como la medicina genómica o la agrogenómica.

Objetivo del trabajo

El objetivo de este trabajo es la aplicación de un tipo especial de red neuronal artificial de aprendizaje no supervisado, el algoritmo de mapas autoorganizados o SOM[4], para la agrupación de genomas completos de bacterias, basado en una representación estructural usando frecuencias de dinucleótidos en la secuencia. Se espera que genomas estructuralmente parecidos estén al final cercanos en el mapa resultante, es decir que estén en un mismo grupo. En otras palabras, se espera que el SOM agrupe los genomas de los organismos parecidos, se podría decir que de especies similares, o de especies en un mismo género.

Desarrollo del trabajo

En el Capítulo 1 se dará una introducción a las redes neuronales, se mencionarán diferentes tipos básicos y aplicaciones generales.

En el Capítulo 2 se describirá en particular el algoritmo de mapas autoorganizados para la agrupación de datos mediante aprendizaje no supervisado. Se

30 años.

[4] *Self Organizing Map*

presentarán formas de visualización comunes y una descripción breve de sus usos en diferentes áreas.

En el Capítulo 3 se dará una introducción al concepto de genómica, las secuencias de ADN y las bases de datos genómicas. Además, se mencionarán algunos proyectos en ésta área. El propósito de este capítulo es conocer la terminología del área y la composición básica del genoma, para su posterior análisis.

En el Capítulo 4 se expondrá el experimento realizado, la aplicación del SOM en un conjunto de genomas. En primer lugar se describirá la representación vectorial utilizada para cada genoma. Se presentarán experimentos con una codificación propia del algoritmo de mapas autoorganizados, y una visualización sencilla del mapa resultante. También se describirán algunas pruebas con un programa ya existente que implementa el SOM, y se mostrarán los resultados y visualizaciones obtenidos con él.

Finalmente se mencionarán las conclusiones y posible trabajo futuro.

Agradecimientos

Quiero agradecer en particular al Dr. Pedro Miramontes por su infinita paciencia, a Lucy Gasca, José Galaviz, Salvador López y Sergio Hernández. También a la Dra. Alejandra Medina y la Dra. Maribel Hernández por sus comentarios.

Capítulo 1

Redes neuronales

1.1. Antecedentes

En el año de 1943 apareció un artículo en el que se decía que "el sistema nervioso es una red de neuronas" y que tanto la actividad nerviosa como el comportamiento de la red puede tratarse por medio de la lógica proposicional[37]. Además se demuestra que puede tener el mismo poder de cálculo que una máquina universal de Turing. Este trabajo es reconocido en general como el primero del área que después se conocería como inteligencia artificial o IA[52].

Si bien la IA se *fundó* oficialmente en 1956[1], es heredera de ideas y técnicas de otras disciplinas, algunas milenarias, como la filosofía, las matemáticas, la lógica, la psicología, la lingüística y las ciencias de la computación. Por ejemplo, el mencionado artículo de 1943 de McCulloch, neurofisiólogo, y Pitts, matemático, toma las bases de trabajos de Russell y Whitehead de 1912[65] y de Turing de 1936[57]. Alan Turing propuso las *máquinas automáticas*, que fueron modeladas por McCulloch y Pitts[37].

Otra aportación importante de Turing a la IA fue la llamada *prueba de Turing*[58]. Ésta última propone contestar la pregunta "¿Pueden las máquinas pensar?" mediante la imitación del humano, fundamentada en la hipótesis de que si una máquina se comporta en todos los aspectos como inteligente, entonces debe ser inteligente. Esos trabajos dieron la pauta para que después emergiera el aprendizaje de máquinas, entendiendo el aprendizaje como un comportamiento

[1] En una conferencia en Dartmouth, donde estuvieron presentes varios científicos que después serían personalidades del área, como John McCarthy, Marvin Minsky, Claude Shannon, Herbert Simon y Allen Newell.

inteligente.

Hubo mucho entusiasmo al principio en el desarrollo de la IA, y de hecho hubo avances durante una década. Después se encontraron obstáculos y hubo una época de desencantamiento[2], sobre todo tal vez por el excesivo optimismo inicial, por el desacuerdo en los enfoques y en las posibilidades[3] y por la cada vez mayor complejidad de los problemas que se abordaban[52].

Al principio la idea central de la IA era tratar de resolver tareas que se pensaba que requerían mucho razonamiento para los humanos, como el juego de ajedrez o la resolución de problemas matemáticos[4]. En los años posteriores se ha visto que tareas comunes o mundanas para los humanos, como tareas de percepción (visión, audición, movimiento) o procesamiento del lenguaje, son muy difíciles de programar en una computadora. Se ha visto que las computadoras pueden hacer cálculos aritméticos complicados sin error en nanosegundos, pero los humanos no; los humanos pueden ver o distinguir fácilmente cosas o rostros, escuchar palabras, entender oraciones completas, pero las computadoras no. ¿Por qué? Se piensa que estas tareas requieren de gran procesamiento simultáneo de muchos datos y que tal vez la estructura del cerebro humano, con millones de neuronas interconectadas y trabajando en paralelo, es mejor para esas tareas[52]. Así, se han desarrollado nuevas arquitecturas y algoritmos de redes neuronales, basados en las ideas iniciales de McCulloch y Pitts.

Aproximadamente desde 1980 ha habido un resurgimiento de la inteligencia artificial en general, debido principalmente al tremendo desarrollo de las computadoras como dispositivos de cálculo y almacenamiento, de ideas como los sistemas expertos, del desarrollo de la programación; y en particular en el área de redes neuronales, por el desarrollo de nuevas arquitecturas y algoritmos, como las ideas de Hopfield y el redescubrimiento de la retropropagación.

La IA se ha abordado desde diferentes enfoques. A grandes rasgos, uno tiene que ver con hacer programas que piensen o actúen de forma inteligente por como están programados, y otro en que piensen o actúen imitando a los humanos. En el caso de las redes neuronales se trata de replicar el funcionamiento del cerebro humano, o de hacer una *metáfora del cerebro*. Estos métodos son llamados subsimbólicos o conexionistas y se diferencían de los métodos *tradicionales* de la IA, llamados simbólicos, como los algoritmos de búsqueda, sistemas expertos, entre

[2] Conocida como *The AI winter*

[3] Por ejemplo, Minksy y Papert en su libro "Perceptrons" de 1969 mencionaban bondades, pero también limitantes - p.ej. problema XOR, que se superó después.

[4] *The Logic Theorist*, de Simon y Newell, en 1955, considerado el primer programa de IA, fue capaz de probar teoremas del *Principia Mathematica* de Russell y Whitehead.

otros.

El conexionismo es un concepto más amplio que involucra diversas disciplinas que estudian los fenómenos de la mente y del comportamiento, y los modelan como procesos que emergen de un conjunto de unidades sencillas altamente conectadas. Las redes neuronales son los modelos conexionistas más conocidos.

Aprendizaje

Los enfoques simbólico y conexionista de la IA difieren en cómo abordan algunos problemas, como la búsqueda, la representación del conocimiento o el aprendizaje. En general el aprendizaje es el proceso por el que se adquieren o modifican habilidades, conocimientos o conductas como resultado del estudio, la experiencia, la instrucción, el razonamiento y la observación.

El aprendizaje de máquinas[5] trata de generar algoritmos que puedan encontrar patrones o alguna otra información útil para tomar decisiones, predecir o generalizar un comportamiento, basándose en una gran cantidad de datos. Comúnmente esta área se relaciona con la estadística pues ambas se basan en el análisis de datos. Hay varios tipos de algoritmos de aprendizaje, pero en general se pueden agrupar en dos clases, en función de su salida: supervisado y no supervisado.

Aprendizaje supervisado

En este caso se efectúa el aprendizaje en una situación en la que es posible percibir las entradas y las salidas de los componentes. El algoritmo produce una función que establece una relación entre las entradas y las salidas deseadas del sistema. El aprendizaje se da por un entrenamiento con datos de entrada que están *etiquetados*, es decir, se sabe qué salida tendrá. Cada dato de entrada o ejemplo se representa como un par que consiste en el vector de entrada y la salida deseada. El algoritmo procesa los pares de entrada y genera una función, que puede ser usada para procesar correctamente nuevas entradas. Se puede pensar como aprendizaje con un *profesor*, porque siempre se obtiene retroalimentación de los resultados que se van obteniendo.

Una variante, se puede decir que es el aprendizaje *por refuerzo*, en el que se usan recompensas o castigos al calcular salidas. Comúnmente también se define el aprendizaje por refuerzo como una tercera categoría de aprendizaje.

[5] O aprendizaje automático.

Las redes neuronales en general son algoritmos de aprendizaje supervisado pues requieren la identificación previa de entradas y salidas correctas en la etapa de entrenamiento. Otros ejemplos de este tipo de algoritmos son los árboles de decisión, el algoritmo *k−vecinos más cercanos* y las máquinas de vectores de sorporte[6].

Este tipo de algoritmos se usa naturalmente en problemas de clasificación, que trata de etiquetar las entradas en una serie de categorías o clases. También se usa para regresión o aproximación de funciones. Además, se puede aplicar en datos secuenciales, para reconocimiento de lenguaje y gestos.

Un ejemplo más de aprendizaje supervisado es el método LVQ[7] para clasificación. Es una versión supervisada de los métodos de cuantificación vectorial. En éste se conoce de antemano el número de clases. Este método fue creado por Teuvo Kohonen[30] y es la versión supervisada de los mapas autoorganizados del mismo autor.

Aprendizaje no supervisado

Se efectúa sin ninguna indicación sobre cuáles son las salidas correctas. En este caso se dice que los datos de entrada no están etiquetados. No hay información a priori de las salidas esperadas ni indicación de recompensas o castigos. El algoritmo toma las entradas como variables aleatorias y construye una función de densidad de probabilidad para el conjunto de datos.

Una forma de aprendizaje no supervisado es el aprendizaje *competitivo*, en el que las neuronas compiten para responder a los datos de entrada. En este caso la red comúnmente tiene una capa intermedia conocida como capa competitiva.

Otro ejemplo es el aprendizaje *hebbiano*[8]. Está basado en el mecanismo de la plasticidad neuronal o sináptica, que dice que el valor de una conexión sináptica se incrementa si las neuronas de ambos lados se activan varias veces de forma simultánea. Esta teoría propone una explicación de la adaptación de las neuronas en el cerebro durante el aprendizaje.

Algunos algoritmos que pueden seguir este aprendizaje son la cuantificación vectorial o VQ[9]. La cuantificación vectorial es una técnica originaria del área de procesamiento de señales, que permite el modelado de funciones de densidad de probabilidad mediante la distribución de vectores de entrada, y útil para la

[6] *Support Vector Machines*
[7] *Learning Vector Quantization*
[8] Parte de una teoría propuesta por Donald Hebb en 1949
[9] *Vector quantization*

compresión de datos. En redes neuronales, algunos algoritmos de aprendizaje no supervisado son los mapas autoorganizados o SOM[10] y algunos modelos de la teoría de la resonancia adaptativa o ART[11].

El aprendizaje no supervisado se puede usar para descubrir características en un conjunto de datos, por ejemplo para la agrupación o *clustering* de datos. Los criterios de agrupamiento pueden ser distancia o similitud de los vectores de entrada. Los grupos o *clusters* encontrados comparten propiedades en común. En esta área se usan algoritmos $k-means$ o redes neuronales como los mapas autoorganizados. El algoritmo $k-means$ es un método de la cuantificación vectorial que particiona los datos de entrada en k grupos, en el que cada dato pertenece al grupo más cercano a la media.

En años recientes se habla mucho de aprendizaje profundo o *deep learning*, como parte de los métodos o algoritmos del aprendizaje de máquinas, con la característica de que el número de transformaciones paramétricas (como pesos y umbrales) de una señal desde la capa de entrada a la salida es mucho mayor. Más recientemente algunos usan el término como una nueva palabra de moda para referirse a los métodos de redes neuronales.

El aprendizaje de máquinas está estrechamente relacionado con el análisis de datos, la estadística y la optimización, entre otras cosas para tratar de hacer predicciones. Por eso a veces se confunde con la minería de datos. En los últimos años se ha usado el término *big data* para referirse a los métodos de análisis de enormes cantidades de datos. También se ha usado el término *data science* para unificar estas áreas de estudio, podría decirse que también como un término de moda para lo que se conoce como descubrimiento de conocimiento.

En lo que sigue del capítulo, se dará una definición de las redes neuronales y se mencionarán algunos tipos. En el siguiente capítulo se describirá en particular el algoritmo de mapas autoorganizados para la agrupación de datos mediante aprendizaje no supervisado.

1.2. Definición

Las redes neuronales son sistemas inspirados en la idea de replicar el funcionamiento de las neuronas en el cerebro. Desde siglos atrás se sabe que el cerebro está relacionado con el aprendizaje y el pensamiento, y se ha estudiado en diferentes áreas: biología, psicología, filosofía, etcétera.

[10] Self Organizing Maps
[11] Adaptive Resonance Theory

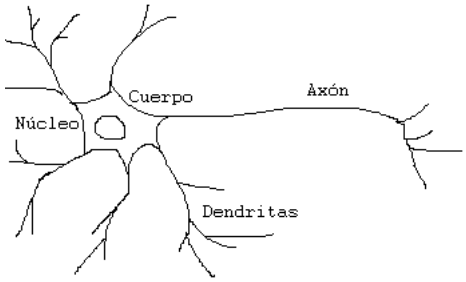


Figura 1.1: Neurona biológica. Se muestran sus componentes básicos: el cuerpo, las dendritas y el axón.

Biológicamente sabemos que la *neurona*, una célula nerviosa, es la unidad funcional básica del sistema nervioso, incluido el cerebro. Las neuronas están altamente interconectadas formando una red en la que se produce una reacción específica de acuerdo a un estímulo dado modificando la composición química de las uniones entre neuronas, lo que además modifica el estímulo que la neurona da a otras. Esta es la forma en que creemos que el cerebro humano aprende[49].

El cerebro humano se compone de aproximadamente 10 billones de neuronas. Las neuronas en el cerebro tienen cuatro componentes básicos (figura 1.1):

- las *dendritas*, por donde reciben señales de otras neuronas;
- el *soma* o cuerpo de la célula;
- el *axón*, por donde viajan impulsos eléctricos de salida, y
- la *sinapsis*, el contacto electroquímico entre las neuronas. Una neurona puede establecer sinapsis con decenas o miles de otras neuronas.

La neurona manda neurotransmisores a través del axón, al final del cual, en la sinapsis, se transmite información a través del impulso eléctrico, que habilita o inhibe la actividad de la neurona conectada. Cuando la neurona recibe un impulso excitatorio mucho mayor que uno inhibitorio, manda a su vez actividad eléctrica por su axón. El aprendizaje ocurre adecuando la efectividad de la sinapsis para influenciar en las otras neuronas. En el cerebro, éstas acciones se llevan a cabo masivamente en paralelo y en milisegundos.

En computación, se define una red neuronal usando la analogía con las neuronas en el cerebro humano. No pretenden ser un modelo de las neuronas humanas. Simplemente se inspiran en ellas.

Una red neuronal consta de muchas unidades de proceso conectadas, cuya salida es una reacción de la combinación de sus entradas y sus conexiones. Estos procesadores elementales se conocen como neuronas artificiales[12] y están enlazados a otros de manera que las salidas de unos son entradas de otros, formando una red (ver por ejemplo la figura 1.3). Dicho de otra forma, una red neuronal puede verse como un conjunto de elementos de cálculo aritmético, unidos en una red que representa una función (de modo similar a los circuitos formados por compuertas lógicas que representan funciones lógicas). A las conexiones entre neuronas se les asigna un valor numérico o peso, que indica la fortaleza de la conexión, y que puede ser adecuado de acuerdo a las entradas durante la etapa de entrenamiento, lo que hace a la red neuronal un sistema adaptable y capaz de aprender. El aprendizaje ocurre al adecuar los pesos de las conexiones.

Como los humanos, las redes neuronales aprenden mediante el ejemplo, por lo tanto deben ser entrenadas o instruidas. En la etapa de entrenamiento se aplica el algoritmo de aprendizaje. Se debe tener un buen entrenamiento inicial para tener una buena operación. Siguiendo la analogía con el cerebro humano, se debe tener experiencia para aprender.

Ventajas y desventajas

Las redes neuronales funcionan bien en tareas como clasificación y reconocimiento de patrones; pero no son buenas en razonamiento basado en reglas. Entre las ventajas de las redes neuronales está que son sistemas dinámicos autoadaptativos, son flexibles y tienen tolerancia a fallos, por ejemplo, funcionan con datos con *ruido* o si falla parte de la red. Otra ventaja es que realizan procesamiento en paralelo, y por ello, con el desarrollo de hardware con gran poder de cómputo, pueden ser más eficientes en algunas aplicaciones.

Una de las desventajas puede ser la naturaleza de *caja negra*, no interpreta lo aprendido, no explica por qué obtuvo esa salida, y los errores son difíciles de rastrear, a diferencia de, por ejemplo, modelos estadísticos. Otra desventaja es que no hay reglas generales para diseñar una red, definir el número de nodos, capas y parámetros de entrada. Se diseñan para aplicaciones específicas, no de forma general.

[12] Definidos por McCulloch y Pitts[37].

1.3. Tipos de redes neuronales

En esta sección se describen los tipos de redes neuronales más comúnes, con el propósito de resaltar algunas partes generales en los algoritmos.

Hay varios tipos de estructuras de red, cada una de las cuales muestra distintas características de cómputo. En general los modelos siguen la siguiente característica común[52]: cada unidad de procesamiento toma las señales y los pesos de sus conexiones de entrada, hace un cálculo (una suma ponderada) y le aplica una función de activación, cuyo resultado es enviado a través de sus conexiones de salida. Ese resultado es el valor de activación de la unidad.

Los tipos de redes neuronales se diferencían principalmente en la dirección del flujo de las señales.

- Redes de prealimentación[13]: las conexiones son unidireccionales, no hay ciclos, se pueden organizar en capas, las unidades de una capa están conectadas solamente a las de la capa siguiente. Este tipo de red forma una gráfica acíclica dirigida o DAG[14]. El ejemplo clásico es el perceptrón. Se puede considerar que los mapas autoorganizados de Kohonen tienen una estructura de red de prealimentación, con una sola capa cuyas unidades están acomodadas en filas y columnas.

- Redes recurrentes: puede haber conexiones de regreso o en dirección contraria. En estas redes la activación puede alimentar a unidades que la originaron previamente, por lo que cuenta con un estado interno guardado en los niveles de activación de las unidades. Se dice que estas redes tienen memoria. Ejemplos: redes de Hopfield, redes BAM[15]. Las redes neuronales recurrentes son un subconjunto de las redes neuronales recursivas.

Las funciones de activación comunes son la función escalón, la función signo y la función sigmoide. Se pueden usar muchas otras funciones de activación, por ejemplo, la función identidad, la función rectificador, que se dice que es biológicamente más razonable[19], la tangente hiperbólica, y la función exponencial normalizada o *softmax*. También se pueden usar funciones de activación distintas en diferentes capas de la red.

La adecuación de los pesos en una red neuronal se puede ver como un problema de optimización. En este punto se pueden usar varios métodos, los más

[13] *Feedforward networks*
[14] *Directed Acyclic Graph*
[15] *Bidirectional Associative Memory*

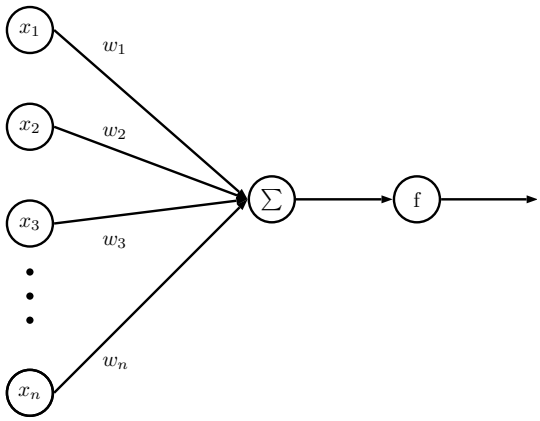


Figura 1.2: Perceptrón simple. El vector de entrada X y el vector de pesos W son combinados en la función $g(x) = \sum$ y el resultado es la entrada de la función de activación f.

comunes son el descenso del gradiente, retropropagación, o una combinación de ambos; aunque en algunos casos también se pueden usar algoritmos genéticos y recocido simulado[16].

Perceptrón

Rosenblatt definió el perceptrón en 1958[50]. Es el modelo matemático más sencillo de una neurona biológica, y es el tipo más sencillo de red neuronal de prealimentación.

La idea es que el perceptrón puede aprender a responder verdadero o falso por medio de la presentación repetida de ejemplos, una fase de entrenamiento que se conoce como la regla de aprendizaje del perceptrón. El perceptrón es especialmente útil en problemas de clasificación.

El aprendizaje del perceptrón es supervisado, esto es que se deben conocer de antemano suficientes ejemplos y a qué clase pertenecen.

El perceptrón modela una neurona que toma entradas que son procesadas y obtienen una salida. En particular el perceptrón modela una neurona artificial que usa la función escalón como función de activación.

[16] *Simulated annealing*

La figura 1.2 muestra el perceptrón. El vector de entrada $X = (x_1, x_2, ..., x_n)$ y el vector de pesos de conexión $W = (w_1, w_2, ..., w_n)$ son de dimensión n y tienen valores reales.

La salida depende de la suma del producto de las entradas con los pesos correspondientes, de acuerdo a cierto valor umbral b. Si el resultado es mayor que ese valor, la salida es 1, en caso contrario es 0.

Se define la función de entrada $g(x)$ como la suma ponderada de los valores del vector de entrada X y el vector de pesos W, ver la ecuación 1.1.

$$g(x) = \sum_{i=0}^{n} w_i x_i \tag{1.1}$$

Se define también la función de activación o salida $f(x)$ como una función escalón, como en la ecuación 1.2.

$$f(x) = \begin{cases} 1, & \text{si } g(x) \geq b \\ 0, & \text{si } g(x) < b \end{cases} \tag{1.2}$$

Debido a que el perceptrón usa como función de activación una función escalón (que responde 0 o 1), se usa para clasificar las entradas, ya sea en verdadero o falso, en positivo o negativo, o en una clase u otra.

Es conveniente manejar el umbral b como un peso de entrada adicional w_0. Esto permite trabajar sólo con pesos durante el aprendizaje, sin manejar de forma separada el umbral.

Con la anterior modificación, se calcula la salida de acuerdo a la ecuación 1.3, considerando el umbral como $w_0 = b$ y $x_0 = 1$.

$$f(x) = \begin{cases} 1, & \text{si } g(x) \geq 0 \\ 0, & \text{si } g(x) < 0 \end{cases} \tag{1.3}$$

El proceso de entrenamiento, en el que se da el aprendizaje, consiste en ajustar o actualizar los pesos de las entradas (recordando que el valor umbral está incluido en los pesos).

Dado un conjunto de datos de entrenamiento, conformado por N vectores x de dimensión n, el primer paso es iniciar los pesos w de manera aleatoria. Después, en cada paso t, para cada entrada calcular la diferencia de la salida esperada d con la salida obtenida $f(x)$ y actualizar los pesos, de acuerdo a la ecuación 1.4.

$$\forall i, 0 \leq i \leq n, w_i(t+1) = w_i(t) + \alpha * (d - f(x)) * x_i(t) \tag{1.4}$$

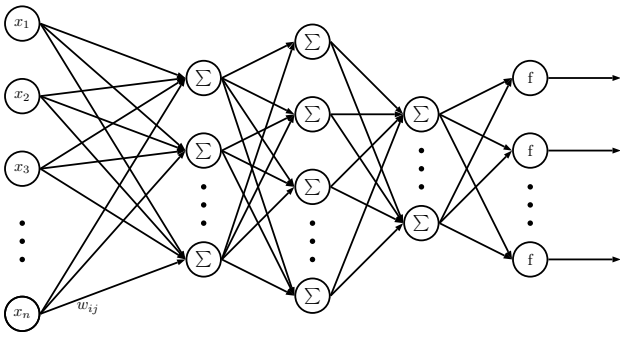


Figura 1.3: Perceptrón multicapa, varias capas ocultas.

α es un número real entre 0 y 1, y representa una tasa de aprendizaje.

Si los datos de entrada son linealmente separables, los pesos convergen después de algunos pasos. Después del periodo de aprendizaje, con los pesos obtenidos, cualquier entrada será clasificada correctamente. Por eso se dice que el perceptrón es un clasificador lineal. Los valores de salida solamente pueden ser de una u otra clase. Esa es la propiedad más importante de los perceptrones: si lo logran calcular, lo aprenden.

Si los datos de entrada no son linealmente separables, el aprendizaje del perceptrón no tendrá éxito. La aplicación de los perceptrones tuvo un estancamiento[17] hasta que se combinaron para formar una red más compleja. Los perceptrones pueden ser organizados en varias capas para tener un *perceptrón multicapa*, con una capa de neuronas de entrada, una o más capas escondidas y una capa de salida como se ve en la figura 1.3. La capa de salida puede tener varios perceptrones (o neuronas), y en este caso las salidas son independientes unas de otras (los pesos afectan sólo a una de las salidas). Puede pensarse que los componentes pueden ser entrenados de forma independiente como perceptrones simples. En realidad un perceptrón multicapa conforma una red neuronal más compleja, en la que se puede usar otros algoritmos de aprendizaje.

Si la dimensión del espacio de entrada es muy grande, puede ser difícil visualizar la clasificación o separabilidad de las entradas. Para explicarlo más claramente, daremos un ejemplo con vectores de entrada de dimensión $n = 2$. En este caso, el perceptrón puede representar una función para clasificar conjuntos de vectores de

[17] Ver *The AI winter* en la sección 1.1

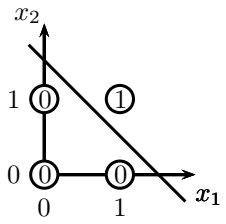


Figura 1.4: Problema de la separabilidad lineal de la función AND

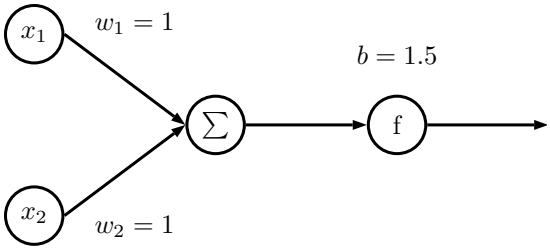


Figura 1.5: Perceptrón simple para la función AND

entrada solamente si se puede dibujar una línea[18] para separarlos, es decir, si son *linealmente separables*.

Un ejemplo típico de estas funciones linealmente separables con dos entradas son las funciones o compuertas lógicas AND y OR. En la figura 1.5 se muestra un perceptrón que representa la función lógica AND, con los pesos y el valor umbral adecuado. En la figura 1.4 se muestra la separabilidad lineal del perceptrón de la función AND. La tabla 1.1 muestra los resultados correctos de la función de activación $f(x)$ definida como en la ecuación 1.2, con el valor umbral $b = 1.5$ adecuado para representar la función AND.

Sin embargo, hay problemas que no son linealmente separables, y por lo tanto que el perceptrón simple no puede resolver. Un ejemplo de ellos es el de la función lógica XOR. Este problema se puede resolver con perceptrones multicapa y algoritmos más complicados, como retropropagación. Los perceptrones multicapa pueden representar cualquier función continua de las entradas, e incluso funciones discontinuas[11, 34].

[18] O un plano si $n = 3$

x_1	x_2	$g(x)$	$f(x)$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	2	1

Tabla 1.1: Tabla de valores de la función lógica AND representada con un perceptrón.

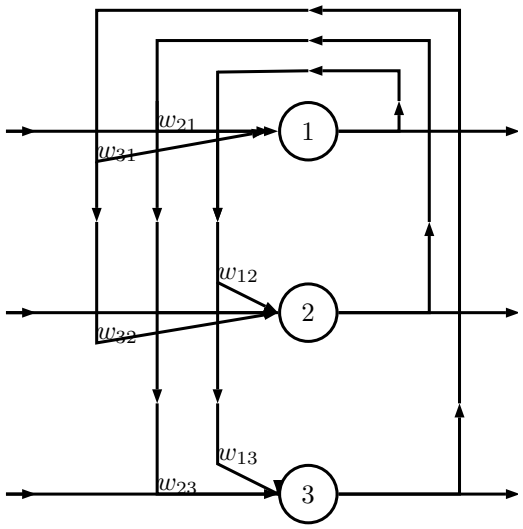


Figura 1.6: Red de Hopfield de 3 nodos.

Redes de Hopfield

Hopfield introdujo una red neuronal recurrente describiendo el uso de la memoria en 1982[22].

La red de Hopfield se muestra en la figura 1.6. Consiste en *N* neuronas interconectadas que:

- utilizan conexiones bidireccionales con pesos simétricos
- todas las unidades son de entrada y de salida, no hay capas intermedias, y todas están conectadas entre sí
- generalmente se usa la función signo como función de activación, los únicos

niveles de activación son $+1$ y -1

- actualizan su valor de activación de forma asíncrona e independiente

La red de Hopfield funciona como una *memoria asociativa*: después de la fase de entrenamiento, una nueva entrada hace que la red se estabilice en un patrón de activación correspondiente a la entrada del conjunto de entrenamiento que más se parezca a la nueva. Es decir, dado un patrón (o una parte de él) la red encuentra el patrón coincidente más cercano o parecido. Los patrones se almacenan en los pesos de la red. Esto da también tolerancia a fallos, si algunas unidades fallan, la red puede seguir funcionando correctamente.

La red funciona de la siguiente manera: se escoge una unidad de forma aleatoria. Si alguna de sus vecinas está activa, se calcula la suma de los pesos de las conexiones a las unidades activas. Si la suma es positiva, la unidad se activa; en caso contrario, se desactiva. Se escoge otra unidad aleatoria y el proceso se repite hasta que la red alcance un estado estable, es decir, hasta que las unidades no cambien de estado.

Hopfield demostró que dado cualquier conjunto de pesos y cualquier estado inicial, el algoritmo llevará a la red a un estado estable.

Una variante de las redes de Hopfield son las máquinas de Boltzmann. En ellas se incluyen unidades internas, o que no son ni de entrada ni de salida, y además usan una función de activación estocástica, por ejemplo, una probabilidad de activación.

Entrenamiento por retropropagación

Minsky y Papert mostraron que un perceptrón de prealimentación de dos capas puede superar varios problemas[40], pero no dijeron cómo ajustar los pesos. Una solución fue presentada en primer lugar por Werbos[64] y después por Parker[46] y Rumelhart, Hinton y Williams[51]. La idea central de las soluciones presentadas es que los errores de las unidades en las capas intermedias son calculados *regresando* los errores de las unidades de la capa de salida. Esta propagación hacia atrás de los errores se conoce como retropropagación[19].

La retropropagación se puede usar en combinación con otros métodos de optimización, como el descenso del gradiente, para entrenar redes neuronales de prealimentación o recurrentes. Normalmente se usa en redes de prealimentación, por lo que se considera un método de aprendizaje supervisado. En particular se

[19] *Backpropagation*

usa en redes multicapa de perceptrones. Como se mencionó, una red multicapa, con una o dos capas ocultas, puede representar cualquier función. El algoritmo de retropropagación busca mantener la propiedad del perceptrón, de que si lo puede calcular, lo puede aprender.

En las redes que aprenden por retropropagación comúnmente se usa la función sigmoide como función de activación, que es una función continua y diferenciable como se ve en la ecuación 1.5.

$$f(x) = \frac{1}{1+e^{-x}} \tag{1.5}$$

El aprendizaje por retropropagación inicia con una conjunto de pesos aleatorios. Los pesos son ajustados cada vez que se requiere corregir un error entre una entrada y una salida esperada. Se requiere dos etapas: un primer pase empezando por las unidades de entrada que calcula las salidas, y después un pase empezando por el nivel de salida, en el que los valores de error son regresados al nivel anterior, y los pesos entre los dos niveles son ajustados para minimizar el error. Cuando se han tomado todas las posibles parejas de entrada-salida, se dice que se ha cumplido un *epoch* o época. El entrenamiento requiere varias épocas.

La retropropagación tienen sus ventajas y desventajas. Una ventaja es que pueden representar cualquier función. Una desventaja puede ser la lentitud en el aprendizaje, aunque han aparecido variaciones mejorando la velocidad.

1.4. Aplicaciones de las redes neuronales

La utilidad de las redes neuronales está en que pueden generar una función a partir de observaciones.

Las aplicaciones de las redes neuronales se pueden clasificar en las siguientes áreas: aproximación de funciones o análisis de regresión, clasificación o reconocimiento de patrones y memoria asociativa, análisis de datos, como *clustering*, filtrado, compresión.

Algunos ejemplos de aplicaciones:

- reconocimiento de patrones: identificación de rostros, reconocimiento de objetos
- reconocimiento de texto escrito
- procesamiento del lenguaje natural

- diagnóstico médico, reconocimiento en imágenes médicas
- procesamiento de señales
- aplicaciones financieras, por ejemplo para toma de decisiones en el mercado de acciones
- filtrado de correo no deseado (*spam*)
- sistemas de detección de intrusos en equipos de cómputo
- astronomía: clasificación de galaxias
- bioinformática
- juegos: backgammon, ajedrez, póker

En el capítulo siguiente se describirá más a detalle el algoritmo de mapas autoorganizados, un tipo de red neuronal basada en aprendizaje no supervisado, ideal para aplicaciones específicas de agrupación de datos.

Capítulo 2

Mapas autoorganizados

2.1. Antecedentes

La idea de que un sistema puede tender al orden por sí mismo es antigua. Fue mencionada por René Descartes en su Discurso del método en el siglo XVII. También mencionada por Immanuel Kant hablando del propósito de las cosas en el siglo XVIII. Algunos naturalistas ya la mencionaban al tratar de explicar las formas de los organismos vivos, en el siglo XVIII. En el siglo XIX se descubrió la Segunda Ley de la Termodinámica, que dice que para un sistema aislado, en un proceso natural la entropía se mantiene o se incrementa con el tiempo. En termodinámica, el término *entropía* se usa como una medida de la energía que no puede realizar un trabajo en un sistema, o bien que no puede ser transferida a su exterior. Por ejemplo, en un sistema con una región con temperatura alta y otra región con temperatura baja, de forma espontánea[1] el calor se distribuirá uniformemente en todo el sistema, la capacidad de realizar un trabajo decrece (por que la diferencia de temperatura entre las regiones decrece), y por lo tanto la entropía aumenta. Puede decirse que la entropía es una medida del grado de distribución de la energía en un sistema.

La entropía también se relaciona con la idea de la irreversabilidad de los procesos, en el sentido de que la entropía siempre se incrementa en un proceso irreversible. En un sentido más amplio comúnmente se usa como una medida del grado de organización de un sistema.

La organización es un concepto muy amplio pero se puede identificar con el concepto de orden. La entropía comúnmente también se mide en términos de

[1] Es decir, sin requerir energía externa al sistema

información, en el sentido de que mientras más información se necesita para describir un sistema, más desordenado está. En teoría de la información, la entropía es una medida numérica que representa la incertidumbre de algo.

La autoorganización se refiere a la idea de que un orden o coordinación global puede ser generado a partir de interacciones locales. Este proceso es espontáneo, en el sentido de que no es dirigido o controlado por un agente externo. Se origina de forma aleatoria y es alentado por retroalimentaciones positivas. La organización resultante es descentralizada o distribuida en todo el sistema.

El término *autoorganización* fue formalizado por W. R. Ashby en un artículo de 1947[3]. En 1960 H. von Foerster habló de la paradoja de la autoorganización y del "orden a partir de ruido"[61]. El concepto también fue mencionado por Norbert Wiener en la segunda edición de su libro Cybernetics[2] de 1961[66]. Fue adoptado por los físicos en el estudio de los sistemas complejos en la década de los 1970s. En el estudio de las ciencias de la complejidad, se menciona la autoorganización como un método para la adaptación en los sistemas complejos[18].

La autoorganización es un fenómeno que puede verse en muchas áreas, por ejemplo:

- Física: sistemas dinámicos, segunda ley de la termódinamica, cristalización, magnetización
- Biología: evolución, plegamiento de proteínas, estructura del ADN, morfogénesis, colonias de hormigas, el vuelo de aves, un ecosistema
- Ciencias sociales: el comportamiento de grupos de animales
- Economía: el mercado de valores
- Ciencias de la computación: algoritmos de optimización, aprendizaje de máquinas

En el área de la IA el aprendizaje de una red neuronal se puede ver como un proceso de autoorganización. La autoorganización puede producir agrupaciones de los datos de entrada más *parecidos*, por lo que se aplica naturalmente para la agrupación o *clustering*.

Existen varios algoritmos en el que se presenta la autoorganización, uno de ellos es claramente el algoritmo de mapas autoorganizados de Kohonen[3]. Es el

[2]En dos capítulos adicionales a la primera edición del libro de 1948.

[3]Se usará el término *mapas autoorganizados* para referirse al algoritmo de mapas autoorganizados de Kohonen.

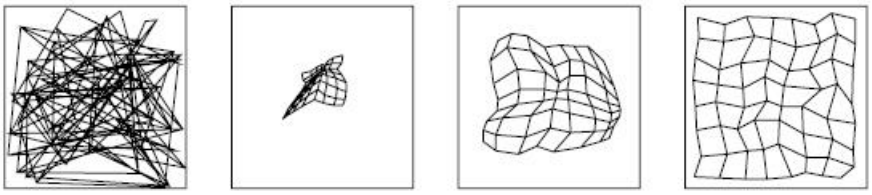


Figura 2.1: Ordenamiento de las entradas en el SOM. Las neuronas de salida se enlazan con sus vecinas de acuerdo al vector de pesos. En cada paso o iteración *t* se ve que el mapa converge y se conserva la topología.

único en donde se presenta el ordenamiento automático en el espacio de las representaciones de las entradas. Ese proceso se puede observar en la Figura 2.1. En la siguiente sección se explica a detalle el algoritmo SOM.

2.2. Definición

El algoritmo de mapas autoorganizados o SOM[4], desarrollado por Teuvo Kohonen a principios de los años 80[29], es uno de los algoritmos de redes neuronales artificiales más populares y poderosos.

Es una red neuronal única basada en aprendizaje no supervisado, en específico aprendizaje competitivo, que produce una representación de baja dimensionalidad (típicamente dos) de un conjunto de datos de entrada multidimensionales. A esta representación se le llama mapa. Este proceso de reducción de dimensionalidad se puede ver como característica de la compresión de la cuantificación vectorial.

Además el algoritmo almacena la información de una manera que se mantienen las relaciones topológicas de los datos de entrada. También se puede decir que el SOM es una aproximación a la función de densidad de probabilidad de los datos de entrada. Al generar mapas bidimensionales son muy útiles para la visualización de los datos de entrada.

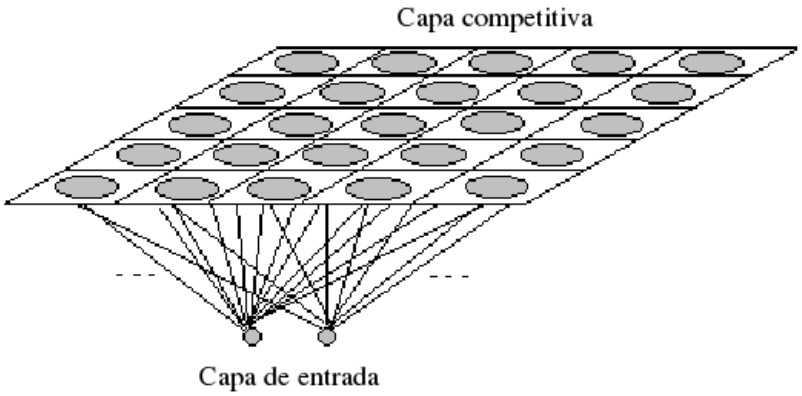


Figura 2.2: Red de Kohonen, se muestra la capa de entrada y la capa competitiva o mapa.

2.3. Arquitectura

Estructuralmente el SOM consiste solamente de dos capas: una capa unidimensional de neuronas de entrada y una capa de neuronas de salida llamada *capa competitiva*; esta última organizada en una malla bidimensional o mapa. En la figura 2.2 se muestra una red con 2 neuronas de entrada y 25 neuronas en la capa competitiva.

Cada neurona de entrada está conectada a cada neurona de la capa competitiva.

La capa de entrada modela un vector de características, cada neurona o nodo es un modelo de una observación, y tiene la misma función que en otros tipos de redes. Las neuronas de la capa competitiva tienen una propiedad diferente pues pueden encontrar las relaciones entre los patrones de entrada que procesan.

La capa de salida puede formar una malla con conexiones rectangulares, como en la figura 2.3, o hexagonales, como en la figura 2.4. Se puede decir que un mapa tiene topología rectangular si la vecindad de una neurona es de forma rectangular, y similarmente para una topología hexagonal. Normalmente se prefieren los mapas con topología hexagonal.

En la figura 2.5 se presenta una malla rectangular con una neurona o unidad U_c y su vecindad rectangular resaltada.

La capa de salida o mapa puede tener forma rectangular, cilíndrica o toroide. Es cilíndrica si un lado está conectado con el opuesto, es decir, si la vecindad de una neurona en el borde toma en cuenta las neuronas opuestas a ese borde. Es

[4] *Self-Organizing Maps*

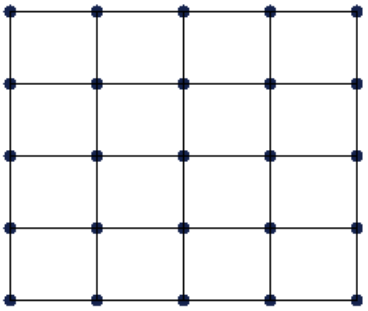


Figura 2.3: Malla rectangular

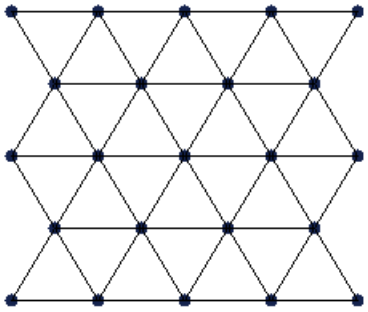


Figura 2.4: Malla hexagonal

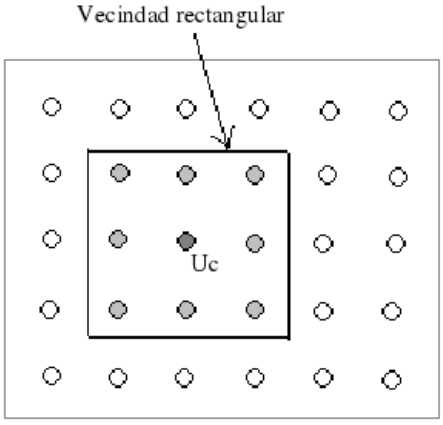


Figura 2.5: Vecindad rectangular de la unidad Uc

toroide si los lados opuestos están conectados, o lo que es lo mismo, si el mapa no tiene bordes.

2.4. El algoritmo básico

El SOM, por ser un algoritmo de aprendizaje no supervisado, no necesita que se le especifique una salida esperada.

Cada neurona tiene una posición (x,y) en la malla y tiene asociado un vector de pesos de la misma dimensión que los vectores de entrada.

Sea n el número de neuronas en la capa de entrada y $x = (x_1, x_2, ..., x_n) \in R^{\mathrm{n}}$ el conjunto de valores de las neuronas de entrada. El vector x es un patrón de entrenamiento cualquiera de dimensión n.

Sea m el número de neuronas en la capa competitiva y $u = (u_1, u_2, ..., u_m)$ el conjunto total de neuronas en la capa competitiva. Llamamos u_i a la i-ésima neurona en esta capa.

Cada neurona u_i de la capa competitiva recibe n conexiones (de las n neuronas de la capa de entrada), cada una con una peso asociado. El conjunto o vector de pesos que recibe u_i es $m_i = (m_{i1}, m_{i2}, ..., m_{in})$.

Al principio se inician aleatoriamente los vectores de pesos de cada neurona.

Se toma como entrada un patrón de entrenamiento seleccionado aleatoriamente. Este patrón es presentado a la capa de entrada, y la idea es encontrar una neurona competitiva cuyo vector de pesos permita que se asemeje más o esté más cercana al patrón de entrada. Esta se conoce como la neurona competitiva ganadora o BMU[5]. Una vez seleccionada la neurona competitiva ganadora, se selecciona una vecindad en la cual serán actualizados los vectores de pesos. La vecindad se define de acuerdo a una función de vecindad. El tamaño o radio de la vecindad y la función de vecindad son decrecientes en cada paso convenientemente seleccionados. Se actualizan los vectores de pesos de la vecindad, tal que queden más cercanos al vector de entrada.

Se realiza este procedimiento para todos los patrones de entrada un cierto número de pasos predefinido.

Más a detalle:

Sea $x = (x_1, x_2, ..., x_n)$ un patrón de entrada y $m_i = (m_{i1}, m_{i2}, ..., m_{in})$ el vector de pesos que recibe la neurona competitiva u_i desde la capa de entrada.

[5] *Best matching unit.*

Para cada paso $t = 0, 1, ..., T$, se selecciona, secuencial o aleatoriamente, un patrón de entrada x y se busca una neurona competitiva ganadora, denotada por m_c, tal que la distancia de x a ella sea mínima (ecuación 2.1):

$$\|x(t) - m_c(t)\| \leq \|x(t) - m_i(t)\| \forall i \tag{2.1}$$

o bien (ecuación 2.2):

$$\|x(t) - m_c(t)\| \leq \min_i(\|x(t) - m_i(t)\|) \tag{2.2}$$

donde $\|\cdot\|$ denota la norma de $x - m_i$, o la distancia euclidiana de la neurona u_i dado un patrón de entrada x, definida en la ecuación 2.3:

$$\|x - m_i\| = \sqrt{\sum_{j=1}^{n}(x_j - m_{ij})^2} \tag{2.3}$$

Después, se define la vecindad de la neurona ganadora m_c, y se actualizan los vectores de pesos de las neuronas en dicha vecindad, de acuerdo a la ecuación 2.4.

$$m_i(t+1) = m_i(t) + h_{c,i}(t) * (x(t) - m_i(t)) \tag{2.4}$$

donde $h_{c,i}(t)$ es la *función de vecindad* centrada en la neurona ganadora m_c. Es una función decreciente que depende de la distancia entre las unidades $i-$ésima y $c-$ésima y del tiempo t. Esta función define la región de influencia que tiene el patrón de entrada sobre el SOM. Puede definirse como la gaussiana de la ecuación 2.5.

$$h_{c,i}(t) = \alpha(t) * e^{\frac{-\|r_i - r_c\|^2}{2\sigma^2(t)}} \tag{2.5}$$

donde:

- $0 < \alpha(t) < 1$ es el factor de aprendizaje en el tiempo t, y es una función decreciente con t. Comúnmente se utilizan una función lineal $\alpha(t) = \alpha(0)(1 - \frac{t}{T})$ (T es el número de pasos del aprendizaje) o una inversa $\alpha(t) = \frac{A}{t+B}$ (A y B son constantes).
- $r_i \in R^2$ es la localización vectorial de la unidad i en la malla.
- $\sigma(t)$ es el radio de la vecindad en el tiempo t. Es también una función decreciente y puede definirse similarmente a α. Usualmente comienza con un valor grande.

La idea de la gaussiana es que las neuronas más cercanas a la ganadora serán más afectadas en la actualización de su vector de pesos.

Se repite el proceso de seleccionar un vector de entrada, encontrar la neurona ganadora y actualizar los pesos, para cada t, hasta cumplir T pasos, ciclos, iteraciones o épocas[6].

Usualmente el proceso de entrenamiento se realiza en dos fases: la primera, una fase de ordenamiento, con valores iniciales de α y σ relativamente grandes, y la segunda fase, de convergencia, con valores pequeños. Esto corresponde a actualizaciones mayores al principio y actualizaciones más finas al final.

El algoritmo SOM puede tender gradualmente a un estado en el que los patrones de activación de los vectores de entrada estén ordenados, despúes de cierto número de pasos, y escogiendo adecuadamente los parámetros α y σ.

El SOM produce una matriz de vectores de pesos tales que la distancia a cada vector de entrada es mínima. Esto permite obtener como salida, para cada vector de entrada, su ubicación (x,y) en el mapa bidimensional tal que la distancia entre ese vector de entrada y el vector de pesos asociado a la ubicación (x,y) es mínima.

Como medida de error se puede calcular el promedio de las distancias mínimas de los n vectores de entrada. A ese valor se le conoce como error de cuantificación[7][47]. Hay otras medidas de calidad de los mapas, pero esta se usa tradicionalmente en los algoritmos de cuantificación vectorial y agrupamiento. Está relacionada con el tamaño del mapa, en general si el mapa crece, el error de cuantificación decrece. Esto quiere decir que en mapas pequeños se generan pocos grupos, más generales, y en mapas grandes se generan más grupos, más compactos.

Recomendaciones

Es recomendable tomar ciertas precauciones en la definición de los valores de los parámetros de entrada del algoritmo.

En general se prefiere una malla con topología hexagonal en un mapa rectangular, no cuadrado, es decir que las m neuronas del mapa estén acomodadas en una malla de $p \times q$ tal que $p > q$.

Para la primera fase, de entrenamiento, se recomienda usar pocos vectores de entrada, tal vez seleccionados aleatoriamente del conjunto completo de entrada. Para la segunda fase, se recomienda usar el conjunto completo, valores menores para α y σ y un mayor número de ciclos o *epochs*.

[6] *epochs*
[7] Quantization error

Se recomienda hacer varias corridas del algoritmo, con diferentes valores para los parámetros, e irlos adecuando, de tal forma que el error de cuantificación sea cada vez menor. Se debe tomar en cuenta que el error de cuantificación no se puede usar para comparar mapas de diferente tamaño. Además, hay que tener cuidado pues incrementar el tamaño del mapa para minimizar el error de cuantificación puede afectar su topología[47].

Si la varianza o dispersión de los valores de los componentes de los datos de entrada es muy alta, se recomienda escalar o normalizar los datos. Esto previene que una variable domine en el modelo y haya una orientación en el mapa resultante[25].

Variantes

Hay variantes simples en las que solamente se ajustan ciertos parámetros del algoritmo básico[32]. Por ejemplo, la función de vecindad, o las funciones de decrecimiento para α y σ.

Una definición más sencilla de $h_{c,i}(t)$ es la mostrada en la ecuación 2.6:

$$\begin{aligned} h_{c,i}(t) &= \alpha(t) \text{ si } \|r_i - r_c\| < \sigma(t) \\ &= 0 \text{ en otro caso} \end{aligned} \tag{2.6}$$

Esta función de vecindad es constante para las neuronas en la vecindad de radio σ de la neurona ganadora en el tiempo t, y 0 para las neuronas fuera de la vecindad. Esto nos dice que solamente se actualizarán los vectores de pesos de las neuronas en la vecindad y los demás no serán afectados.

Se puede usar también otra función de distancia, en lugar de la distancia euclidiana, por ejemplo una distancia euclidiana ponderada[25].

Algunas variantes intentan mejorar la preservación de la topología usando estructuras más flexibles, y no un mapa fijo. El problema es que la visualización no es fácil[5].

Otras variantes intentan agilizar la búsqueda de la neurona ganadora, por ejemplo usando un SOM jerárquico. En este último se construye una estructura en forma de árbol donde cada capa incluye un SOM por separado. El árbol es entrenado nivel por nivel, en cada paso la búsqueda de la neurona ganadora se restringe a un subconjunto de acuerdo a la neurona ganadora del nivel anterior. Otra opción puede ser que los SOM de las capas inferiores trabajen con subconjuntos de los componentes de los vectores de entrada, y sus salidas se combinen en los SOM

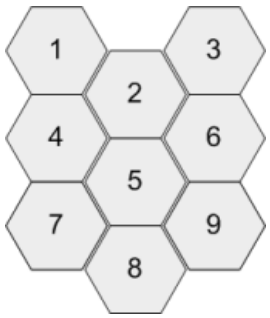


Figura 2.6: Mapa con unidades hexagonales

de las capas superiores[28]. Estos mapas jerárquicos se han usado mucho en reconocimiento de voz, donde cada capa trata con unidades como fonemas, sílabas, palabras.

Existe una versión del algoritmo conocida como *batch*[31], en la que los pesos de la vecindad son actualizados por cada época, acumulando las actualizaciones de cada vector de entrada. Esto permite que este algoritmo se pueda paralelizar. Para las versiones paralelas del SOM hay dos opciones: segmentar la red, o bien los datos de entrada. Se ha visto que esta segunda opción permite mayor escalabilidad[33].

Cada una de estas variantes busca mejorar el algoritmo, en cuanto a agilizar el tiempo de cómputo. Debe tenerse cuidado en la aplicación de las variantes, en el sentido de que no funcionan en general para todos los problemas, y se debe comparar la calidad de los resultados para problemas específicos[28].

Complejidad

La complejidad computacional del algoritmo básico es lineal respecto al número de datos de entrada, y es cuadrática con respecto al número de unidades en el mapa. Esto es, la complejidad del algoritmo es $O(n^2)$[60, 13].

2.5. Visualización

Una representación común del mapa autoorganizado es la matriz de distancia unificada o U-matrix[8], que visualiza la distancia entre las neuronas. Fue propuesto

[8] *Unified distance matrix*

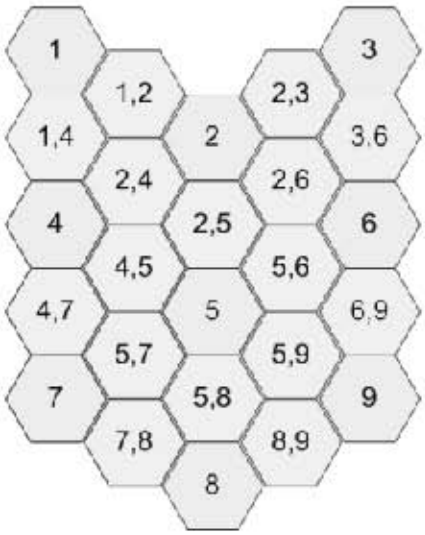


Figura 2.7: Visualización U-matrix del mapa de la figura 2.6. El par x, y es la distancia entre las neuronas x y y.

por Ultsch y Siemon en 1990 [59]. Se calcula la distancia entre neuronas adyacentes y se presenta en diferentes tonos de gris o colores. Un tono obscuro significa mayor distancia. Un tono claro significa que los vectores están más cerca. Se pueden ver las áreas claras como grupos y las obscuras como separadores. En la figura 2.7 se muestra cómo construir el mapa U-matrix a partir del mapa con unidades hexagonales mostrado en la figura 2.6. En las figuras 2.8 y 4.2 se muestran mapas tipo U-matrix.

Otra representación es el llamado mapeo o proyección de Sammon, propuesto por John W. Sammon en 1969. Es un algoritmo que sigue un enfoque de escalamiento multidimensional no lineal. Hace un mapeo no lineal de un conjunto de puntos en un plano tratando de preservar la distancia relativa entre ellos. Las relaciones topológicas pueden ser representadas con líneas entre dos neuronas vecinas. La generación de un mapa de Sammon puede ser pesada computacionalmente. En las figuras 2.9 y 4.3 se ven mapas de Sammon.

Otra forma útil de visualizar los datos es la representación de los planos componentes. Cada plano muestra la distribución relativa de cada componente de los vectores de datos. En esta representación, colores obscuros representan valores relativamente pequeños, y colores claros representan valores relativamente grandes. Se puede decir que dos componentes se correlacionan si su representación se ve similar. En las figuras 2.10 y 4.4 se observan mapas de los componentes.

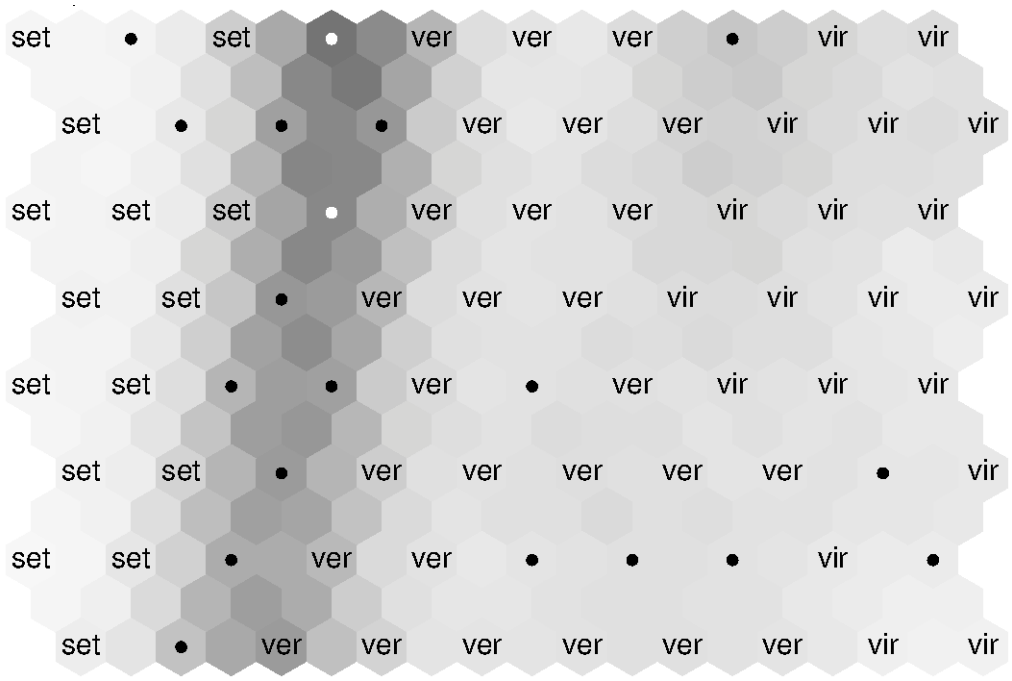


Figura 2.8: Visualización U-matrix del conjunto de datos IRIS.

2.6. Ejemplo

Para mostrar lo que puede hacer el SOM usamos el conjunto de datos de Iris[9][15, 48]. Éste contiene 150 registros de datos de tres especies de flores del mismo género *Iris*: *Iris setosa*, *Iris virginica* y *Iris versicolor*. Los datos son representaciones de las flores tomando en cuenta 4 características: longitud y anchura de los sépalos y pétalos. Así, cada registro es un vector de 4 componentes. Se sabe de antemano que una clase es linealmente separable de las otras dos. El SOM lo confirma y permite visualizarlo.

En este ejemplo se representa las especies en los datos de entrada etiquetados como *set*, *vir* y *ver*, respectivamente.

Se usó el paquete SOM_PAK[43] descrito en el Apéndice B, de la Universidad de Tecnología de Helsinki para ejecutar el algoritmo y obtener las visualizaciones U-matrix y Sammon.

En la figura 2.10 se ven los planos de los 4 componentes de los datos de entrada. Se puede ver que los componentes 1, 3 y 4 están más correlacionados entre

[9]Descargado de https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data

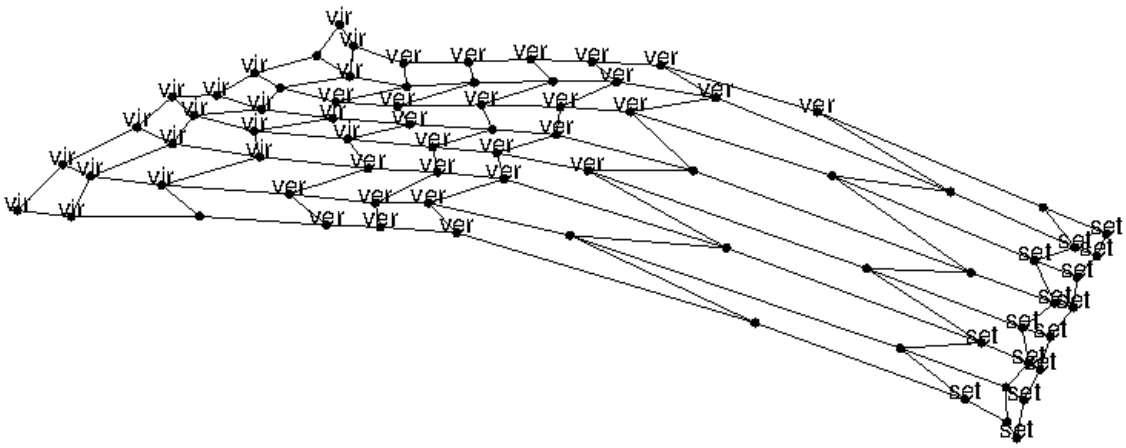


Figura 2.9: Mapeo Sammon del conjunto de datos IRIS.

sí.

En la figura 2.8 se muestra la visualización U-matrix de los datos. Se ve claramente una zona obscura que separa los datos etiquetados como *set* a la izquierda de los demás. La zona clara de la derecha muestra también la separación de los datos *ver* y *vir*.

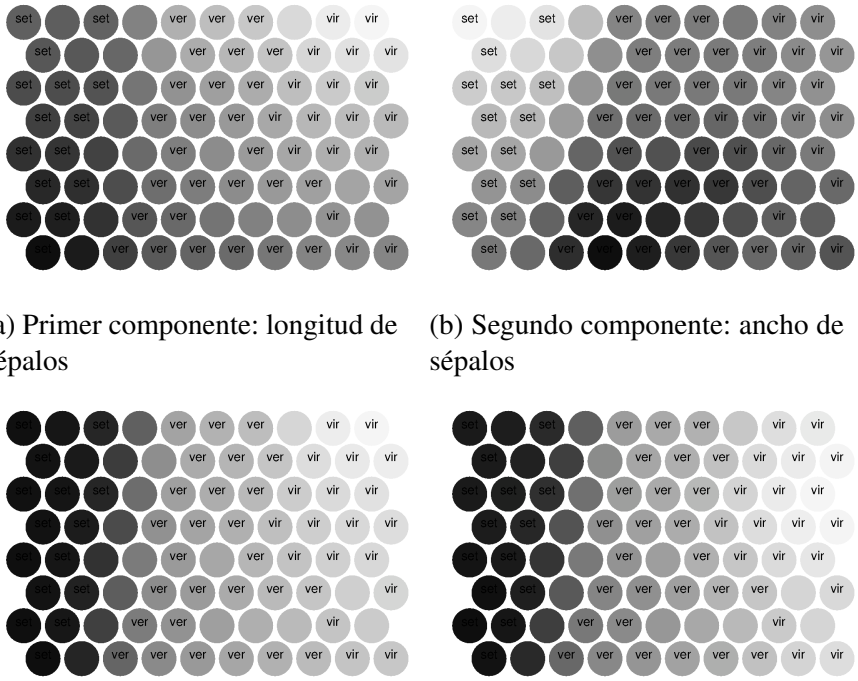
En la figura 2.9 se muestra el mapa Sammon de los datos, también se ve una separación clara de los datos *set* de los demás.

2.7. Aplicaciones

El algoritmo de mapas autoorganizados crea modelos o abstracciones de diferentes tipos de datos, por lo que se aplica a cualquier conjunto de datos en general. Inicialmente era para aplicaciones en la ingeniería, pero actualmente su uso va desde campos como la medicina y biología hasta la economía.

La mayor utilidad del SOM es que organiza los datos (o su representación o modelo) en una malla bidimensional que permite un despliegue gráfico para la visualización. Eso lo hace ideal para la exploración de datos, con aplicaciones por ejemplo en datos socio-ecónomicos, análisis financieros, y para la recuperación de información.

El SOM puede ser usado en combinación con redes supervisadas. Una red supervisada requiere un objetivo, lo que muchas veces no está disponible en los datos existentes. Una red no supervisada puede usarse para definir ese objetivo.



(a) Primer componente: longitud de sépalos

(b) Segundo componente: ancho de sépalos

(c) Tercer componente: longitud de pétalos

(d) Cuarto componente: ancho de pétalos

Figura 2.10: Planos de los 4 componentes del conjunto de datos IRIS, numerados del 1 al 4 de izquierda a derecha y de arriba a abajo.

Se entrena la red no supervisada y cada grupo posible pasa a ser un objetivo de la red supervisada.

Algunos ejemplos de aplicaciones son:

- Limpieza de datos para minería de datos: los datos extraídos de una base de datos a analizar pueden ser agrupados y el análisis puede hacerse sobre los grupos en lugar de sobre la base de datos completa.
- Mercadotecnia: por ejemplo, para detectar grupos de consumidores con ciertas características para dirigir hacia ellos la nueva campaña.
- Descubrir anomalías: en el monitoreo del funcionamiento de una máquina, puede detectarse comportamiento anómalo clasificando el comportamiento normal en grupos de funcionalidad, cuando un resultado no cae en ningún grupo.
- Descubrimiento de conocimiento en bases de datos bibliográficas. Por ejemplo, la herramienta ViBlioSOM[55]
- Análisis del genoma, usando índices o frecuencias de las bases en la secuencia. En este trabajo se hace una aplicación de este tipo, para lo cual, en el siguiente capítulo, se dará una descripción del genoma y conceptos relacionados.

Capítulo 3

El genoma

En este capítulo se da una breve introducción histórica al área de la biología que estudia los genomas, se mencionan los concéptos básicos de la genómica y algunos estudios en el área. No se entra al detalle biológico, sólo se menciona lo necesario para el análisis de secuencias objeto de este trabajo.

3.1. Historia

La genética en principio comenzó intentando definir las propiedades del material que contiene la información hereditaria, y cómo se usa esa información. Después se descubrió que ese material genético consiste de un tipo particular de molécula, diferente a otras moléculas presentes en los organismos, y se empezó a estudiar las características complejas de los organismos con base en las propiedades de las moléculas que los componen.

En este sentido hay importantes preguntas, por ejemplo, cómo se reproduce tan bien el material genético que se hereda de generación en generación, y cómo es que esa información especifica la construcción de las estructuras que componen un organismo vivo.

La herencia es un concepto conocido por mucho tiempo. La noción de que los hijos se parecen a los padres es clara. Hay características que se transmiten de generación en generación: la apariencia física, el temperamento y algunos otros rasgos. Pero el concepto de algún elemento o componente hereditario se concibió hasta mediados del siglo XIX.

Gregor Mendel, con base en experimentos con plantas, vió que había ciertas reglas hereditarias que se podían definir con cierta álgebra[39]. El llamó a esos

paquetes "elementos", lo que ahora conocemos como genes.

Un poco más tarde, en 1869, Friedrich Miescher, un bioquímico suizo, identificó y aisló un componente ácido, rico en fósforo, aparentemente hecho de moléculas grandes y llamó a esa sustancia *nucleína*[12]. Es lo que ahora se conoce como ADN o ácido desoxirribonucléico. Unos años más tarde un discípulo suyo, Richard Altmann, introdujo el término *ácido nucléico*. Más tarde, en 1878, Albretch Kossel aisló las bases nucléicas primarias.

Miescher incluso escribió de forma no muy precisa que algunas moléculas compuestas de la repetición de pequeñas piezas similares podrían expresar la variedad hereditaria, como las palabras expresan conceptos usando pocas letras[12].

A principios del siglo XX se redescubrieron las ideas de Mendel y se habían identificado los tres componentes de los ácidos nucléicos: un azúcar, un fosfato y una base nitrogenada.

En 1927, Nikolai Koltsov propuso que los cromosomas son moléculas gigantes de proteínas, compuestas por dos tiras que se replican por complementariedad.

En 1943, mediante experimentos realizados por Oswald Avery, se vió que ciertos rasgos podían ser transferidos y heredados, y se identificó la nucleína como la portadora de la información hereditaria (previamente se pensaba que era en proteínas).

En 1944, Erwin Schröndinger escribió que los cromosomas mantenían el código o guión hereditario y además la función principal en la aparición de rasgos. Su tarea de analizar la estructura de los cromosomas y encontrar cómo hacen su trabajo animó a muchos, incluido James Watson. Schröndinger además postuló que las propiedades del material genético debían ser explicadas con nuevas leyes de la física, relacionó la vida con las leyes de la termodinámica, diciendo que un sistema biológico amplía su complejidad exportando su entropía[54].

Linus Pauling descubrió en 1948 que muchas proteínas contienen formas helicoidales, a partir de observaciones con rayos X de moléculas de hemoglobina.

El biofísico Maurice Wilkins y la bioquímica Rosalind Franklin examinaron patrones que se obtenían con técnicas de difracción de rayos X en fibras de ADN.

En 1947, el químico austriaco Erwin Chargaff descubrió que las proporciones de las cuatro bases nitrogenadas varían de una muestra de ADN a otra, pero que para ciertos pares específicos de bases, adenina con timina y citosina con guanina, la proporción de los dos nucleótidos era siempre igual, es decir, que la cantidad de adenina es igual a la de timina, y la cantidad de citosina, es igual a la de guanina.

Francis Crick (físico y biólogo) y James Watson (zoólogo) fueron motivados por el descubrimiento de Chargaff, y encontraron que cambiando la dirección de una tira de polinucleótidos y emparejándola con otra se podía explicar el descubri-

miento de Chargaff. También se apoyaron en los descubrimientos de Pauling y de forma considerable en los trabajos de Franklin, en imágenes de rayos X[1] de alta calidad que mostraban patrones que les ayudaron a obtener las medidas exactas de la hélice. Watson y Crick no conocían el trabajo de Koltsov. En 1953 publicaron su ahora famoso artículo "A Structure for deoxyribose nucleic acid"[62].

Con esa estructura, se explica además el mecanismo de copiado o replicación del material genético, el fenómeno de la división celular a nivel cromosómico. Posteriormente se descifró el código genético, las reglas para la síntesis de proteínas a partir del ADN.

En los años 70 comenzó la secuenciación del ADN, impulsada por métodos de Frederick Sanger y Walter Gilbert. En 1981 se obtuvo la secuencia del primer organelo, de una mitocondria humana. En 1995 se obtuvo la primera secuencia completa de todo el genoma de un organismo, una bacteria. Antes de mediados de los años 90 no se tenían secuencias completas de genomas y se estudiaba más la actividad individual de los genes o a lo más de pequeños grupos de genes. Con los avances en la secuenciación de genomas, se sigue estudiando la expresión y regulación, pero ahora más del genoma completo como un todo.

A partir de la secuenciación de muchos organismos se ha generado una gran cantidad de datos, y el análisis de estas secuencias genómicas se ha vuelto muy importante en la investigación en biología y áreas relacionadas.

3.2. ADN

El ADN o ácido desoxirribonucléico constituye el material genético de los organismos. Se compone de dos moléculas polímeras hechas de cadenas lineales de nucleótidos.

Cada nucleótido tiene 3 partes: un fosfato, un azúcar y una base nitrogenada. La base es la parte del nucleótido que lleva la información genética y puede ser púrica como adenina (A) o citosina (C), o una base pirimidínica como guanina (G) o timina (T).

Los nucleótidos se unen mediante un enlace fosfodiéster para formar un polinucleótido (o polímero o molécula). Este polímero puede tener varios millones de nucleótidos.

El ADN se forma con la unión de dos de estas largas hebras de polinucleótidos, enrolladas una con la otra corriendo en sentido contrario[2] en una estructura de

[1] Una foto, ahora famosa, conocida como *foto 51*.

[2] Se dice que van del extremo 5'fosfato al extremo 3'hidroxil en direcciones opuestas.

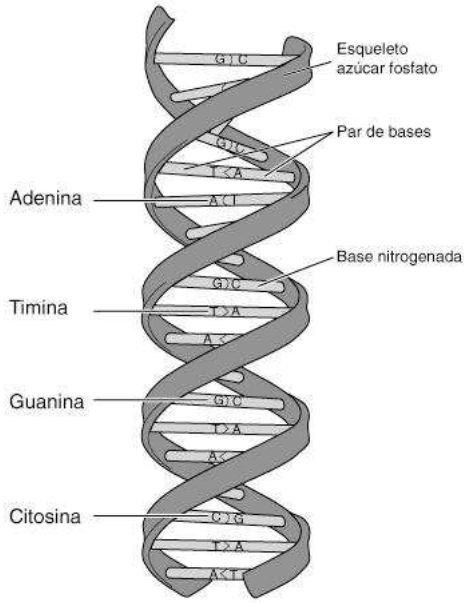


Figura 3.1: La doble hélice[23].

doble hélice, que tiene alrededor de 10 pares de nucleótidos enlazados por sus bases nitrogenadas en cada vuelta. La molécula del ADN parece una escalera retorcida cuyos lados son las moléculas de azúcar y fosfatos y cuyos peldaños son las bases nitrogenadas unidas, como se ve en la figura 3.1.

La unión de las bases de los nucleótidos se da mediante enlaces de hidrógeno y con las reglas siguientes, que se ilustran en la figura 3.2:

- Adenina (A) se enlaza con Timina (T) mediante 2 puentes de hidrógeno. Se dice que forman un enlace débil.
- Citosina (C) se enlaza con Guanina (G) mediante 3 puentes de hidrógeno. Forman un enlace fuerte. Se podría decir que las secuencias con más C y G serían más estables porque tienen más puentes de hidrógeno.

Esto implica que las dos hebras de polinucleótidos son complementarias, por lo que si se tiene una de ellas, se puede conocer la otra.

La molécula de ADN es muy larga, y se enrolla o contorsiona sobre sí misma en conformaciones que aún se están estudiando.

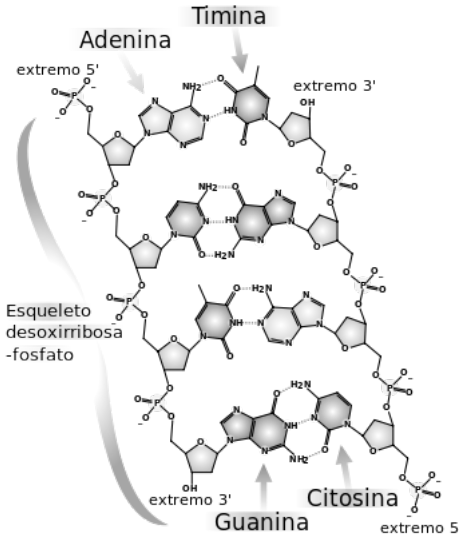


Figura 3.2: Detalle de la molécula de ADN. Se muestran las bases nitrogenadas A, T, G, C, los puentes de hidrógeno, y los enlaces fosfodiéster entre los nucleótidos[8].

Los investigadores miden la longitud de la molécula contando el número de bases en una hebra que se unen para formarla. La unidad que se utiliza es el par de bases o *bp*, y comúnmente se usan *Kbp* para mil pares y *Mbp* para un millón.

El ADN contiene genes, que son subintervalos de la molécula.

3.3. Genes

Un gen es la unidad fundamental de la herencia. Los genes determinan las características físicas de un organismo, por ejemplo, puesto de la manera más sencilla, un gen puede especificar el color del cabello, otro gen especifica el tamaño de la boca. A esto se le llama fenotipo, el rasgo físico que expresan los genes. El genotipo se refiere a la estructura biológica química de los genes.

El número de genes varía de una especie a otra. Se pensaría que el número de genes es mayor en organismos más complejos, pero esto no es siempre así: hay bacterias que tienen miles de genes, el ser humano parece tener no más de treinta mil.

Todos los individuos de una misma especie tienen los mismos genes. Lo que hace diferentes a los individuos es que un gen puede tener diferentes formas o ale-

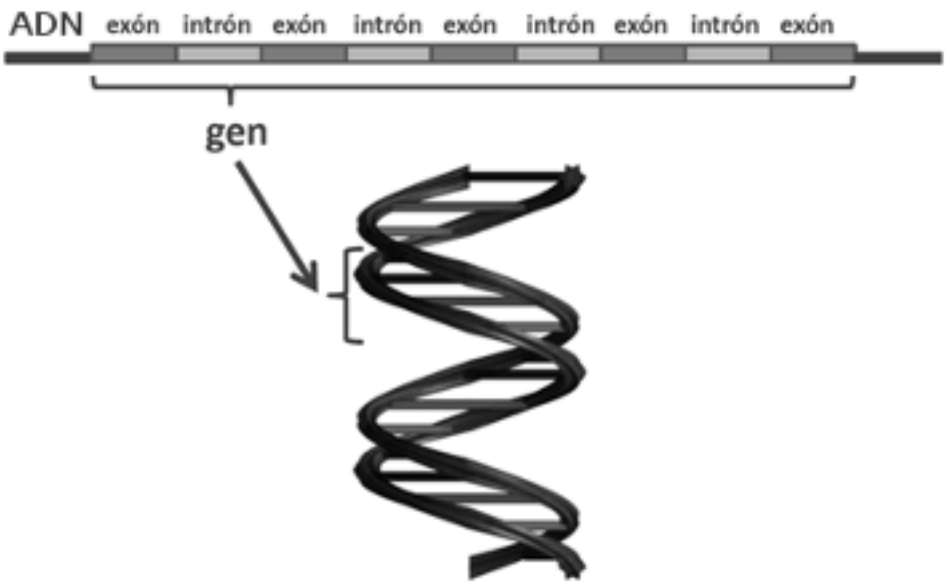


Figura 3.3: El gen en la secuencia de ADN, mostrando exones e intrones.

los. Cada uno recibe dos copias de cada gen, una de cada padre. Lo que determina las características del individuo es cómo interactúan esas dos copias.

Se habla de características o rasgos dominantes, que son los que se expresan más, y de rasgos recesivos, los que se expresan menos. Esto explica el mecanismo más básico de herencia. Sin embargo, generalmente es más complicado. Por ejemplo, un gen puede definir varios rasgos, un rasgo puede ser definido por varios genes, y los rasgos pueden ser influenciados también por el ambiente o entorno del individuo, como la alimentación o el hábitat.

Composición

Un gen es un segmento del genoma. Los genes se encuentran en los cromosomas.

Un gen puede ser discontinuo, es decir, tener intervalos de ADN que no forman parte del gen en sí. Las partes que forma el gen se llaman exones, y las que no, se llaman intrones.

El gen se transcribe en ácido ribonucléico o ARN, que a su vez se puede

traducir en una proteína[3]. En las partes del gen que se traducen a proteínas, cada triplete del ARN transcrito se llama *codón*, y se traduce a un aminoácido.

Función

Los genes contienen las instrucciones para que la célula fabrique proteínas. Las proteínas son moléculas grandes y complejas que son muy importantes para el funcionamiento y mantenimiento del organismo.

Los genes afectan los rasgos o características físicas de forma indirecta diciendo qué proteínas fabricar, cuánto, cuándo y dónde.

La secuencia de los genes especifica la secuencia de las proteínas, y por lo tanto su estructura molecular.

La expresión génica se refiere al proceso de la conversión genómica a proteínas funcionales, o bien la interpretación del código genético para la síntesis de proteínas, lo que eventualmente da lugar al fenotipo del organismo.

Para el ADN este proceso originalmente se veía compuesto de dos partes:

- Transcripción: a partir del ADN se produce una copia en ARN del gen.
- Traducción: síntesis de la proteína cuya secuencia de aminoácidos es determinada (vía codigo genético) por la secuencia de nucleótidos del ARN de la transcripción.

La regulación de la expresión génica le permite a la célula tener control sobre su estructura y función, expresando las proteínas que necesite. Esta regulación es la base para la diferenciación celular, la morfogénesis, y la versatilidad y adaptabilidad de un organismo.

Se han descubierto grupos de genes con secuencias similares o idénticas, que se usan para cumplir una misma función. Estos grupos de llaman familias de genes. Los genes de una familia pueden estar cercanos o dispersos en el genoma. Las secuencias de los genes nos dicen que tienen un origen evolutivo común, nos dicen las relaciones dentro de la familia, pero también pueden indicar relaciones entre diferentes familias.

También puede haber genes contiguos, es decir que no haya o haya muy poco espacio entre ellos, que sean expresados como una unidad. Ese grupo de genes que expresan una misma función forman un *operón*.

[3] Si es el caso, el ARN se llama *mensajero*.

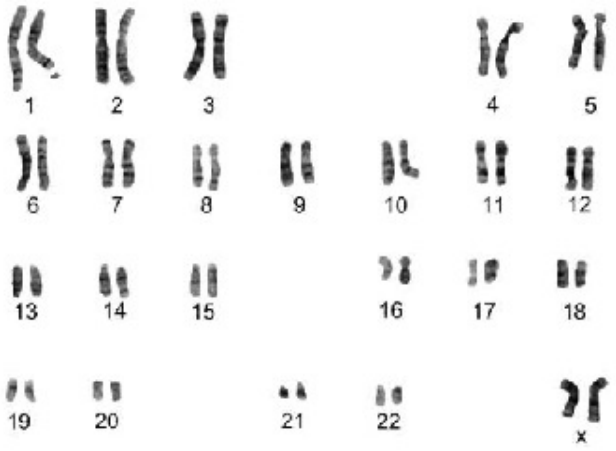


Figura 3.4: Cariotipo del humano (femenino)[14].

3.4. Cromosomas

La molécula de ADN contiene la información genética del organismo. Esta molécula está contenida en diferentes paquetes junto con otras proteínas localizados en la célula. Esos paquetes se llaman cromosomas y ayudan a mantener el ADN organizado y compacto.

En un mismo organismo, cada cromosoma tiene un cierto número de genes, y pueden tener diferente longitud. Se puede decir que un cromosoma afecta una característica del organismo por los genes que contiene. Los investigadores ordenan los cromosomas de acuerdo a su longitud, desde los más largos a los más cortos, en una tabla que se llama cariotipo. El cariotipo no entra en detalle a nivel de genes, pero puede servir para casos específicos para dar una idea de las características del organismo. En la figura 3.4 se muestra el cariotipo del ser humano.

Dentro del cromosoma, la molécula de ADN está doblada en una forma para que sea más compacta. Se piensa que ese empaquetamiento o conformación o estructura influye en la función de la información contenida en el ADN.

Los organismos más simples, como las bacterias, tienen el ADN en un sólo cromosoma, localizado en la parte central de la célula, llamada *nucleoide*. Los organismos más complejos tienen más de un cromosoma, el número depende de la especie. Especies más cercanas tienen un número parecido de cromosomas.

Sin embargo, no hay una regla más específica acerca de cuántos cromosomas debe tener un organismo. Podría pensarse que organismos más sencillos tendrían menos cromosomas, y los más complejos, más cromosomas, en una relación cre-

ciente con la complejidad. Pero esto no es así, por ejemplo, el ser humano tiene 46 cromosomas (agrupados en pares) y el pez dorado tiene 94.

3.5. El genoma

El genoma es la secuencia de ADN completa contenido en casi todas las células del organismo. El genoma comprende el material genético completo de un organismo, contiene todas las instrucciones necesarias para construir y mantener una instancia viviente de un organismo y para transmitir las características a la siguiente generación.

En la mayoría de los organismos, el genoma está formado de ADN. En algunos virus su genoma contiene ARN. El genoma contiene genes y se divide en cromosomas.

Cada especie tiene un genoma diferente, cuya longitud puede depender de la complejidad del organismo, aunque no necesariamente. También se puede decir que cada individuo de una especie tiene un genoma único, pues no hay dos individuos que tengan el genoma exactamente igual, aunque sí muy similar.

En general, los genes solamente ocupan una pequeña parte del genoma. Lo demás puede incluir:

- intrones, que son las partes intermedias que no codifican en un gen
- pseudogenes: copias no funcionales de un gen
- secuencias de repetición
- microsatélites: secuencias muy cortas repetidas muchas veces, que pueden funcionar como marcadores.

Para su estudio biológico, los organismos vivos se clasifican en una taxonomía de acuerdo características comúnes. Ha habido muchas clasificaciones propuestas a lo largo de la historia. En el siglo XVII el botánico Carl Linneo propuso una clasificación, pero sobre todo un sistema de nomenclatura binomial para los organismos. Este sistema de nomenclatura se sigue usando todavía. Actualmente una taxonomía aceptada habla de tres *dominios* principales[68]: *Archaea*, Bacteria (que tienen células procariontes) y Eukaryota (con células eucariontes). En general se puede decir que los dominios se dividen en *reinos*, y sucesivamente en *filos*, *clases*, *órdenes*, *familias*, *géneros* y *especies*. La clasificación se basa en cambios evolutivos y relaciones filogenéticas.

La estructura del genoma es diferente entre diferentes organismos vivos. En especial se diferencía en los genomas de los organismos eucariontes y procariontes (Archaea y Bacterias).

Genomas de eucariontes

Los organismos del dominio Eukarya tienen células con membranas que definen compartimientos como el núcleo y los organelos.

La estructura del genoma es similar en todos los organismos eucariontes. Se habla del genoma nuclear, localizado en el núcleo de las células, pero también hay genoma en los organelos, como las mitocondrias y los cloroplastos. Lo que varía entre organismos es el tamaño o longitud. Para el genoma nuclear va desde 10Mb hasta 100000Mb.

En muchos casos la longitud del genoma puede relacionarse con la complejidad del organismo. Los eucariontes más simples como los hongos tienen los genomas más pequeños, los más complejos como los vertebrados y las plantas tienen genomas más largos. Pero en eucariotes en general no se puede definir esa relación. Por ejemplo, hay organismos como peces, anfibios o plantas, que tiene genomas más largos que algunos mamíferos. Hay amebas que tienen el genoma 200 veces más largo que el humano. Y una ameba no es más compleja que un humano.

Se esperaría que hubiera una relación porque la complejidad del organismo se podría relacionar con el número de genes. Un organismo más complejo necesita más genes, por lo tanto más largo su genoma. Pero esta última relación con el número de genes no es precisa, y parecería que en los organismos más complejos se "desperdicia" mucho espacio en el genoma. En organismos sencillos, hay más genes si se compara con un segmento de la misma longitud de un organismo complejo. En organismos sencillos, los genes comúnmente no son discontinuos, o son menos discontinuos (casi no hay intrones), son más compactos, los espacios entre genes son más cortos que en organismos complejos. También hay muchas menos repeticiones en el genoma que en organismos complejos. La organización del genoma en organismos sencillos es más "económica".

Genomas de procariontes

Los organismos de *Archaea* y Bacteria tienen células procariontes, que no tienen un núcleo ni organelos diferenciados.

En comparación con los de eucariontes, los genomas de procariontes son más pequeños. Muchos genomas de procariontes son menores a 5Mb, aunque hay unos pocos que son más largos, hasta 30Mb.

También en comparación con los eucariontes, tienen menos genes. El genoma es más compacto aún que los organismos eucariontes más sencillos. Casi no hay intrones o puede no haber ninguno. En los genomas de procariontes hay muy pocas secuencias repetidas. Se cree que esta organización compacta es buena para las bacterias porque les permite una replicación rápida.

Generalmente está contenido en una sola molécula de ADN, y generalmente circular, en el nucleoide, una región un poco diferenciada de la célula. Sin embargo, las células de procariontes tienen genes adicionales en otras moléculas de ADN más pequeñas, circulares o lineales, que se llaman plásmidos. Algunos plásmidos se integran al genoma pero otros permanecen independientes. Los plásmidos contienen genes que en muchos casos no son esenciales para la bacteria.Muchos plásmidos son capaces de transferirse de una célula a otra, incluso hay plásmidos similares en diferentes especies. Esto sugiere que son entidades independientes, y su contenido genético no debería ser incluido en el genoma de la bacteria.

Por el hecho de ser más pequeños permiten un análisis de secuencia rápido, lo que ha resultado en la publicación de muchas secuencias completas de bacterias y *archaea* en los últimos años. Como consecuencia se entiende un poco más de los genomas de los procariontes y de los organismos en sí que de los eucariontes. Mucho de lo que se sabe de los procariontes viene del estudio de la *E. Coli*. Con estudios recientes se ha visto que el genoma de *E. Coli* es de los más simples. Se han encontrado cada vez más genomas lineales, y plásmidos que pueden considerarse esenciales.

3.6. Estudio del genoma

El estudio del genoma puede ser sobre una parte de él o sobre el genoma completo, se puede estudiar la secuencia de los genes, o la función de los genes, o las partes del genoma que regulan a los genes, o las partes del genoma que no son genes, o la localización de los genes en el genoma e investigar cómo se relacionan unos genes con otros, o el genoma desde el punto de vista de la estructura de la molécula donde está contenido.

Aunque se tienen ya muchos genomas completos, todavía se desconoce la función de muchos genes. De los genes cuya función se conoce, es interesante ana-

```
>gi|558678294|ref|NC_022873.1| Bacillus thuringiensis YBT-1518, complete genome
TTTGAATAAGTTTTCCACACCTTTATCTTATCCACAATTTGTGTATAACATGTGGACAGTTTTAATCACA
TGTGGGTAAATAGTTGTCCACATTTGCTTTTTTTTGTCGAAAACCCTATCTCATATACAAACGACGTTTTT
AGGTTTTAAAATACGTTTCGTATAAATATACATTTTATATTTATTAGGTTGTACATTTGTTGCGCAACCT
TATTCTTTTACCATCTTAGTAAAGGAGGGACACCTTTGGAAAATATCTCTGATTTATGGAATAGTGCCTT
AAAAGAATTAGAAAAAAAGGTAAGCAAGCCTAGTTATGAAACATGGTTAAAATCAACAACGGCTCATAAC
TTGAAAAAAGACGTATTAACGATTACAGCTCCAAATGAATTTGCTCGTGACTGGCTAGAATCTCATTACT
CAGAACTAATTTCGGAAACACTATACGATTTAACAGGGGCAAAATTAGCAATTCGCTTTATTATTCCCCA
AAGTCAATCGGAAGAGGACATTGATCTTCCTCCAGTTAAGCGGAATCCAGCACAAGATGATTCAGCTCAT
TTACCACAGAGCATGTTAAATCCAAAATATACATTTGATACATTTGTTATCGGCTCTGGTAACCGTTTTG
```

Figura 3.5: Ejemplo de un archivo de una secuencia en formato FASTA.

lizar cuántos están involucrados en una u otra actividad bioquímica, es lo que se conoce como catálogos de genes. También es interesante comparar los catálogos entre especies, y tratar de encontrar cuál puede ser el conjunto mínimo de genes similares para realizar alguna función; o bien cuáles son los genes que distinguen una especie de otra.

En la investigación genómica se usan diferentes métodos. En general se puede decir que se combinan resultados de mapeo y secuenciación genómica para entender su estructura y funcionamiento.

El mapeo pretende encontrar la ubicación de un gen, o la distancia relativa entre genes, en un cromosoma.

La secuenciación del ADN se refiere al proceso de determinar el orden de los nucleótidos en la molécula de ADN. Se puede usar para definir la secuencia de genes individuales, grupos de genes u operones, cromosomas, o genomas enteros. Se usan las bases nitrogenadas para identificar los nucleótidos. Como se mencionó anteriormente, el ADN se compone de las bases Adenina, Citosina, Guanina y Timina, y se usan las letras A, C, G y T respectivamente para representarlas. Esta representación fue formalizada en 1970 por la Unión Internacional de Química Pura y Aplicada o IUPAC[4][45]. Si en una posición de la secuencia no se tiene certeza o hay ambigüedad en qué base es, se usan otras letras, dependiendo de si puede ser una u otra base conocida.

Hay muchos métodos de secuenciación, y cada vez requieren más poder de cómputo. En resumen generan datos crudos como cadenas cortas que se sobrelapan, y éstas necesitan ser ensambladas en cadenas más largas para lograr la secuencia completa.

Para su procesamiento en la computadora, la secuencia se almacena en un

[4] *International Union of Pure and Applied Chemistry*

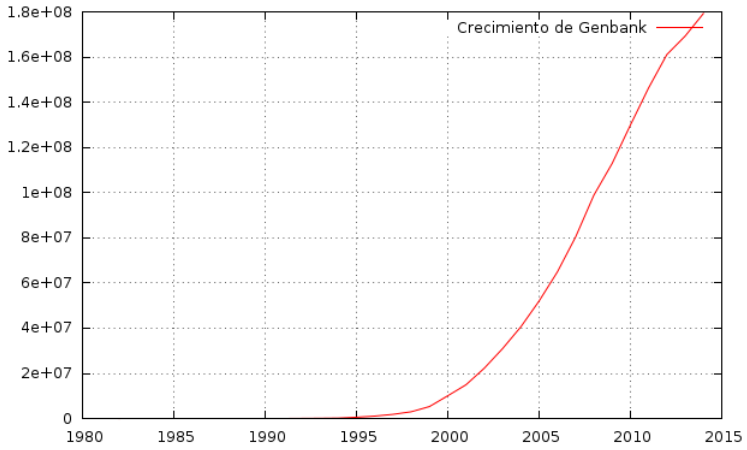


Figura 3.6: Crecimiento del número de secuencias por año en GenBank

archivo. El formato más común es conocido como FASTA. El formato FASTA tiene su origen en el paquete de programas de alineamiento de secuencias del mismo nombre, creado en 1985 por David Lipman y William Pearson[35]. Este formato es un archivo de texto plano con las siguientes características:

- la primera línea es para la descripción. Comienza con el símbolo "$>$" y le sigue un identificador de la secuencia. A continuación puede tener opcionalmente en la misma línea una descripción más completa.
- las demás líneas son la secuencia. Cada línea puede tener máximo 80 caracteres.
- la secuencia termina hasta que termina el archivo o hasta que se encuentra otra línea que comience con el símbolo "$>$".

En la figura 3.5 se muestra un ejemplo de una secuencia en formato FASTA.

El formato FASTA puede usarse para nucleótidos, comúnmente se usan archivos con terminación *.fna*, o para aminoácidos, en archivos con terminación *.faa*.

Hay otros formatos de secuencias genómicas, muchos relacionados con los programas con los que se manejan o analizan.

Las secuencias genómicas obtenidas se agrupan en bases de datos. La más conocida es la base de datos pública de GenBank, del Instituto Nacional de Salud de Estados Unidos. El número de secuencias en GenBank ha crecido exponencialmente como se muestra en la figura 3.6. Para darse una idea, GenBank contabilizaba aproximadamente unas 600 secuencias en 1982, 5.3 millones a fines

de 1999 y 182 millones en 2015[42]. Hay otras bases de datos como KEGG en Japón. También hay bases de datos especializadas en cierto organismo, especie, o familia; o en cierta función como la regulación.

3.7. Proyectos

La investigación del genoma ha tenido un gran impulso por el desarrollo de tecnologías de secuenciación y el incremento del poder de cómputo para el análisis de las secuencias.

Al principio se analizaban secuencias pequeñas, de bacterias, por lo que se sabe mucho de ellas. Después se empezaron a analizar secuencias más grandes, de eucariontes.

Un proyecto fundamental es el Proyecto Genoma Humano, concebido en 1984, iniciado en 1990 y completado en 2003. Fue coordinado por el Instituto Nacional de Salud (NIH) y el Departamento de Energía de Estados Unidos. El objetivo era obtener la secuencia completa del genoma humano e identificar los genes. En 2003 se publicó oficialmente la secuencia del genoma humano. Esto fue sólo el punto de partida para la investigación sobre el genoma humano. Un proyecto similar pero con fondos privados se inició en 1998 por la empresa Celera Genomics. Uno de los resultados obtenidos fue tener una idea más clara sobre el número de genes, anteriomente se esperaba que hubiera unos 80000 genes en el genoma humano, aunque se llegó a pensar que había hasta 150000. En 2001 se reportó que el humano tiene alrededor de 35000 genes[9]. Para 2004 se concluyó que hay aproximadamente unos 25000 genes. Actualmente se piensa que hay 20500 genes. Estos genes forman aproximadamente el 1.5 % de la longitud del genoma humano que es de 3200Mb.

En 2003 se creó un consorcio llamado ENCODE[5] para lanzar un proyecto para encontrar los elementos funcionales del genoma humano[10].

Otros proyectos actualmente analizan el proteoma de los organismos. El proteoma es el conjunto de proteínas expresados por los genes de un genoma. La ciencia que lo estudia es la proteómica. El proteoma es más grande que el genoma. Se puede estudiar el proteoma de una célula, o del organismo completo.

También se habla mucho del microbioma del humano, refiriéndose al conjunto de genomas de los microorganismos que viven en el cuerpo humano. Estos microorganismos tienen muchas funciones, por ejemplo para la digestión, prevención

[5] *ENCyclopedia Of DNA Elements*

de enfermedades, o síntesis de nutrientes y vitaminas esenciales. El número total de genes en el microbioma es cien veces mayor que el número de genes del humano. El estudio del microbioma como un todo es posible por el avance de las tecnologías genómicas. El proyecto del Microbioma Humano, del Instituto Nacional de Salud de Estados Unidos, busca identificar las comunidades de microbios en el cuerpo humano y encontrar correlaciones entre cambios en el microbioma y la salud humana.

Una tecnología que ha acelerado la investigación genómica son los microarreglos o *chips* de ADN. Se organizan en una malla bidimensional, en cada celda hay material biológico que permite la experimentación de la expresión génica. Se usan para medir los niveles de expresión de varios genes o varias regiones del genoma. Su principal característica y ventaja es que se pueden hacer análisis en paralelo y de forma masiva. Los microarreglos de ADN son los más usados y sofisticados, pero también hay microarreglos de proteínas y de péptidos.

En general se usan mucho las técnicas estadísticas y de aprendizaje de máquinas para el análisis de los datos genómicos. Debido al crecimiento exponencial de los datos, es natural el uso de estos métodos en el análisis del genoma.

En particular el algoritmo de los mapas autoorganizados o alguna variante o combinación con otros métodos, se han usado para el estudio de la expresión génica[56, 21], por ejemplo para la interpretación de patrones o en el uso de codones en genes de bacterias[24], así como también para el análisis de la frecuencia de los nucleótidos en la secuencia[1]. En el siguiente capítulo se mostrará una aplicación particular de los mapas autoorganizados sobre secuencias genómicas.

Capítulo 4

SOM para análisis del genoma

4.1. Antecedentes

En la investigación del genoma se han usado varios métodos de estadística, aprendizaje de máquinas, redes neuronales, agrupamiento y en particular mapas autoorganizados. Muchos de los análisis de los genomas se basan en la secuencia completa como tal, más que en los genes del mismo. Hay muchos trabajos sobre el análisis y estimación de *palabras* o subsecuencias de cierta longitud en la secuencia del ADN. La mayoría de los trabajos se concentran en secuencias cortas, como pares de nucleótidos o dinucleótidos, o tetranucleótidos[27, 17]. Algunos más analizan palabras más largas, por ejemplo de longitud 10 o 12[2]. Se usa también el término oligonucleótido[1] para referirse de manera general a estas palabras o subsecuencias, y la terminación *-mero* para referirse a la longitud, por ejemplo dímero, tetrámero, decámero.

Se han detectado abundancia de ciertos oligonucleótidos (por ejemplo del dímero CG) en las secuencias. Estas frecuencias características permiten definir la firma genómica de una secuencia. Se ha observado que las firmas genómicas de genomas relacionados filogenéticamente son similares [26].

Otros trabajos han definido índices del genoma de acuerdo a las frecuencias de los enlaces fuertes (CG o S) y débiles (AT o W), o bien distinguen entre bases purínicas (AG o R) y pirimidínicas (CT o Y). Se han relacionado estas distribuciones con la estructura y el origen evolutivo de los organismos[41].

Algunos otros han analizado la frecuencia de nucleótidos en organismos es-

[1]Los oligonucleótidos son pequeñas moléculas de ADN que pueden sintetizarse en el laboratorio para estudiar la expresión génica

pecíficos, por ejemplo en el humano[36].

Como se mencionó en el capítulo anterior, el algoritmo de los mapas autoorganizados se ha usado para el estudio de la expresión génica[56, 21], la interpretación de patrones, el análisis del uso de codones en genes de bacterias[24], y también en el análisis de la secuencia de acuerdo a la frecuencia de los nucleótidos en ella[1, 6].

En este capítulo se describe un trabajo de análisis de un conjunto de genomas, usando una representación basada en la frecuencia de dinucleótidos en la secuencia, mediante la aplicación del algoritmo SOM para encontrar agrupaciones que puedan indicar similitudes entre secuencias.

4.2. Datos de entrada

Se usó un conjunto secuencias de nucleótidos de genomas completos de bacterias disponibles en el servidor FTP del Centro Nacional de Información Biotecnológica (NCBI, por sus siglas en inglés) del Instituto Nacional de Salud (NIH) de Estados Unidos [16].

Se descargaron los archivos en formato FASTA (ver figura 3.5). Estos son los archivos con terminación *.fna*. Se tomaron en cuenta solamente los genomas completos, sin incluir genomas de plásmidos ni de cromosomas. En total fueron 645 genomas completos diferentes en igual número de archivos (Abril 2015). Cada archivo tiene una sola secuencia. En el Apéndice C se muestra la lista de todos los organismos incluidos.

Lo que se hizo primero fue representar cada genoma como un vector de entrada.

Representación

Se usó una representación en forma de vector para cada genoma, basada en la estructura secuencial del genoma. Se usará el término *cadena* para referirse a la secuencia completa del genoma, y *símbolo* para referirse a una base nitrogenada.

Para cada genoma, se obtiene la probabilidad de ocurrencia o frecuencia de cada símbolo de la cadena, de acuerdo a la ecuación 4.1:

$$f_i = \frac{N_i}{N} \tag{4.1}$$

donde:

- N_i es el número de veces que ocurre el símbolo $i \in (A,C,G,T)$ en la cadena. Solamente se tomaron en cuenta esos cuatro caracteres.
- N es el número total de símbolos o longitud de la cadena.

Se define la probabilidad de ocurrencia conjunta como en la ecuación 4.2:

$$f_{ij} = \frac{N_{ij}}{N-1} \tag{4.2}$$

donde:

- N_{ij} es el número de veces que ocurre el símbolo i seguido de j, con $i, j \in (A,C,G,T)$. Es decir, la frecuencia de los dinucleótidos formado por las combinaciones en pares de las cuatro bases, que son: AA, AC, AG, AT, CA, CC, CG, CT, GA, GC, GG, GT, TA, TC, TG, TT.

Se define el índice de correlación C_{ij} como en la ecuación 4.3 ([38]):

$$C_{ij} = f_{ij} - f_i * f_j \tag{4.3}$$

Así, se puede obtener la matriz $C \in M_{4\times4}$ con elementos C_{ij}. Esa matriz C de 4×4 se puede manejar como un vector de 16 elementos, acomodando los 4 renglones de la matrix en uno solo. Los componentes del vector representan el índice de correlación de cada pareja de nucleótidos. Ese vector es el que representa cada genoma.

Se realizó un programa en lenguaje C, `vector.c`, para obtener el vector de probabilidades tomando una secuencia de un genoma como entrada.

Se hizo el script `matrix.sh` para construir una matriz de datos, compuesta por todos los vectores de probabilidades de las secuencias de los genomas. Este script usa el programa `vector` en cada secuencia y manda los vectores a un mismo archivo de datos.

El archivo resultante es la matriz de datos de entrada para el SOM. En éste se agregó como primer renglón la dimensión de los vectores, en este caso 16, y también una etiqueta para cada genoma, al final de cada vector o renglón.

En el Apéndice A se describen los programas realizados para este trabajo.

4.3. Experimentación

Se implementó el algoritmo SOM en lenguaje C. En la sección som.c del Apéndice A se describen detalles del programa.

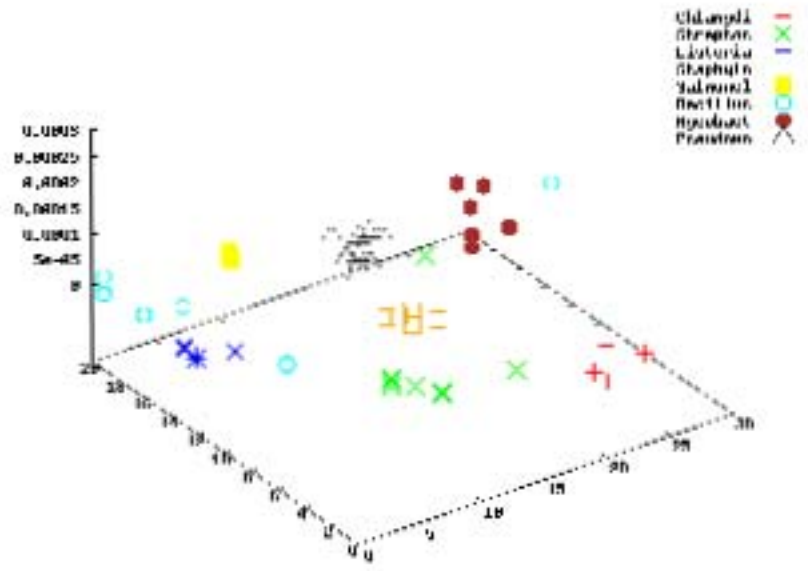


Figura 4.1: Gráfica de los resultados del programa `som` usando `gnuplot`

Para el programa `som` se usó una matriz de datos de entrada con una etiqueta de texto para cada genoma, correspondiente a las ocho primeras letras del nombre del organismo. Por ejemplo, *Streptoc* para *Streptococcus anginosus* y *Streptococcus pneumoniae*. Se usó así porque se pensó que esa etiqueta puede ayudar a diferenciar organismos de diferente especie.

El programa `som` tiene como salida un archivo con las columnas: un peso, coordenada x, coordenada y, y la etiqueta de la secuencia. Este archivo puede servir como entrada para hacer una visualización sencilla en 3D con el programa `gnuplot`[67], consistente en graficar las coordenadas x, y, y el peso como coordenada z.

Se ejecutó el programa `som` varias veces sobre la matriz de datos de entrada, con diferentes parámetros, tomando en cuenta algunas recomendaciones generales. Después de varias pruebas se definieron los parámetros siguientes: tamaño del mapa=30x20, epochs=50000, $\alpha = 0.05$, $\sigma = 5$. Los demás parámetros se tomaron por omisión, en particular, se dejó la selección aleatoria de la siguiente entrada, y la actualización de la vecindad de la unidad ganadora tomando un vecindario gaussiano.

Además se hicieron pruebas con otros programas ya implementados, como el

SOM_PAK[43]. En el Apéndice B se describe brevemente el paquete SOM_PAK y los programas incluidos. También se hicieron pruebas con los paquetes `som`, `kohonen`[63] y `SOMbrero`[4] del lenguaje R, y con el módulo PyMVPA de Python. Se decidió mostrar los resultados obtenidos con el paquete SOM_PAK, por los programas incluidos para generar las visualizaciones U-matrix y Sammon.

En las ejecuciones con los programas del paquete SOM_PAK se usó un archivo de datos de entrada con una etiqueta numérica. Los genomas se ordenaron alfabéticamente por nombre, y se numeraron consecutivamente. Con este etiquetado los organismos de un mismo género tenían identificadores numéricos cercanos.

En cuanto a los parámetros usados con el SOM_PAK, se hicieron varias pruebas, pero al final se usaron los mismos que con el programa `som`: tamaño del mapa=30x20, topología hexagonal, vecindad gaussiana; para la primera fase de entrenamiento: *epochs*=10000, $\alpha = 0.05$, radio=5; para la segunda fase: *epochs*=100000, $\alpha = 0.01$, radio=1. En este caso se usó un script en bash, `sompak.sh`, para las ejecuciones.

4.4. Visualización

En el caso de las corridas con el programa `som`, se usó el programa `gnuplot` para graficar los resultados obtenidos. En la figura 4.1 se muestra una gráfica de los resultados con su etiqueta correspondiente. Para cada etiqueta se usó un tipo de punto y color diferente.

Se generaron visualizaciones tipo U-matrix y de mapas Sammon de los resultados, usando los comandos *umat* y *sammon* del paquete SOM_PAK. En la figura 4.2 se observa un mapa tipo U-matrix. En la figura 4.3 se muestra el mapa Sammon. En cada caso se muestran los identificadores numéricos de los genomas como etiquetas.

En las figuras 4.4 y 4.5 se muestran los planos de los componentes de los vectores de entrada, numerados de izquierda a derecha y de arriba a abajo.

4.5. Resultados

En la visualización de los resultados obtenidos con el programa `som` implementado para este trabajo (que se ve en la figura 4.1), se observa que organismos de la misma especie, o género (resaltados por la etiqueta del mismo color), están claramente agrupados en el mapa. En particular, los organismos etiquetados con

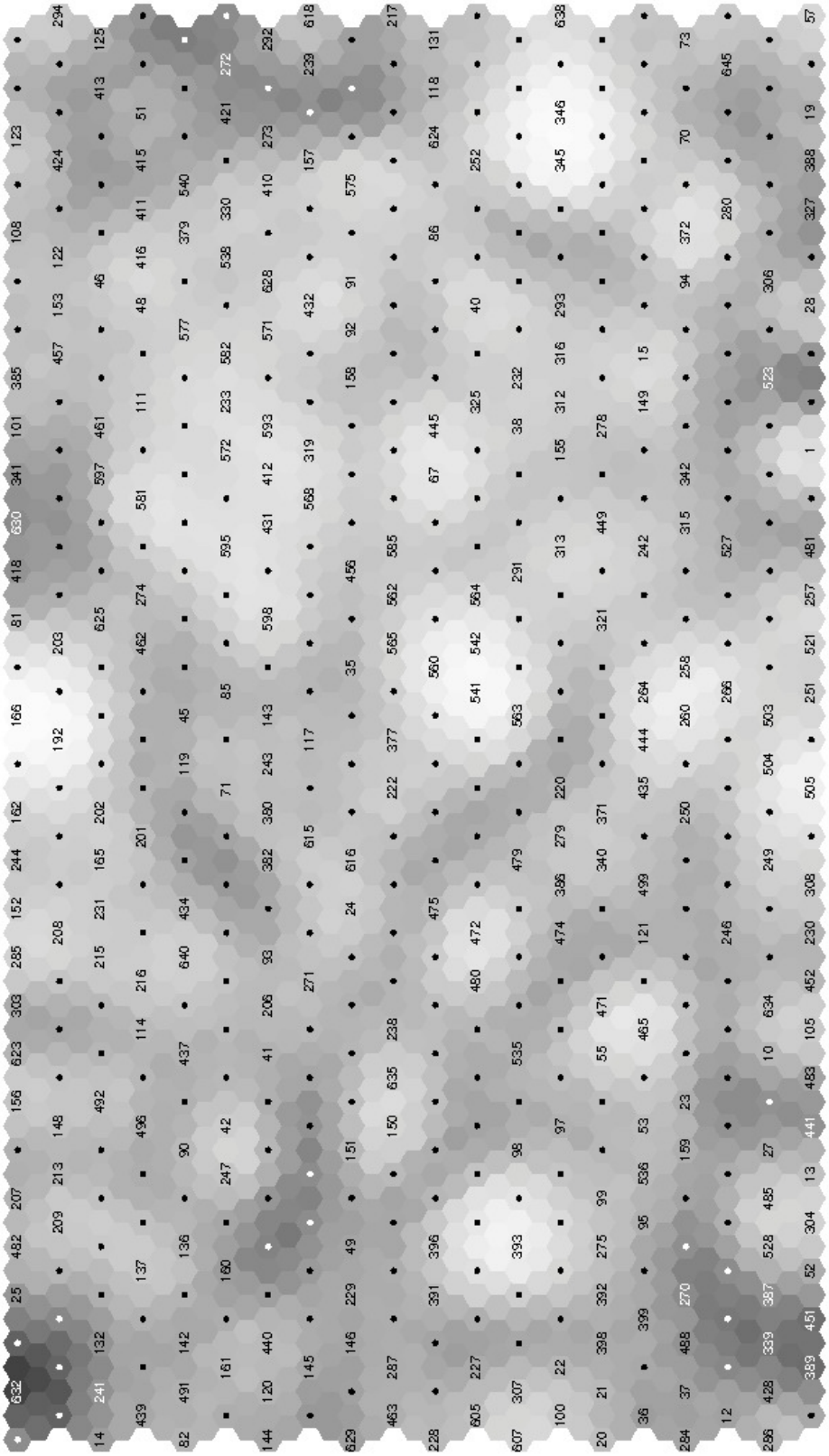


Figura 4.2: Mapa tipo U-matrix usando el SOM_PAK

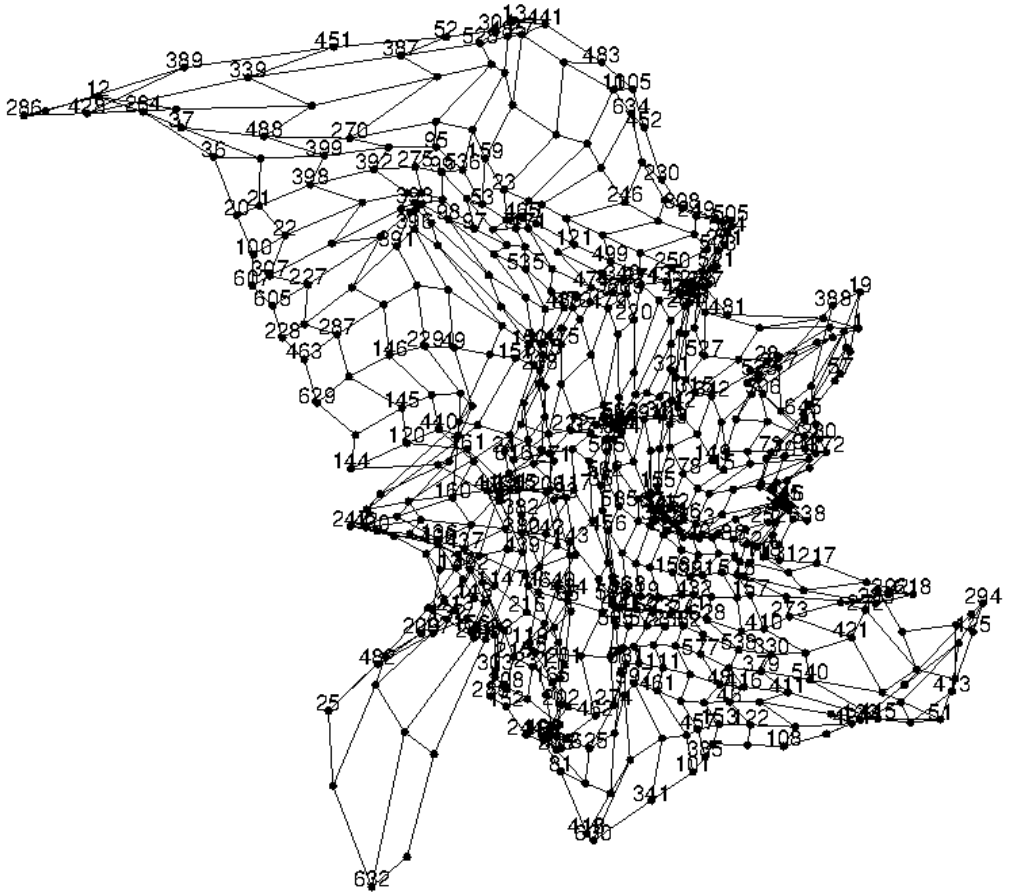
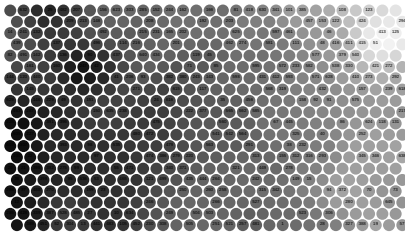
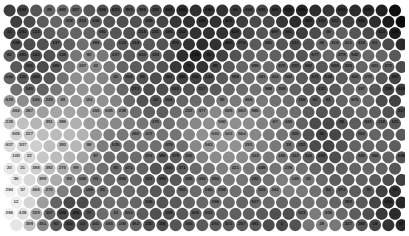


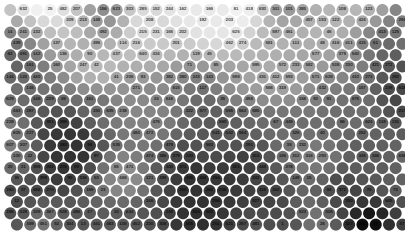
Figura 4.3: Mapa Sammon usando el SOM_PAK



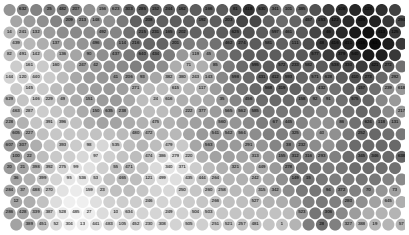
(a) Componente AA



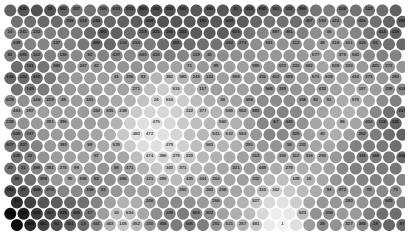
(b) Componente AC



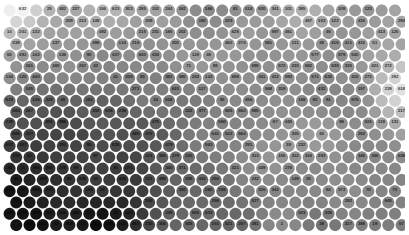
(c) Componente AG



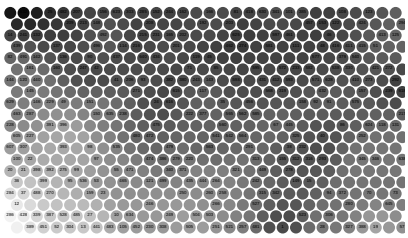
(d) Componente AT



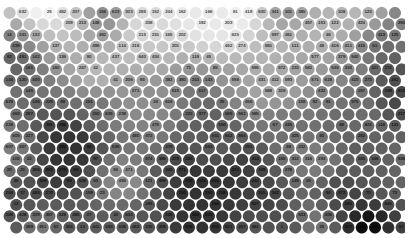
(e) Componente CA



(f) Componente CC

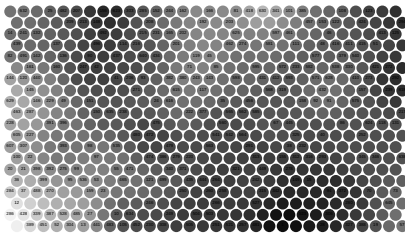


(g) Componente CG

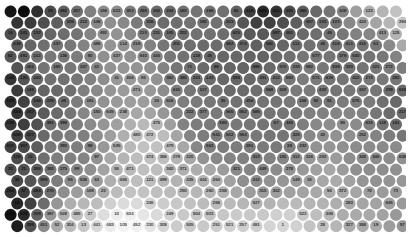


(h) Componente CT

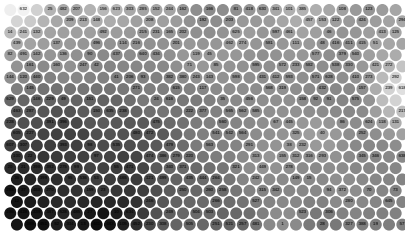
Figura 4.4: Planos de los componentes 1 al 8, numerados de izquierda a derecha y de arriba a abajo.



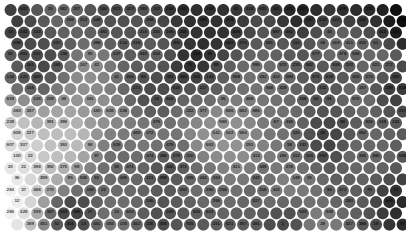
(a) Componente GA



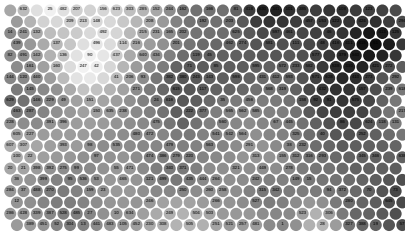
(b) Componente GC



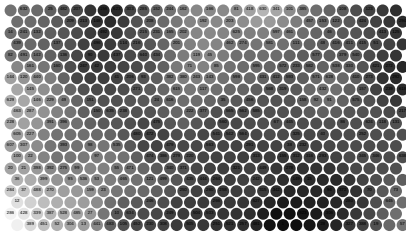
(c) Componente GG



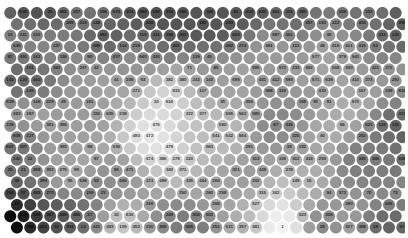
(d) Componente GT



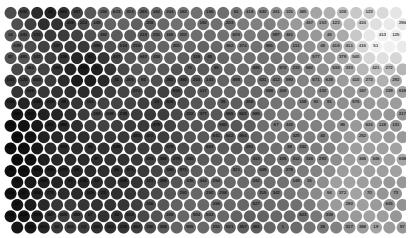
(e) Componente TA



(f) Componente TC



(g) Componente TG



(h) Componente TT

Figura 4.5: Planos componentes 9 al 16, numerados de izquierda a derecha y de arriba a abajo.

Chlamydi, *Listeria*, *Mycobact* y *Staphylo*. En el caso de *Bacillus*, se ven separados, esto podría ser porque uno corresponde a los organismos *Bacillus amyloliquefaciens* y el otro a los *Bacillus subtilis*. Esto podría confirmarse usando una etiqueta un poco más larga, o alguna otra forma de identificación de los elementos. El error de cuantificación fue de 0.000064.

En cuanto a los resultados obtenidos con el SOM_PAK, observando el mapa U-matrix de la figura 4.2 se encuentran varios grupos o *clusters* de genomas. Para cada grupo se muestran los elementos con su identificador numérico, y el nombre de la bacteria a la que corresponde el genoma. Los grupos detectados a simple vista se listan en la tabla 4.1. Se calculó el error de cuantificación de cada resultado usando el programa `qerror` del SOM_PAK. Se tomó como *mejor* resultado aquel con error menor. En el caso del resultado mostrado, el error de cuantificación es de 0.005596 por vector. Se puede aplicar un algoritmo de optimización (en este caso encontrar el mínimo) del error sobre todas las ejecuciones. El programa `vfind` del SOM_PAK hace justo eso, pero dejando los parámetros del SOM fijos.

Se puede ver que el SOM encontró grupos de genomas de bacterias del mismo género. Por ejemplo, es notorio en la lista de la tabla 4.1, en los grupos 2, 3 y 5, se aprecian bacterias de los géneros *Streptococcus*, *Staphylococcus* y *Escherichia coli*, respectivamente.

En los planos de los componentes, se puede observar que los componentes 1 (AA) y 16 (TT) están más correlacionados entre sí. Lo mismo para los componentes 6 (CC) y 11 (GG).

Estos resultados muestran que el algoritmo SOM puede efectivamente agrupar genomas de organismos similares usando una las frecuencias de dinucleótidos como representación de su estructura.

4.6. Consideraciones computacionales

Los genomas se descargaron usando el comando `wget`. El tamaño total de los 645 genomas de bacterias fue de aproximadamente 563 MB (Abril 2015). Cada archivo de secuencia está comprimido con `gzip`.

La ejecución de los programas se hizo en una computadora portátil con procesador Intel Core i5 a 2.67GHz con 4 núcleos de 64 bits, 8 GB de memoria RAM y sistema operativo Linux Fedora 20.

La compilación de programas en C se hizo con el compilador `gcc` 4.8.3. Para las gráficas se usó `gnuplot` 4.6.

El tiempo de ejecución para generar la matriz de vectores fue de alrededor de 3

minutos y solamente se hizo una vez. En este paso cada genoma se descomprime al vuelo para generar su vector.

Cada ejecución del script matrix.sh, que ejecuta el SOM y genera las visualizaciones con los programas del SOM_PAK, tardó aproximadamente 5 minutos con los parámetros ya definidos.

.

1.	
345	Listeria monocytogenes 08-5578, complete genome
346	Listeria monocytogenes ATCC 19117, complete genome
2.	
581	Streptococcus mitis B6, complete genome
595	Streptococcus salivarius CCHSS3, complete genome
597	Streptococcus sp. I-G2, complete genome
598	Streptococcus suis 98HAH33, complete genome
3.	
541	Staphylococcus aureus M1 complete genome
542	Staphylococcus aureus RF122, complete genome
560	Staphylococcus aureus subsp. aureus TW20, complete genome
563	Staphylococcus lugdunensis N920143, complete genome
564	Staphylococcus saprophyticus subsp. saprophyticus ATCC 15305, complete genome
4.	
166	Chlamydia trachomatis A2497, complete genome
192	Chlamydia trachomatis RC-F(s)/342, complete genome
5.	
258	Escherichia coli 042, complete genome
260	Escherichia coli APEC O78, complete genome
264	Escherichia coli O103:H2 str. 12009, complete genome
266	Escherichia coli str. K-12 substr. MDS42 DNA, complete genome
6.	
503	Salmonella bongori N268-08, complete genome
504	Salmonella bongori NCTC 12419, complete genome
505	Salmonella enterica subsp. enterica serovar Agona str. 24249, complete genome
7.	
472	Pseudomonas monteilii SB3078, complete genome
475	Pseudomonas protegens CHA0, complete genome
480	Pseudomonas sp. VLB120, complete genome
8.	
55	Azotobacter vinelandii CA6, complete genome
465	Pseudomonas aeruginosa B136-33, complete genome
471	Pseudomonas denitrificans ATCC 13867, complete genome
9.	
275	Gemmatimonas aurantiaca T-27, complete genome
391	Mycobacterium abscessus subsp. bolletii 50594, complete genome
392	Mycobacterium avium subsp. paratuberculosis MAP4, complete genome
393	Mycobacterium canettii CIPT 140060008 complete genome
396	Mycobacterium liflandii 128FXT, complete genome
10.	
150	Candidatus Saccharibacteria bacterium RAAC3_TM7_1, complete genome
151	Candidatus Saccharimonas aalborgensis, complete genome
635	Treponema pallidum subsp. pallidum SS14, complete genome

Tabla 4.1: Lista de grupos de genomas observados por el SOM

Conclusiones

En este trabajo se revisaron brevemente los antecedentes del área de la inteligencia artificial, y se trató de describir el origen y las ideas principales de las redes neuronales artificiales. Se describieron varios tipos comunes de redes neuronales, y en particular los mapas autoorganizados, una red basada en aprendizaje no supervisado. El algoritmo de mapas autoorganizados es ideal para el análisis de datos, en especial la agrupación, y su visualización.

Se investigaron los conceptos biológicos básicos del área conocida hoy como ciencias genómicas: bases nucléicas, genes, cromosomas, genomas. Se describió la representación FASTA para las secuencias genómicas. Se describieron algunos proyectos genómicos, relacionados estrechamente con la estadística y las ciencias de la computación, y se resaltó la importancia del análisis de los datos genómicos. Lo anterior se hizo para conocer la terminología y tener alguna idea de los trabajos en el área.

Como propósito principal de este trabajo se aplicó el algoritmo de mapas autoorganizados sobre una representación estructural de un conjunto de genomas completos de bacterias. La representación usada calcula las frecuencias de dinucléotidos en la secuencia, y define un índice de correlación para cada pareja de nucleótidos.

En la experimentación se realizaron los programas para generar la representación de cada genoma, el propio algoritmo SOM, varios scripts para la ejecución de los anteriores, así como scripts para una graficación sencilla de resultados. Además se exploraron diferentes herramientas ya disponibles públicamente que implementan el algoritmo SOM, como por ejemplo el SOM_PAK. Se decidió usar el SOM_PAK para la aplicación del SOM porque ha sido desarrollado por el mismo equipo del autor del algoritmo, y tiene programas que generan buenas visualizaciones, en particular U-matrix y Sammon.

Con la aplicación del SOM y la visualización de los resultados se encontraron resultados que confirman la idea de la similitud estructural de genomas de orga-

nismos similares. Esto es, genomas de organismos del mismo género, por ejemplo *Streptococcus*, quedaron cercanos en el mapa. Se detectaron varios grupos más de genomas del mismo género.

En cuanto a trabajos relacionados se podría hacer un análisis similar para otros genomas completos, por ejemplo para el conjunto de genomas de los cromosomas del ser humano, o para el conjunto de genomas de algunos organismos eucariontes. Otro camino es analizar los genómas de plásmidos de diferentes organismos y buscar relaciones entre ellos. O bien, genomas incompletos o pedazos de genoma antes de ser ensamblados. O se podría hacer la representación de los genomas usando trinucleótidos en lugar de dinucleótidos, lo que aumentaría la dimensionalidad de los datos de entrada. Otra cosa a explorar puede ser el tomar en cuenta las bases desconocidas o ambiguas, ver qué tanto porcentaje de la secuencia ocupan, y ver si afecta o no el ignorarlas. O tal vez predecirlas con base en la agrupación resultante. Una cosa más que se puede hacer es tomar genomas no sólo de bacterias, si no también de *archaea* y de eucariontes, y aplicar el algoritmo sobre todos, para tratar de ver quiénes están más cercanos entre sí.

Computacionalmente se puede paralelizar la generación de los vectores, separando el conjunto de genomas. También se puede paralelizar el conjunto de ejecuciones del SOM, separándolas definiendo diferentes parámetros para cada una, con el fin de encontrar los parámetros adecuados.

Apéndice A

Programas realizados

El código fuente de los programas realizados para este trabajo se puede encontrar en el sitio web https://github.com/chuseq/som [20].

El código fuente se publica con un objetivo puramente académico. El lector interesado puede usar el código completo, o parte de, bajo su propio riesgo.

vector.c

Programa en lenguaje C que lee una secuencia de DNA en format fasta y genera un vector de dimensión 16, calculando frecuencias de dinucleótidos.

Uso: `$ zcat SECUENCIA.fna.gz | ./vector`

matrix.sh

Script en bash que lee todos los archivos de secuencias en formato FASTA en un directorio, ejecuta el programa `vector` sobre cada uno, y genera una matriz de dimensión $n \times 16$.

Uso: `$ bash matrix.sh`

Variables en el script:

- `GENOMES`: la ruta al subdirectorio que contiene los archivos de secuencias
- `MATRIX`: el nombre del archivo de salida

som.c

Programa en lenguaje C que implementa el algoritmo SOM. Esta es una implementación básica y general, que se puede usar con cualquier conjunto de datos. El programa recibe como entrada un archivo que contiene en la primera línea la dimensión de los datos d y a continuación n vectores de dimensión d, uno por línea. Genera un archivo de salida con la lista de coordenadas para cada uno de los n vectores.

Uso: `$ ./som [-x NROWS] [-y NCOLS] [-e EPOCHS] [-a ALPHA0] [-s SIGMA0] [-o ARCHIVOSALIDA] [-r] [-l LOG] ARCHIVOENTRADA`

- `[-x NROWS] [-y NCOLS]` dimensión del mapa, deben ser valores enteros.
- `[-e EPOCHS]` número de épocas, es un valor entero.
- `[-a ALPHA0]` taza de aprendizaje inicial, es un valor real.
- `[-s SIGMA0]` sigma inicial para calcular el radio, es un valor real.
- `[-o ARCHIVOSALIDA]` nombre del archivo de salida.
- `[-r]` si se hará selección aleatoria de los vectores de entrada en cada epoch. si no se especifica, por omisión se hace selección secuencial.
- `[-l LOG]` si se guarda registro de la ejecución. Posibles valores: 0, 1. Por omisión sí se guarda registro.
- `ARCHIVOENTRADA` nombre del archivo de la matriz de datos de entrada

En el caso de este trabajo el archivo de entrada es el generado por el script `matrix.sh`.

plot.sh

Script en bash que lee el archivo de coordenadas generado por el SOM, ordena de acuerdo a las etiquetas de los datos, y genera un script para `gnuplot` [67].

Uso: `$ bash plot.sh RESULTADOSOM`

En este caso el archivo de entrada es el archivo de resultados generado por el programa `som`.

Variables en el script:

- `NUMCLUST`: número de diferentes grupos o colores a usar

- `PAUSE`: si es 0, guarda la gráfica directamente a un archivo de imagen PNG; si es 1, deja la ventana de gnuplot abierta
- `LABELS`: si es 0, grafica los puntos solamente; si es 1, coloca las etiquetas junto a los puntos

wget.sh

Script en bash para descargar los archivos de las secuencias de Bacterias en formato FASTA desde el sitio FTP del NCBI [16]. Nota: en abril de 2015 se descargaron 645 archivos que ocupan aproximadamente 560MB de espacio en disco.

sompak.sh

Script en bash que ejecuta los programas del paquete SOM_PAK [43]. Ver también el Apéndice B.

Apéndice B

Paquete SOM_PAK

El paquete SOM_PAK fue desarrollado por la Universidad Tecnológica de Helsinki, Finlandia ([43]) en 1992, por un equipo dirigido por el mismo Teuvo Kohonen, autor del algoritmo SOM. En este trabajo se usó la versión 3.1 publicada en 1995 del SOM_PAK.

El paquete SOM_PAK contiene todos los programas necesarios para la aplicación del algoritmo SOM en cualquier conjunto de datos y para visualizaciones complejas. Los programas están escritos en lenguaje C. Se deben compilar para generar los programas ejecutables.

Los datos de entrada se deben guardar en un archivo de texto plano. En éste la primera línea está reservada para la información que usará el algoritmo y debe tener los siguientes datos:

- la dimensión, por ejemplo, en el caso de este trabajo, $n = 16$
- la topología, puede ser *hexa* o *rect*
- la dimensión x del mapa
- la dimensión y del mapa
- tipo de vecindad, puede ser *bubble* o *gaussian*

Solamente el primer dato es obligatorio. Los demás son opcionales, y también se pueden pasar como valores de parámetros en las opciones de los programas.

A partir de la segunda línea, deben ir los valores de los vectores de entrada, seguidos de un identificador alfanumérico. En este trabajo son vectores de $n = 16$ números reales, seguidos de un identificador numérico.

Los programas disponibles en el paquete SOM_PAK son:

- `randinit`: inicia el mapa, puede recibir las dimensiones *x* y *y* del mapa, la topología, el tipo de vecindad y la matriz de entrada
- `vsom`: lleva a cabo el entrenamiento, recibe la matriz de entrada, el número de iteraciones o *epochs*, la taza de aprendizaje α, y el radio
- `qerror`: cálculo del error de cuantificación promedio
- `vcal`: para la calibración del mapa, previa a la visualización, esto es, etiqueta las unidades del mapa con su identificador
- `visual`: genera una lista de coordenadas de 3 dimensiones, las coordenadas x,y de la unidad ganadora, el error promedio y la etiqueta
- `sammon`: genera el mapa Sammon
- `umat`: genera el mapa U-matrix
- `planes`: genera las *n* representaciones o planos, uno por cada componente de los vectores de entrada
- `vfind`: este programa hace automáticamente varias ejecuciones de los programas anteriores, con parámetros fijos definidos al inicio, y encuentra el mejor mapa de acuerdo al criterio de un menor error de cuantificación

Para conocer con más detalle los programas del paquete SOM_PAK se puede consultar [43] y en particular [44].

Apéndice C

Genomas de bacterias utilizados

El listado de los genomas de bacterias descargados del sitio FTP [16] utilizados en este proyecto se puede encontrar en el sitio web [20].

Acceso directo: https://github.com/chuseq/som/blob/master/genomas.txt

Bibliografía

[1] Takashi Abe, Shigehiko Kanaya, Makoto Kinouchi, Yuta Ichiba, Tokio Kozuki, and Toshimichi Ikemura. A novel bioinformatic strategy for unveiling hidden genome signatures of eukaryotes: self-organizing map of oligonucleotide frequency. *Genome Informatics*, 13:12–20, 2002. 3.7, 4.1

[2] Vicente Arnau and Ignacio Marín. Análisis comparativo y exhaustivo de secuencias de longitud 12 nucleótidos en cromosomas humanos. *XIV Jornadas de paralelismo*, 2003. 4.1

[3] William Ross Ashby. Principles of the self-organizing dynamic system. *The journal of general psychology*, 37(2):125–128, 1947. 2.1

[4] Laura Bendhaiba, Julien Boelaert, Madalina Olteanu, and Nathalie Villa-Vialaneix. Sombrero: Som bound to realize euclidean and relational outputs. https://cran.r-project.org/web/packages/SOMbrero. 4.3

[5] J. Blackmore and Risto Miikkulainen. Incremental grid growing: Encoding high-dimensional structure into a two-dimensional feature map. 1992. 2.4

[6] Jon Bohlin and Eystein Skjerve. Examination of genome homogeneity in prokaryotes using genomics signatures. *PLoS ONE*, 2009. 4.1

[7] Terence A. Brown. *Genomes*. John Wiley and Sons Inc., 1999.

[8] Wikimedia Commons. Estructura química del adn, 2008. https://upload.wikimedia.org/wikipedia/commons/4/4e/DNA_chemical_structure_es-2008-08-01.svg. 3.2

[9] International Human Genome Sequencing Consortium. Initial sequencing and analysis of the human genome. *Nature*, 409:860–921, 2001. 3.7

[10] The ENCODE Project Consortium. The ENCODE (ENCyclopedia Of DNA Elements) project. *Science*, 306(5695):636–640, 2004. 3.7

[11] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989. 1.3

[12] Ralf Dahm. Friedrich miescher and the discovery of {DNA}. *Developmental Biology*, 278(2):274 – 288, 2005. 3.1

[13] Athanasios S. Drigas and John Vrettaros. Using the self-organizing map (SOM) algorithm, as a prototype e-content retrieval tool. *Springer-Verlag Berlin Heidelberg*, 2008. 2.4

[14] Miniespacio educativo. Prof. Sofia Vitale. Cariotipo humano normal femenino, 2009. http://4.bp.blogspot.com/_5KSanhHpTHY/SmDw0BtZlbI/AAAAAAAAAq8/A7SQwIz6LIc/s320/10-17-G.gif. 3.4

[15] R.A. Fisher. The use of multiple measurements in taxonomic problems. 7(2):179–188, 1936. 2.6

[16] National Center for Biotechnology Information of the National Institutes of Health. Bacterial genomes (abril 2015). ftp://ftp.ncbi.nih.gov/genomes/Bacteria. 4.2, A, C

[17] Andrew J. Gentles and Samuel Karlin. Genome-scale compositional comparison in eukaryotes. *Genome Research*, 11:540–546, 2001. 4.1

[18] Carlos Gershenson. La complejidad como paradigma científico. adaptación y autoorganización, 2012. http://turing.iimas.unam.mx/ cgg/teach/Pamplona/01-Cx.pdf. 2.1

[19] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011. 1.3

[20] Alfredo Hernández. Basic self organizing map implementation for genome analysis. https://github.com/chuseq/som. A, C

[21] Javier Herrero and Joaquín Dopazo. Combining hierarchical clustering and self-organizing maps for exploratory analysis of gene expression patterns. *Journal of Proteome Research*, pages 467–470, 2002. 3.7, 4.1

[22] John Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79(8):2554,2558, 1982. PMID:6953413. 1.3

[23] http://biologia.laguia2000.com/. Adn, 2010. http://biologia.laguia2000.com/wp-content/uploads/2010/03/adn_thumb.gif. 3.1

[24] Shigehiko Kanaya, Makoto Kinouchi, Takashi Abe, Yoshihiro Kudo, Yuko Yamada, Tatsuya Nishi, Hirotada Mori, and Toshimishi Ikemura. Analysis of codon usage diversity of bacterial genes with a self-organizing map (SOM): characterization of horizontally transferred genes with emphasis on the *E. coli* o157 genome. *Gene*, 276:89–99, 2001. 3.7, 4.1

[25] J.A. Kangas, Teuvo Kohonen, and J.T. Laaksonen. Variants of self-organizing maps. *IEEE Transactions on Neural Networks*, 1:93–99, 1990. 2.4, 2.4

[26] Samuel Kariin and Chris Burge. Dinucleotide relative abundance extremes: a genomic signature. *Trends in Genetics*, 11(7):283 – 290, 1995. 4.1

[27] Samuel Karlin, Alan M. Campbell, and Jan Mrázek. Comparative DNA analysis across diverse genomes. *Annu. Rev. Genet.*, pages 185–225, 1998. 4.1

[28] S Kaski. Data exploration using self-organizing maps. *Acta Polytechnica Scandinavica, Mathematics, Computing and Management in Engineering Series*, 82, 1997. 2.4

[29] Teuvo Kohonen. Self-organized formation of topologicaly correct feature maps. *Biol. Cybern. 43*, pages 59–69, 1982. 2.2

[30] Teuvo Kohonen. Learning vector quantization. *Neural Networks*, page 303, 1988. 1.1

[31] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990. 2.4

[32] Teuvo Kohonen. Self-organizing maps. 1995. 2.4

[33] Richard D. Lawrence, George S. Almasi, and Holly E. Rushmeier. A scalable parallel algorithm for self-organizing maps with applications to sparse data mining problems. *Data Mining and Knowledge Discovery*, 3(2):171–195, 1999. 2.4

[34] Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6):861–867, 1993. 1.3

[35] David J. Lipman and William R. Pearson. Rapid and sensitive protein similarity searches. *Science*, 227(4693):1435–1441, 1985. 3.6

[36] Elizabeth Louie, Jurg Ott, and Jacek Majewski. Nucleotide frequency variation across human genes. *Genome Research*, 13:2594–2601, 2003. 4.1

[37] Warren S. McCulloch and Walter A. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of mathematics and biophysics*, 5:115–133, 1943. 1.1, 12

[38] Luis Medrano, Germinal Cocho, Pedro Miramontes, and J.L. Rius. The effect of DNA stability on mutation and sequence evolution. *Evolutionary Theory*, pages 249–258, 1994. 4.2

[39] Gregor Mendel. Experiments in plant hybridization. 1865. (document), 3.1

[40] Marvin Minsky and Seymour Papert. *Perceptrons*. 1969. 1.3

[41] Pedro Miramontes, Luis Medrano, C. Cerpa, Robert Cedergren, Gerardo Ferbeyre, and Germinal Cocho. Structural and thermodynamic properties of DNA uncover different evolutionary histories. *Journal of Molecular Evolution*, pages 698–704, 1995. 4.1

[42] National Institutes of Health. Growth of genbank and wgs. http://www.ncbi.nlm.nih.gov/genbank/statistics. 3.6

[43] Helsinki University of Technology. Som_pak. http://www.cis.hut.fi/research/som-research/nnrc-programs.shtml. 2.6, 4.3, A, B

[44] Helsinki University of Technology. Som_pak: The self-organizing map program package, 1995. http://www.cis.hut.fi/research/som_pak/som_doc.ps. B

[45] IUPAC-IUB Comission on Biochemical Nomenclature (CBN). Abbreviations and symbols for nucleic acids, polynucleotides, and their constituents. *Biochemistry*, 9(20):4022–4027, 1970. 3.6

[46] David B Parker. Learning logic. 1985. 1.3

[47] Georg Pölzlbauer. *Survey and comparison of quality measures for self-organizing maps*. 2004. 2.4, 2.4

[48] UCI Machine Learning Repository. Iris data set. https://archive.ics.uci.edu/ml/datasets/Iris/. 2.6

[49] Elaine Rich and Kevin Knight. *Artificial Intelligence*. Mc Graw Hill, 2a edition, 1991. 1.2

[50] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, November 1958. 1.3

[51] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986. 1.3

[52] Stuart Rusell and Peter Norvig. *Inteligencia Artificial. Un enfoque moderno*. Prentice Hall, 1996. (document), 1.1, 1.3

[53] Mark P. Sawicki, Ghassan Samara, Michael Hurwitz, and Edward Jr Passaro. Human genome project. *American Journal of Surgery*, 1993. PMID:8427408. (document)

[54] Erwin Schrondinger. *What is life?* 1944. 3.1

[55] Gilberto Sotolongo-Aguilar, María Guzmán-Sánchez, and Humberto Carrillo. ViBlioSOM: Visualización de información bibliométrica mediante el mapeo autoorganizado. *Rev. Esp. Doc. Cient.*, pages 477–484, 2002. 2.7

[56] Pablo Tamayo, Donna Slonim, Jill Mesirov, Qing Zhu, Sutisak Kitareewan, Ethan Dmitrovsky, Eric S. Lander, and Todd R. Golub. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proc. Natl. Acad. Sci. USA*, 96:2907–2912, 1999. 3.7, 4.1

[57] Alan M. Turing. On computable numbers with an application on the entscheidungsproblem. pages 230–265, 1936. 1.1

[58] Alan M. Turing. Computing machinery and intelligence. *Mind*, 59:433–460, 1950. 1.1

[59] Alfred Ultsch and H. Peter Siemo. Kohonen's self organizing feature maps for exploratory data analysis. *Proc. INNC 90, International Neural Network Conference*, pages 305–308, 1990. 2.5

[60] Juha Vesanto and Esa Alhoniemi. Clustering of the self-organizing map. *IEEE Transactions on neural networks*, 11(3):586–600, 2000. 2.4

[61] Heinz von Foerster. On self-organizing systems and their environments. 1960. 2.1

[62] James D. Watson and Francis H. Crick. Molecular structure of nucleic acids. a structure for deoxyribose nucleic acid. *Nature*, 1953. (document), 3.1

[63] Ron Wehrens. kohonen: Supervised and unsupervised self-organising maps. https://cran.r-project.org/web/packages/kohonen. 4.3

[64] Paul Werbos. Beyond regression: New tools for prediction and analysis in the behavioral sciences. 1974. 1.3

[65] Alfred North Whitehead and Bertrand Russell. *Principia mathematica*, volume 2. University Press, 1912. 1.1

[66] Norbert Wiener. *Cybernetics*. 2 edition, 1961. 2.1

[67] Thomas Williams and Colin Kelley. Gnuplot. http://www.gnuplot.info. 4.3, A

[68] C. Woese, O. Kandler, and M. Wheelis. Towards a natural system of organisms: proposal for the domains archaea, bacteria and eucarya. *Proc Natl Acad Sci*, 1990. 3.5

Índice alfabético